The suitability of conceptual graphs in strategic management accountancy

This item was submitted to Loughborough University's Institutional Repository by the/an author.

Additional Information:

- A Doctoral Thesis. Submitted in partial fulfillment of the requirements for the award of Doctor of Philosophy of Loughborough University.

Metadata Record: https://dspace.lboro.ac.uk/2134/7512

Publisher: © Simon Polovina

Please cite the published version.
This item is held in Loughborough University’s Institutional Repository (https://dspace.lboro.ac.uk/) and was harvested from the British Library’s EThOS service (http://www.ethos.bl.uk/). It is made available under the following Creative Commons Licence conditions.

For the full text of this licence, please go to: http://creativecommons.org/licenses/by-nc-nd/2.5/
The Suitability of Conceptual Graphs in Strategic Management Accountancy.

by

Simon Polovina, BA (Hons.), MSc

A Doctoral Thesis

Submitted in partial fulfilment of the requirements for the award of

Doctor of Philosophy of the Loughborough University of Technology.

August, 1993

© by Simon Polovina, 1993

The author certifies that he is responsible for the work submitted in this thesis, that the original work is his own except as specified in acknowledgements or in footnotes, and that neither the thesis nor the original work contained therein has been submitted to this or any other institution for a higher degree.
Contents

Abstract vii

Preface and Acknowledgements viii

1 The Usefulness of Knowledge-Based Techniques to the Management Accountant 1

1.01 Introduction 1
1.02 The Need to Consider New Approaches 1
1.03 Adopting the Knowledge-Based Approach 7
1.04 The Nature of Existing Expert Systems 10
1.05 Case Study 1: 'XLSIOR' Reinsurance Expert System 17
1.06 Experiences from Existing Expert Systems 21
1.07 Case Study 2: Systems Dynamics 27
1.08 A Positive Comment about Systems Dynamics 32
1.09 Concluding Remarks 33

2 The Usefulness of Conceptual Graphs to the Strategic Management Accountant 35

2.01 Introduction 35
2.02 The Worth of Problem Structuring Aids 35
2.03 The Need for Structured Diagram Approaches 37
2.04 The Limits and Scope of Structured Diagram Approaches 38
2.05 Case Study 3: An Accountant's Flowchart for Relevant Values 39
2.06 The Particular Problem With the Accountant's Flowchart 40
2.07 Comments in Respect of All the Above 40
2.08 The Candidate Solution: Conceptual Graphs 41
2.09 Introducing Conceptual Graphs 41
2.10 Concepts and Relations 43
2.11 The Type Hierarchy 46
2.12 Projection 48
2.13 Combining Graphs 49
2.14 Inference 50
2.15 Case Study 4: Scrupulous Chemicals Co. (Relevant Values) 55
2.16 Comments Arising from the Scrupulous Chemicals Case Study 62
### 3 Conceptual Graphs and Cognitive Mapping

- **3.01 Introduction**
- **3.02 The Example Problem**
- **3.03 The Cognitive Map for the Example Problem**
- **3.04 Modelling the Poles in Conceptual Graphs**
- **3.05 Refining the Graphs**
- **3.06 Generalising the Model**
- **3.07 Modelling the Undefined Poles in Conceptual Graphs**
- **3.08 Modelling the Links in Conceptual Graphs**
- **3.09 Modelling the Other Knowledge in the Example Problem**
- **3.10 Allowing Inferencing**
- **3.11 Visual Cues**
- **3.12 Merging Graphs**
- **3.13 Comments About Cognitive Mapping in Conceptual Graphs**
- **3.14 Concluding Remarks**

### 4 Conceptual Graphs and Economic Accounting

- **4.01 Introduction**
- **4.02 The Essence of Economics**
- **4.03 Supply and Demand**
- **4.04 Case Study 5: Supply and Demand in Conceptual Graphs**
- **4.05 Comments Arising From Supply and Demand, & Case Study 5**
- **4.06 The Essence of Accounting**
- **4.07 The Double Entry Bookkeeping Model**
- **4.08 Problems with the Bookkeeping Model**
- **4.09 The Events Accounting Model**
- **4.10 The Events Accounting Model as Conceptual Graphs**
- **4.11 Duality**
- **4.12 Intangible Values**
- **4.13 Comments on the Events Accounting Model**
- **4.14 Introducing the Cognitive Mapping Dimension**
- **4.14 A Bridge Between Events Accounting and Cognitive Mapping**
- **4.15 Concluding Remarks**
5 Determining the Methodology to Test the Conceptual Graphs Approach

5.01 Introduction
5.02 The Practicalities of Eliciting and Reviewing Knowledge
5.03 Clarifying The Appropriateness of Conceptual Graphs
5.04 The Need for Conceptual Graphs Software
5.05 Existing Conceptual Graph Software, e.g. CAMES
5.06 JEHCGP: A More Promising Conceptual Graph Processor
5.07 Events at a Usability Pilot Study Based on JEHCGP
5.08 The Conceptual Analysis and Review Environment, CARE
5.09 The CARE Software, and its Design Issues
5.10 Concluding Remarks

6 The Usability of Conceptual Graphs by Strategic Management Accountants

6.01 Introduction
6.02 Responses to CARE at a Conceptual Graphs Workshop
6.03 Comments on Workshop Responses
6.04 The Way Forward from the Workshop Responses
6.05 Enhancing CARE from Version 1.0 to 1.1
6.06 Responses to CARE by Expert 1, Session 1, a.m.
6.07 Preparing the Ground for the p.m. Session
6.08 Responses to CARE by Expert 1, Session 1, p.m.
6.09 Preparing the Ground for Another Session with Expert 1
6.10 Enhancing CARE from Version 1.1 to 1.1a
6.11 Responses to CARE by Expert 1, Session 2
6.12 Comments on the Sessions with Expert 1
6.13 Enhancing CARE from Version 1.1a to 1.2
6.14 The Way Forward from the Sessions with Expert 1
6.15 Responses to CARE by Expert 2
6.16 Comments on the Session with Expert 2
6.17 The Way Forward from the Session with Expert 2
6.18 Responses to CARE by Expert 3
6.19 Comments on the Session with Expert 3
6.20 Enhancing CARE from Version 1.2 to 1.3
6.21 The Way Forward from the Session with Expert 3
6.22 Taking CARE to Senior Accountants at Melton
6.23 Background to the Melton Participants
6.24 Revised Format for Melton Session 169
6.25 Responses to CARE at Melton Session 170
6.26 Melton Participant Disposition Towards Conceptual Graphs 181
6.27 Analysis of Questionnaire From Melton Session 185
6.28 About Written Comments Added by The Melton Participants 192
6.29 Comments on the Melton Session 194
6.30 Summary of Evaluation Sessions with Domain Expert Users 194
6.31 Concluding Remarks 196

7 Conclusions 198
7.01 The Hypothesis 198
7.02 Retracing the Events 198
7.03 Suggested Implications 200

Bibliography 202

The Appendices 220
A/01: Pilot Study Paper 220
A/02: Pilot Study Instructions I 237
A/03: Pilot Study Instructions II 238
A/04: Prolog Code for the CARE Interface 248
A/05: Guideline Procedures in Adapting JEHCNP Program Itself for CARE. 295
A/06: Example CARE Screens (version 1.0) 297
B/01: Attendees at the Seventh Annual Workshop on Conceptual Graphs 299
B/02: Workshop CARE Quick Reference I 301
B/03: Workshop CARE Quick Reference II 302
B/04: Workshop CARE Questionnaire 303
B/05: The Problem Given by Expert 1 304
B/06: The Initial Conceptual Graphs Representing Expert 1’s Chosen Problem 307
B/07: Pro Forma ‘If-Then’ Graphs 311
B/08: The Subsequent Conceptual Graphs Representing Guideline ‘a’ 312
B/09: The Many ‘if-then’ Rule Equivalents of Guideline ‘a’ 328
B/10: Graphs Partially Analysing the ‘Other factor’ Dimension in the Expert 1 Problem 331
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B/11</td>
<td>Pro Forma Browse Graphs</td>
<td>335</td>
</tr>
<tr>
<td>B/12</td>
<td>Melton Questionnaire</td>
<td>336</td>
</tr>
<tr>
<td>B/13</td>
<td>CARE screens from Melton session</td>
<td>342</td>
</tr>
<tr>
<td>B/14</td>
<td>Summary of Questionnaire Responses from Melton Session</td>
<td>343</td>
</tr>
<tr>
<td>B/15</td>
<td>Significant Spearman \ Kendall Correlations from Melton Session</td>
<td>345</td>
</tr>
<tr>
<td>C/01</td>
<td>&quot;Towards the Thought Machine&quot;</td>
<td>349</td>
</tr>
<tr>
<td>C/02</td>
<td>&quot;Tomorrow's Spreadsheets: Conceptual Graphs as the Knowledge Based Decision Support Tool for the Management Accountant&quot;</td>
<td>352</td>
</tr>
<tr>
<td>C/03</td>
<td>&quot;An Introduction to Conceptual Graphs&quot;</td>
<td>364</td>
</tr>
<tr>
<td>C/04</td>
<td>&quot;Bridging Accounting and Business Strategic Planning Using Conceptual Graphs&quot;</td>
<td>373</td>
</tr>
<tr>
<td>C/05</td>
<td>&quot;The Economics Of Supply And Demand: An Important Challenge For Conceptual Graphs&quot;</td>
<td>384</td>
</tr>
</tbody>
</table>
Abstract

The hypothesis of the research is "conceptual graphs are a suitable knowledge-based decision support tool for use by management accountants in strategic planning", explained as follows.

Knowledge-based approaches can help accountants apply their skills in the direction of strategic management problems. Such problem domains cannot be modelled effectively by computer alone, hence we are only interested in those advanced knowledge-based methodologies that can be adequately reviewed by strategic management accountants in the light of their own continually changing tacit and implicit knowledge.

Structured diagram techniques, such as flowcharting, are well known by accountants and are a clearly understandable yet important aid in problem review. Apart from being founded on a logically complete reasoning system, the knowledge-based methodology of conceptual graphs was formulated to be an enhancement of these other methods. Furthermore the graphical form of conceptual graphs enjoy an apparent similarity to the 'negating' brackets in the accountant's traditional bookkeeping model.

After conducting a comparative study with two similar methodologies in current use showing the technical advantages of conceptual graphs, the Conceptual Analysis and Review Environment computer software was devised and implemented. CARE was used to test the accepted graphical form of conceptual graphs through a series of user evaluation sessions. The evaluations started out with subjects from the conceptual graphs community itself, then key business school staff, and culminated in a session with senior practising accountants. In addition, CARE was enhanced iteratively in accordance with the results of each evaluation session.

Despite their strong prima facie attractiveness and positive response from the conceptual graphs community session, as the user evaluations progressed it became increasingly evident that the inherent complexity of conceptual graphs fundamentally undermined them as a viable tool, other than for very trivial problems well below the level needed to be viable for strategic management accountancy. Therefore the original contribution of this research is that its hypothesis turns out to be false.
Preface and Acknowledgements

The hypothesis of this research is that "conceptual graphs are a suitable knowledge-based decision support tool for use by management accountants in strategic planning", the background to which is as follows.

In approximately six years of being an accountant in industry, I became increasingly aware of the large part that the computer was going to play in the accounting profession. Whilst in the profession, I had seen the electronic spreadsheet transform one of the accountant’s longest and most tedious tasks, namely adding rows and columns of figures with no more than a calculator, into an elementary exercise. Furthermore, the accountant’s use of the powerful computer spreadsheet had opened up a new, and much more meaningful, range of financial information for the organisation in which that accountant was employed. I was so interested in the significance of the computer for the accountant, that I undertook the one year MSc Information Technology course at Loughborough University of Technology.

During that course I became aware of the ‘artificial intelligence’ field. From the work in this area, I saw that the computer could benefit the accountant in more ways than allowed by numeric and data-based analysis. Potentially, artificial intelligence could enable the accountant in tackling the qualitative dimensions of business too. Thereby the accountant’s soundly developed techniques could be extended into the top-level area of strategic business knowledge, whose qualitatively-based nature had essentially caused this problem domain to remain outside the scope of the already mentioned numeric and data-based computational methods. In pursuit of the above, I conducted this doctoral research project of which this thesis is the product.

In outline the thesis is laid out as follows. To start with, Chapter 1 identified how advanced knowledge-based approaches can help accountants apply their skills towards strategic management problems. Chapter 2 targeted conceptual graphs as the particular knowledge-based approach for these strategic management accountants. Chapters 3 and 4 compared conceptual graphs on

---

1 In addition to the course lectures on the subject, I further educated myself about artificial intelligence through my MSc project in this area (Polovina, 1989)
technical grounds with two similar methodologies in current use. Chapter 5 discusses how conceptual graphs could best be presented to the strategic management accountant. Chapter 6 tested ‘CARE’, that best presented conceptual graphs form, amongst experts who would be representative of the strategic management accountant. After this, the Conclusions are drawn followed by the Bibliography. Lastly there are the Appendices: ‘A/01’ to ‘A/06’ essentially relate to the matters discussed in Chapter 5; ‘B/01’ to ‘B/15’ relate to that of Chapter 6; ‘C/01’ to ‘C/05’ contain my prior published work relevant to this thesis.

Acknowledgements are due to the following persons: Chris Hinde, my supervisor for this research and an excellent coach in its conduct; John Heaton, for our countless conceptual graph technical discussions and co-operation with his ‘JEHGCP’ software; Pavel Kocura, who made me aware of the conceptual graphs theory; Roger Hopson of ‘Insurance Market Solutions’, for explaining to me the XLSIOR expert system that appears in Chapter 1; Paul Finlay for supplying information about ‘cognitive mapping’, discussed in Chapter 3; Malcolm King for advice regarding the ‘events accounting’ model of Chapter 4; Mark Sofroniou, for supplying the algorithms that drew the ‘arrow heads’ in the CARE software I designed; The participants for taking part in the user evaluations of Chapter 6; Clare Carrington-Windo, for her ideas regarding the contents of the questionnaire employed at the last user evaluation in Chapter 6; Conn Copas, for his help on the statistical analysis in Chapter 6; My parents for allowing me to take over their home through my final writing up of this thesis; Numerous others for their occasional help, and to whom I apologise for not referring to them by name.

My research was funded by the Science and Engineering Research Council, or ‘SERC’ for short.

I dedicate this thesis to my beloved niece, Nicola Polovina.
1 The Usefulness of Knowledge-Based Techniques to the Management Accountant

1.01 Introduction

This chapter examines what problems management accountants face when attempting to analyse the decision problems facing businesses, and evaluate how and in what way 'knowledge-based' techniques may be useful in improving the quality of that decision analysis. The chapter includes an overview of what knowledge-based techniques are about and two case studies to highlight the points raised.

1.02 The Need to Consider New Approaches

Within any given business, the accountant’s report of its performance is subject to precise constraints exerted from outside the business by both the accountancy professional bodies and by government. The former sets high professional standards and the latter passes legislation. The effect is that the accountant is forced to follow what are known collectively as generally accepted accounting principles. These measures are intended to restrain the accountant from being tempted to present misleading performance statements through unscrupulous creative accounting ploys. All this is unmistakable in key texts on the major accounting disciplines of financial accounting (Lee 1986), taxation (Beardon 1988), and auditing (Woolf 1990). Horngren and Foster (1990) define the other major accounting discipline of management accountancy as (page 2):

---

1 In the United Kingdom, these total five. They are (with their official acronyms in parentheses):

1) The Institute of Chartered Accountants in England and Wales (ICAEW)
2) The Institute of Chartered Accountants in Scotland (ICAS)
3) The Chartered Association of Certified Accountants (ACCA)
4) The Chartered Institute of Management Accountants (CIMA)
5) The Association of Accounting Technicians (AAT)
..... the identification, measurement, accumulation, analysis, preparation, interpretation, and communication of information that assists executives in fulfilling organisational objectives.

Johnson and Kaplan (1987) reveal the same generally accepted accountancy principles have extended into, *but to the detriment of*, management accountancy. According to Johnson and Kaplan such restrictions severely undermine the usefulness of management accounting and will lead to its eventual demise. Claret (1992c) reiterates this view and, in an earlier article (Claret 1990), illustrates the problem by the following example (page 24):

The consultants had been called in by the operations director. He was having difficulty controlling the cost of transport. All the accountants could give him were total figures broken down into such headings as petrol and oil purchases, drivers’ wages, maintenance salaries and materials, purchase of tools and hire of trucks. Even compared to budget, the figures meant little to him.

What complicated matters was the variety of vehicles: several sizes of tankers, delivery vans and lorries, larger trucks for bulk transfers from region to region, and all the salesmen’s and executives’ cars.

The director wanted to be able to monitor continuously the efficiency of use of the various elements of his fleet. He was also concerned about making decisions about what vehicles to use for different kinds of journey and about how many he actually needed.

He was particularly anxious about having recently bought new specialised tankers to replace several with high maintenance costs, only to be told that he had to keep the latter in use because they were still needed.

His needs were many and varied. The consultants’ study proposed a way of bringing together information about costs and usage. This meant collecting figures of mileages, tonnages, idle time, and other non-financial measures.

The figures already existed. They were mostly in informal records kept by transport managers, vehicle schedulers, loading bay foremen and maintenance departments. The new system would collect these and relate them regularly to new analyses of the costs. The operations director would at last have the means of knowing what the business was getting for the money he was responsible for spending.
The financial controller had his double entry system and methods of internal control which, for him, ensured accuracy. Reports about miles and hours were outside that controlled environment. The system was never installed.

Claret advises that if accounting systems are to be meaningful for management decision-making purposes such systems must be prepared to incorporate indicators derived from novel sources. A significant study by Berlant, Browning, and Foster (1990) confirms empirically this view when they described how a similar situation was improved upon at Hewlett-Packard through the deployment of new numerically-based analyses that were understandable by non-accountant key decision makers.

Kaplan and Atkinson (1989) express the above need exists because the focus of management accountancy is inherently on the diverse *internal* information needs of the organisation itself, rather than the legalistic requirements of *external* concerns. Management accountants should thereby strive continually to provide better timely, accurate and *meaningful* performance gauges as needed to business managers who can then rely upon these indicators to make suitably informed decisions.

Finlay (1989) explains how computer-based *decision support systems* aid decision making. Davis (1991) shows how current decision support systems can improve financial planning. Wilkinson (1989) specifies where they feature in accounting information systems. According to Wilkinson, decision support systems are most useful in the profound strategic and tactical decision-making performed by senior managers. Problems in these domains are characterised by ill-structured parameters which introduce much subjectivity in determining their solutions, especially in strategic domains (Dermer 1977, Simon 1977). Claret (1991b) and Tomkins (1991) urge management accountants to become significantly involved in *strategic analysis* because such professionals can bring a large element of objectivity into this increasingly important area.

A particularly successful decision support tool is the computer-based *spreadsheet* (Licklider 1989). One survey showed 92% of accountants used this decision support tool (Career Accountant 1990). Graff (1990) argues that spreadsheets can tackle more than the purely quantified aspects in *cost /
benefit analysis, which is a foremost area of concern to management. Although decision support tools such as spreadsheets essentially in their present form are versatile, Minch (1989) shows their scope is in fact limited because their conceptual roots currently lie in numeric and data-based relationships. As such they need extending to represent subjective problems adequately. Drury (1988) advises these qualitative elements need to be somehow translated into a form that enables their significance to be judged correctly.

Figure 1.01: Dynamic Funds Flow Model
(Adapted from Helfert (1982), page 19).

The above discussed choice of technique and its effect regarding this qualitative dimension can be demonstrated by an example. Figure 1.01 models the flow of funds in a typical manufacturing business. The diagram is

---

adapted from Helfert (1982) who provides the following supporting commentary (page 18-20):

The model ..... is another way of showing the dynamic interrelationships of funds flows, ..... in a manufacturing setting. This is achieved through representing the cycle of funds flows as a system of reservoirs and pipes through which which funds are "pumped", that is driven by the marketing effort, and regulated by management using the various valves at key decision points. In the normal course of business, funds flow clockwise through the system. Various funds sources - stockholders and creditors - are employed to purchase materials. Additional funds, largely cash, are committed to processing and completing the goods. Meanwhile, funds are expended on selling and administrative expenses, in effect driving the "pump", which causes a recapturing of funds through cash and credit sales. Eventually, cash is replenished and the cycle of operations, supported by investment and financing, can begin all over again. Note that in this illustration the reservoirs represent funds owned by the respective parties - trade creditors' funds, for example, are selectively released into the company's funds flow and replenished by repayment. Drawing on these funds is the classic use of "other people's money", while repayment removes these resources from the system. This is true as well for the funds of stockholders and long-term creditors. Note also that profitable conditions will cause a net inflow of funds from operations' while loss conditions will drain the system in part. A balance can be maintained by selective activation or deactivation of funds sources, to cover operating and investment needs. In the end, however, only successful operations over the long term will establish the viability of the resource deployment made by management.

Given combinations of pipe sizes, fluid levels in the reservoirs, and speeds of the flows would provide vivid and meaningful indicators of businesses performance to their managers. The example serves to demonstrate how models can be built that improve upon the traditional information of row and column computations with stark headings and numeric values. This is because it captures the underlying concepts and relationships. These, when added to information, provide the knowledge sought ultimately in problem solving situations.

To contrast, Claret (1990) gives a humorous example of an Eastern European concern's over-zealousness in supplying over-simplified solutions in the direction of quantifying the qualitative (page 25):
People tell of the light fittings manufacturer judged on weight of goods shipped. Naturally, it was easier to meet targets by making only the heaviest fittings - with consequent disastrous effects on the ceilings from which they were suspended!

This somewhat extreme example is indeed an illuminating illustration of the nature of the problem. Claret (1992b) also warns, as do Piper and Walley (1990), that even the most currently advanced numeric techniques in management accounting, such as activity based costing, are far from the panaceas that many of their advocates\(^3\) claim. I evaluated the overall situation thus (Polovina 1991a, page 25):

I remember my previous employment in a large multinational public company. I was responsible for the financial and management accounts of the property subsidiary which held all the multinational's buildings. Part of the reporting requirement to the top board directors involved producing an enormous 'manual' spreadsheet analysing five years' budgeted returns and expenditures on those buildings.

I stuck sheets of accountants pad together such that it covered the whole of my seemingly large desk. Many pencils and erasers later I carried the result to the appropriate director who in turn wanted answers to 'what if' questions. Back to my desk.

Fortunately, this exercise happened only once a year. The point is that, armed with a computerised spreadsheet like Lotus 123, this effort now could be performed so easily that it would have been possible to produce the whole item on a monthly basis.

However, computers cannot perform everything. The conceptual roots of current computer software lie in numeric and data-based relationships. The remaining vital qualitative aspects still have to be processed and explained manually. For instance, computer spreadsheets are invaluable as flexible decision support tools, but the accountant still has to relate manually the context under which the figures and charts are produced. Its form is usually verbal, or in an attached text-based report, or both.

The closest these areas come to being computerised is by being word processed. What do I mean by all this? Let us continue with my spreadsheet example above.

\(^3\) For an example see Cooper (1990), who is one such advocate of activity based costing.
Imagine I now present my director with the output of the computerised spreadsheet model. Already, I, not the computer, will have arranged the information into what I believe to be the most meaningful way. The results of calculations which I consider to be important will be highlighted, maybe by being in summary form on the first page. Less important ones and details will be relegated to subsequent pages or appendices. Insignificant items are not included at all. I design any charts similarly.

The director asks the following not untypical kind of question: “I know these figures will not be accurate because some buildings may be sold or demolished, others purchased or improved. Can you give me a revised report accounting for these possibilities?”.

To satisfy this, the report invariably now needs to contain, in words, the reasons why some of the figures no longer follow the general pattern in the spreadsheet.

Adding separate columns alone for these movements is insufficient: The director would focus on them and probably say “Tell me how you arrived at these values”. The reasons would reveal explicitly the appropriate responsible officials’ (and my) subjective judgements which determine the context from which the figures are derived.

This example highlights how problems require solutions not based on calculations alone. The judgements stem from knowledge gained from our life experiences and professional training. Knowledge is what the computer does not yet have.

Hence, management accountants have instead to seek a new basis for capturing the qualitative in a way that can be computer-manipulated.

1.03 Adopting the Knowledge-Based Approach

According to many writers such as Hayes-Roth, Waterman and Lenat (1983) and Patterson (1990), the kinds of problem above are part of a general category of qualitative problems which lend themselves to artificial intelligence techniques as evidenced in practice by the emergence of computer-based expert systems. This area is well documented and hence there is much overlapping literature on both non-financial and financial expert systems. Mahahon (1990) maintains the largest known bibliography on expert systems.

These knowledge-based approaches attempt to replicate logically the judgemental human reasoning process, or heuristic, in the computer. This method is supported by evidence from cognitive psychology (Smyth, Morris, Levy and Ellis 1987). Bratko (1986) distinguishes expert systems from knowledge-based systems as follows (page 314):

..... not every knowledge-based system can be considered an expert system. An expert system also has to be capable, in some way, of explaining its behaviour and its decisions to the user, as human experts do. Such an explanation feature is especially necessary in uncertain domains (such as medical diagnosis) in order to enhance the user's confidence in the system's advice, or to enable the user to detect a possible flaw in the system's reasoning. Therefore, expert systems have to have a friendly user-interaction capability that will make the system's reasoning transparent to the user.

The incompleteness theorem of Gödel (1931) had demonstrated already that in any sufficiently powerful and logically consistent formulation of mathematics there would be well-formed formulae which cannot be proved or disproved, thus making mathematics incomplete. This also meant that the consistency of infinite progressions, such as arithmetic, cannot be proven mathematically. Ackoff (1987) pointed out that the mathematically-oriented techniques of operational research have to make over-simplified assumptions about the complex parameters within which businesses operate. The seminal text on investment appraisal and financing decisions by Lumby (1991) acknowledges this reality permeates any numerically-based model of business performance. Lumby reveals that some such models, such as the internal rate of return, are in fact fundamentally flawed and may lead not just to inaccurate but potentially damaging decisions in particular cases. Thierauf (1990), in discussing financial and accounting expert systems, further points out human experts think heuristically because they would otherwise become ensnared in a combinatorial explosion as the number of possible outcomes of real decisions are impossibly vast. Furthermore, there is a limit on how much humans can reason even heuristically before they become mentally overloaded yet again. A computer is not constrained in this way. This was

⁴ These two sources additionally incorporate annotations and comments on the works cited.

In actual organizational practice, no one attempts to find an optimal solution for the whole problem. Instead, various particular decisions, or groups of decisions, within the whole complex are made by specialized members or units of the organisation. In making these particular decisions, the specialised units do not solve the whole problem, but find a ‘satisfactory’ solution for one or more sub-problems, where some of the effects of the solution on other parts of the system are incorporated in the definition of ‘satisfactory’.

For example, standard costs may be set as constraints for a manufacturing executive. If he finds that his operations are not meeting those constraints, he will search for ways of lowering his costs. Longer production runs may occur to him as a means of accomplishing this end. He can achieve longer production runs if the number of style variations in product is reduced, so he proposes product standardisation as a solution to his cost problem. Presumably, he will not implement the solution until he has tested it against constraints introduced by the sales department - objections that refusal to meet special requirements of customers will lose sales.

Anyone familiar with organisational life can multiply examples of this sort, where different problems will come to attention in different parts of the organisation, or where different solutions will be generated for a problem, depending on where it arises in the organisation. The important point to be noted here is that we do not have to postulate conflict in personal goals or motivation in order to explain such conflicts or discrepancies. They could, and would, equally well arise if each of the organisational decision making roles were being enacted by digital computers, where the usual sorts of personal limits on the acceptance of organizational roles would be entirely absent. The discrepancies arise out of the cognitive inability of the decision makers to deal with the entire problem as a set of of simultaneous relations, each to be treated symmetrically with the others.

Hence, a knowledge-based system is a computer model that takes advantage of the human experts’ factual knowledge and heuristic reasoning process unhindered by personal limits. An expert system extends this by exhibiting a model which the user can follow and interact with. Sowa (1984) illustrates the whole issue of computer heuristics by succinct reference to the relatively straightforward domain of chess playing (page 199):
Making a computer play a perfect game of chess requires an exhaustive search that would take longer than the age of the universe. But the more modest goal of beating 90% of the human players 90% of the time can be met with a $100 microprocessor that takes only two or three minutes a move. Heuristic programming is what makes the difference between an attainable level of competence and an impossible perfection.

Lastly, Thierauf (1990) adds the following about any expert system (page 11):

Once operational, it can be copied and distributed at little marginal cost to assist users, whether experts or inexperienced personnel.

This statement purports that expert systems can replace existing human experts. The reasoning behind this is that expertise is an expensive and thereby scarce resource to businesses. In addition, the human expertise may leave the organisation through a change in employment or death. Hence an expert system helps to spread and retain this valuable commodity within the enterprise.

1.04 The Nature of Existing Expert Systems

King (1983) compares knowledge-based systems to database systems. Knowledge-based systems attempt to organise and manipulate the raw material to be reasoned, namely the facts, into knowledge-bases. If ‘data’ is substituted for ‘knowledge’ in this sentence, this would be called a database. King’s editorial, however, summarises the differences. To paraphrase King, it may be stated that databases contain generally large numbers of records. Knowledge-based systems have fewer but ‘richer’ facts instead. Database query routines, such as Structured Query Language (SQL), need to know how the solutions are formulated. This means the user must choose what to select, delete, insert and update. Knowledge-based systems have more powerful query facilities than this. This is because they are intended to be built on a heuristic structure and therefore manipulate abstract queries in the same way as humans would.

Expert systems may be purpose built, similar to the act of writing particular applications using computer programming languages such as ‘BASIC’, ‘C’ or ‘Pascal’. Alternatively they may be bought ‘off the shelf’ pre-programmed
with 'generic' routines, similar to database software packages for example. The latter systems are known as expert system shells. The modelling of reasoning and facts is obtained usually from the human expert in a knowledge elicitation exercise. This is performed by a knowledge engineer, the equivalent of a systems analyst, who then attempts a knowledge representation of the problem domain.

Jackson (1990) categorises the way knowledge is represented under three main headings:

1. Production Systems. As noted above, an expert system must be user-understandable. Production rule-based systems usually of the 'if-then' form derive from the view these represent the most natural model of human problem solving (Newell and Simon 1972). As a simple example, a credit controller's expertise might include the following rules:

   If a customer does not pay within thirty days, then give them no further credit.

   If a customer is declared bankrupt, then give them no further credit.

   If a customer has poor credit rating, then give them no further credit.

   Any one (or more) of the above grounds would cause the customer's credit facilities to be suspended. Similarly, a customer who continues to enjoy credit is in none of these categories. However a false 'if' does not make a false 'then' which, in fact, is simply not asserted. For instance the fact:

   Western Company is not bankrupt.

   cannot imply that:

   Therefore credit to Western Company is granted.

   because that customer may fall under some other criteria for credit refusal. New inferences of knowledge involve a process of reasoning through
'chaining' together these 'if-then' rules in the same way a human expert is
deemed to do so. In particular, an asserted 'then' would cause other
'if-then' rules to operate provided their 'if' part matches the 'then' part in
the previous rule. This can be illustrated by another example rule:

If a customer is given no further credit,
then that customer’s future purchases must be paid for by
cash on sale.

Hence, by chaining this rule to the earlier non-paying rule it is possible to
state:

If Southern Company has not paid within thirty days,
then Southern Company must pay cash for new purchases.

Production systems typically consist of a knowledge base made up of
hundreds or thousands of rules with many possible consequent chains of
inferences depending upon the queries input by the user.

2. Structured Objects. This is the
generic term for associative
nets, frames, and object-oriented
succinctly describes the features
of structured objects in the
following description of frames
(page 23):

A frame is a data structure that
describes declarative and property
inheritance knowledge about an
item. Using frames is particularly
appropriate where structural
relationships organised as hierarchies
are to be represented - here a hierarchy
for company cars.

Figure 1.02: Figure 2 in Harrison (1989,
page 23): "Frames describe declarative
and property inheritance knowledge
about items, and their use is particularly
appropriate where structural
relationships organised as hierarchies
are to be represented - here a hierarchy
for company cars."
In a frame system, we have classes, sub-classes and instances. Instances represent 'real world' objects, whereas classes represent abstractions or categories. Frames may contain slots (or attributes) with associated values, for example, engine: 1800.

Any frame may inherit the existence of slots and their values from higher up the hierarchy. Values may be overridden at any point lower down. So the value 'small' of the 'space:' slot of instance 'sierra sapphire' would be inherited from the sub-class 'saloons'. The 'performance:' slot of 'cavalier hatchback sri' has overridden the inherited value of 'average' with 'good'. In the same way, all instances except 'sierra estate diesel' will inherit 'fuel: petrol' from the class 'car'.

3. Procedural Deduction. These systems attempt to define what route the reasoning should take in finding solutions. A good example is the logic programming language, prolog. To illustrate procedural deduction, consider an elementary family relations expert system which has the following prolog statements, or clauses:

```
parent( ann, mike). % ann is the parent of mike
parent( john, mike). % john is the parent of mike
parent( ann, simon). % ann is the parent of simon
parent( john, simon). % john is the parent of simon
parent( ann, brian). % ann is the parent of brian
parent( john, brian). % john is the parent of brian
parent( kath, mark). % kath is the parent of mark
parent( mike, mark). % mike is the parent of mark
parent( kath, nicola). % kath is the parent of nicola
parent( mike, nicola). % mike is the parent of nicola
parent( kath, david). % kath is the parent of david
parent( mike, david). % mike is the parent of david

predecessor( X, Z) :- % Rule 1: If X is the parent of Z
    parent( X, Z). % then X is the predecessor of Z

predecessor( X, Z) :- % Rule 2: If X is the parent of Y
    parent( X, Y), % and Y is the predecessor of Z
    predecessor( Y, Z). % then X is the predecessor of Z
```

Chapter 1
Prolog's syntax is based on a derivative of predicate logic known as 'Horn Clauses'. The predicates in the above example are 'parent' and 'predecessor'. Each supports two arguments which are the comma separated items inside the brackets after the predicate name. An argument in lower case typing, such as 'ann', is a fixed constant which means it cannot be altered. An argument in upper case, such as 'X', is a variable which means it can be fixed, or instantiated, by substituting a more precise term, such as a constant, in its place. Any text after a '%' symbol on the same line are merely comments which the computer simply would ignore.

The queries that can be made to this system, together with their answers, might include:

?- parent( ann, simon) % Is ann the parent of simon?  
yes

?- parent( ann, mark) % Is ann the parent of mark?  
no

?- parent( ann, B) % Who is ann the parent of?  
B = mike  
B = simon  
B = brian  
No more solutions

?- parent( A, nicola) % Who is the parent of nicola?  
A = mike  
A = kath  
No more solutions

?- parent( A, B) % Is A the parent of B?  
A = ann, B = mike  
A = john, B = mike  
.....  
A = kath, B = david  
A = mike, B = david  
No more solutions
?- predecessor( ann, simon)
   % Is ann the predecessor of simon?
   yes

?- predecessor( ann, mark)
   % Is ann the predecessor of mark?
   yes

?- predecessor( ann, B)
   % Who is ann the predecessor of?
   B = mike
   B = simon
   B = brian
   B = mark
   B = nicola
   B = david
   No more solutions

?- predecessor( A, nicola)
   %Who is the predecessor of nicola?
   A = kath
   A = mike
   A = ann
   A = john
   No more solutions

?- predecessor( A, B)  % Is A the predecessor of B?
A = ann, B = mike
A = john, B = mike
......
A = ann, B = mark
A = ann, B = nicola
......
A = john, B = nicola
A = john, b = david
No more solutions
In essence, prolog works by attempting to match queries with the given clauses. This occurs by unification, either by an exact match or the binding of a variable to a more particular term through instantiation. Prolog also relies heavily on recursion. This is where a clause calls itself and can be seen in the lower 'predecessor' example clause. Prolog starts at the top clause and then proceeds line by line towards the bottom clause until it finds a solution, if it exists. This solution may be then discounted and the query restarted to find a further solution if it exists, and so on until there are no more solutions. All this happens for the query itself and any sub-query clauses that are enacted. This should be evident from the order in which answers appear in the above example. Therefore the ordering of the clauses is important and determines the search strategy. In addition, prolog can backtrack. This does not occur in the simple example above, but is a feature of more complex prolog programs. Backtracking means if a variable happens during the course of a query to be instantiated to an incorrect term, then prolog will go back to the original point where the variable was initially instantiated and try another suitable term if possible. The effect of all this means prolog has what is called a depth first search strategy.

It should be noted that prolog is more multifarious than evident from the above relatively trivial example and discussion. A much fuller and precise account, including the terminology touched upon in the above brief discussion, appears in Clocksin and Mellish (1981) and Bratko (1986).

Many expert systems support more than one method. The reasoning for structured objects is likely to be augmented by some form of production rules or procedural language. Hybrids include 'Leonardo', a rule and frame expert system shell incorporating a limited procedural deduction programming language, and systems built upon 'Prolog++', combining prolog with object-orientation. Harrison (1989) and Pountain (1990) informatively review the former and latter respectively.

Expert systems may impart knowledge-based extensions to other kinds of computer software. For example, Parsaye, Chignall, Khoshafian, and Wong (1989) derive an intelligent database based upon the interaction of expert systems with databases. Klein and Methlie (1990) demonstrate the integrating of expert systems with numeric financial models and databases to yield a
knowledge-based decision support system. McCandless (1989) discusses how artificial intelligence and expert systems techniques are being incorporated into commercially available spreadsheets and databases. Colson, Helman, Gomes and Cihula (1989) describe the inclusion of an elementary 'if-then' capability within their spreadsheet software. Clarke and Finlay (1989) illustrate the need for Management Intelligence Systems (MINTS) within decision support systems to make them effective for strategic problem solving.

To clarify some of the issues discussed so far, the next section is a small case study about a financial expert system in current use. This leads naturally into the section immediately afterwards. That section continues the discussion by critically evaluating the existing expert system scene and what experiences are to be gained from this.

1.05 Case Study 1: 'XLSIOR' Reinsurance Expert System

The insurance market in the UK has evolved into a very complex form. This is due to the length of time it has been in existence. The 'London Market Insurance', namely Lloyd’s Underwriters and approximately 100 Insurance Companies, is the oldest market of its type in the world. An insurer acts as underwriter meaning it accepts the financial liability for the client's risks as defined in the insurance policy which is a contract agreed between them.

Reinsurance is basically the insurer's insurance. The insurers take this out so as to cover themselves should they be faced with major claims from their clients. This arrangement is, in fact, cyclical as reinsurance companies take out their own reinsurance. Therefore on the occurrence of, say, a major disaster such as the terrible hurricane in South East England in 1987, each insurer will claim increasingly huge sums off their reinsurers in turn.\(^5\)

\(^5\) Because the sums on risk involved tend to be so vast, the insurance company conducts its reinsurance, including all claims, through a broker. The broker places the reinsurance with a number of reinsurance companies. The number increases with the size of the policy. This is because the Department of Trade and Industry places a limit on the proportion that a single risk may constitute of the total risks of any reinsurer. Therefore the reinsurer is prevented from taking any 'gambles' that are likely to affect its own stability. In addition, this spread of risk
Table 1.01:  
**XLSIOR**  
Demonstration Reinsurance Programme  
(Courtesy of Insurance Market Solutions)

### Aviation

<table>
<thead>
<tr>
<th>Policy</th>
<th>Maximum Claim</th>
<th>Excess of Claim</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVN001</td>
<td>50 xs 50</td>
<td>2 @ 100%</td>
<td></td>
</tr>
<tr>
<td>AVN002</td>
<td>150 xs 100</td>
<td>1 @ 75%</td>
<td></td>
</tr>
<tr>
<td>AVN003</td>
<td>750 xs 250</td>
<td>1 @ 100%</td>
<td></td>
</tr>
<tr>
<td>AVN004</td>
<td>500 xs 1000</td>
<td>1 @ 50%, 1 @ 100%, Top &amp; Drop</td>
<td></td>
</tr>
</tbody>
</table>

### Rigs

<table>
<thead>
<tr>
<th>Policy</th>
<th>Maximum Claim</th>
<th>Excess of Claim</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB001</td>
<td>50 xs 50</td>
<td>1 @ 75%</td>
<td></td>
</tr>
<tr>
<td>RGB002</td>
<td>50 xs 50</td>
<td>2 @ 100%, Aggregate 100</td>
<td></td>
</tr>
<tr>
<td>RGB003</td>
<td>150 xs 100</td>
<td>1 @ 100%</td>
<td></td>
</tr>
<tr>
<td>RGB004</td>
<td>250 xs 250</td>
<td>Unlimited free</td>
<td></td>
</tr>
<tr>
<td>RGB005</td>
<td>500 xs 500</td>
<td>Unlimited free</td>
<td></td>
</tr>
</tbody>
</table>

### Generals

<table>
<thead>
<tr>
<th>Policy</th>
<th>Maximum Claim</th>
<th>Excess of Claim</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAT001</td>
<td>250 xs 50</td>
<td>1 @ 100%</td>
<td></td>
</tr>
<tr>
<td>MAT002</td>
<td>250 xs 300</td>
<td>1 @ 100%, Top &amp; Drop</td>
<td></td>
</tr>
<tr>
<td>MAT003</td>
<td>250 xs 550</td>
<td>1 @ 100%, Top &amp; Drop</td>
<td></td>
</tr>
</tbody>
</table>

Table 1.01 is a simplified representation of a reinsurance company’s own reinsurance portfolio. ‘XLSIOR’, marketed by Insurance Market Solutions, London, is the name of the financial expert system which evaluates the claim strategy. The figure shows three categories of insurance: Aviation, Oil Rigs, and General. Taking the first row in the Aviation table as an example, ‘AVN001’ is the policy number. ‘50 xs 50’ means a maximum claim of £100,000 can be submitted but only that in excess of £50,000 will be means there is no ‘pyramid’ structure of reinsurance firm size. Hence, a given insurer can find itself underwriting parts of reinsurance policies where the client is indirectly its own reinsurer.  

6 This category provides cover for claims which are so large as to exceed the specific category protections. It also provides cover when the specific categories have been exhausted through prior claims. In addition, it provides protection for categories which are not normally underwritten by the organisation and so do not justify specific reinsurances.

---

Chapter 1
Chapter 1

Table 1.02:  
Claim for £1.2m on AVN policies, AVN001 exhausted  
*(All figures in £000's)*

<table>
<thead>
<tr>
<th>Claim</th>
<th>Recoverable</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>100 of AVN004 (Drop) as AVN001</td>
</tr>
<tr>
<td>and 250</td>
<td>150</td>
<td>AVN002</td>
</tr>
<tr>
<td>and 1000</td>
<td>750</td>
<td>AVN003</td>
</tr>
<tr>
<td>and 1200</td>
<td>200</td>
<td>1200 of 1400 (1500 less 100) of AVN004 (Top)</td>
</tr>
<tr>
<td></td>
<td>1150</td>
<td>Total Recoverable</td>
</tr>
</tbody>
</table>

forthcoming from the policy. Therefore the maximum that may be recovered is £50,000. Hence the literal translation of, say, AVN002 '150 xs 100' is '£150,000 maximum recoverable with an excess of £100,000'. The maximum claimable is £250,000. If the maximum is not claimed, then the remainder may be claimed on subsequent occasions. Therefore, once a policy has been claimed on it becomes *exhausted*. i.e. no more claims can be made on this policy until its renewal date. However, reverting back to AVN001, '2 @ 100%' means the policy can be *reinstated* up to twice more on payment of a premium equal to 100% of the original premium. RGB004 and RGB005 are not subject to these number of claim restrictions.

RGB002 includes 'Aggregate 100'. This means the first £100,000 of the total claims made during the validity of this policy must be made elsewhere. In the simplest terms, it acts as a 'back up' policy for RGB001 from which the £100,000 in question would be claimed first.

AVN004 has one reinstatement at 50% then one at 100% of the original premium. More importantly, however, it has a 'Top and Drop' facility. This is applicable also to MAT002 and MAT003. This means that when all the other policies have expired and cannot be reinstated any more, these policies can drop down from their relative top positions to fill the gaps left by the expired policies. Note they are top and drop, not top or drop, meaning they may remain effective in their top position even though called to drop. This is brought out in Table 1.02 which illustrates an example claim strategy.
Figure 1.03: Graphical Representation of Demonstration Reinsurance Problem Domain (Courtesy of Insurance Market Solutions).

Figure 1.03 illustrates the relationship of the policies that make up the portfolio in Table 1.01. The arrows show the top and drop capabilities. In a realistic situation there are far more contracts - typically 100 or so. The cost of reinstatement premiums, different policy renewal dates, and the effect of different currencies all have to be considered as well.

XLSIOR determines those contracts from which recoveries can be made, calculates the recoverable amounts and the costs of reinstatement. The system matches the claims with the policies under the criteria of the date of loss and the classes the claims fall under. Additionally, account is taken of special conditions, such as war exclusion. The system resorts to the use of aggregate and top and drop where necessary. The user can get the system to modify its choices to cater for 'one off' arrangements.

The significance of XLSIOR can be appreciated in the cash flow savings made by a typical reinsurance company. It takes manually 6-8 weeks to prepare a recovery of £50m. XLSIOR does it overnight. Taking an interest rate of 15% this amounts to an interest saving of £900,000 per annum.
1.06 Experiences from Existing Expert Systems

Feigenbaum, McCorduck, and Nii (1988) and Leith (1990) mark the two extreme points in a continuum of views on the value of expert systems. Each delivers forcefully their cases in favour and disfavour respectively. The foreword by Tom Peters in Feigenbaum et al. states (page xiii):

I conclude that any senior manager in any business of almost any size who isn't at least learning about expert systems, and sticking a tentative toe or two into expert systems' waters, is simply out of step, dangerously so.7

Feigenbaum et al. is a catalogue of expert systems successes. However, its style emerges to be very 'missionary' in the way as described by Lilienfeld (1978, pages 1-2):

Technical and scientific advances in different fields have resulted in the establishment of new disciplines. These disciplines have been institutionalized and legitimated by the establishment of scholarly societies, most of which offer yearbooks and have founded journals. Most of these journals offer material of two sharply disparate kinds: (1) articles and studies of a highly technical, narrow, and specific nature, focused on concrete problems within the discipline; (2) large numbers of essays and articles of a "missionary" nature which are addressed not only to the technicians within the field, but to the general public at large. These articles attempt to explain to the layman the wider significance of the work being done within the discipline. Almost to a man, the practitioners within these fields appear to feel that their work is of more than merely "technical" value. They appear to be convinced that the discoveries and concepts are of major philosophical, societal, and even religious significance: They offer new images of humanity and society, of God and the creation of human beings, and of their interrelations. In addition to these missionary articles within the professional journals, a third type of publication has emerged — books and anthologies of readings addressed to the public at large, which seek to explain to the lay public what these fields are all about. Through these publications the missionary activities of scientific workers are no longer confined within limited professional media; there is an effort to reach the public at large.

7 The term 'expert systems' is my substitution for Peters' term 'AI' (Artificial Intelligence). This change makes for a more precise comment on the theme of Feigenbaum et al.'s work.
Chapter I

According to Lilienfeld these missionaries develop a mentality which causes them to lose sight of the actuality in what they are persuading others to accept. Eventually each such missionary is self-brainwashed into believing everything can be solved by a single technique. Lilienfeld severely criticised what he saw as the overwhelming missionary discipline of general systems theory (Bertalanffy 1968, Klir 1969, Laszlo 1972) and its interrelated derivatives in cybernetics (Wiener 1948, Ashby 1956), information and communications theory (Shannon and Weaver 1972), games theory (Neumann and Morgenstern 1967), operations research, and artificial intelligence. Lilienfeld backs up his arguments with notable examples. Dreyfus (1979) and Dreyfus and Dreyfus (1986) vigorously denounce any rash pronouncements made about the capability of artificial intelligence on the grounds that such claims are based upon naive reasoning. Leith, mentioned above, unequivocally condemns formalism within computer science and artificial intelligence in particular. Leith defines this term in these subject areas as (page 8):

At its simplest, formalism is the attempt, in computer science, to state that all problems are technical and / or mathematical and that computer science is not, in very large measure, limited by aspects of social complexity.

In Leith's view, there is nothing inherently wrong with formality because it provides a vehicle to understand the fundamentals of any discipline. However, he sees the 'formalists', the proponents of formalism, over-simplifying formality because they attempt to model everything on an over-solid foundation which cannot cope with the changing social and historical context of the real world. This mentality is significant in computer science and artificial intelligence. Here the formalists employ mathematical and logical 'foundationalist' tools which are based on the assumption that computer science and artificial intelligence is an exact science. In fact it cannot be so because the reality it tries to model is inexact. Like Lilienfeld, Leith backs up his arguments by well known examples. Leith's attitude towards artificial intelligence is exemplified by his statement as follows (page 138):

Artificial intelligence, throughout all its nooks and crannies, is a formalist enterprise. The formalism which it evidences stands opposed to the socio-philosophical perspective I present in a very direct manner: it suggests that social interaction is not required as a basis for "intelligent behaviour", since this intelligence can reside in a program as easily as in a man, the social animal. This of course, is the ultimate
formalism: everything we are, and can do, reduced to the 1's and 0's produced as our compiled code. In this view, there is no need for social history, for our intelligent systems are beyond social theory - they encapsulate intelligence simply through technique. Thus, said John McCarthy to Weizenbaum during a debate, “What do judges know that we cannot tell a computer? ... Nothing.”

Leith enumerates the failure of logic programming, in prolog, to accurately model the 1981 British Nationality Act as clear evidence that computers cannot do what judges can. Leith refers to the heated debate he had with Kowalski and Sergot (1987) on this subject as further proof of the inability of those involved within computer science to grip reality (See Leith 1990, page 114). As towards expert systems, Leith states (page 162):

The concept of expert systems is a highly formalist conception, which is slowly crumbling under the reality of application. We are finding that we cannot take expertise or knowledge and “axiomatise” it in the knowledge base of a program. And we are also finding that rules are not the powerful “foundationalist” tools which we thought they were. This formalist programme, like all formalist programmes, cannot cope with the reality of the social complexity of real expertise sited in the real world.

It should be noted that in the above statement, Leith refers to those rules which constitute the disciplines of mathematics and logic. Leith’s formalist is Lilienfeld’s missionary. In conclusion, Leith states (page 211):

There is no logic that we can follow which tells us how to do computer science. We can only argue against each other and be more critical and self-conscious of our own attempts at siting computers in the world, and try to site these with economy and elegance, also being aware of the side-effects we create and can mitigate. Yet when we look around at computer science we see little debate. Our view is obscured by concentration upon the machine and upon the program as the formalist tries hard to keep the social world at bay: the wheel is reinvented constantly in this vain attempt to solve problems of social complexity without admitting that social complexity is part and parcel of computer science.

But my final point, for now, in this debate is that after some twenty five years as a developing discipline, the growing up of computer science can only come through the dismissal of formalist grand plans and the acceptance of computer science as a social science just as much as a technical science. By attacking our problems with the breadth
of social-technical understanding will come strength (as well as debate and argument) but hopefully success too.

Gammack (1991) points out that the ‘future of expert systems is assured’ provided they adhere to tasks that involve a high degree of repetition and possess a clearly understood logic. Dreyfus and Dreyfus (1986) believe expert systems have scope only where there is a low level of expertise involved or where ‘quick’ rudimentary advice is preferred to the ministrations of human experts. Hence expert systems solve only a restricted range of easily definable problems characterised by a high computational element. This is evident in XLSIOR above. It is in these kinds of problem domains which Feigenbaum et al. describe as actual expert system successes but Leith sees as situations that, in reality, never warranted knowledge-based approaches. Wirth (1990) doubts the universal claims made about object-orientation. McBride, O'Leary, and Widmeyer (1989) and Xueli, Tse and Whinston (1989) describe how a knowledge-based approach solves problems in the cash flow planning problem domain, yet Singh (1989) demonstrates such problems can be modelled in as just a meaningful way employing numeric spreadsheets and flowcharts alone.

Edwards and Connell (1989), in work commissioned by the Institute of Chartered Accountants in England and Wales (ICAEW), surveyed the use of expert systems in accountancy. They substantiate the sceptics. Firstly their examples of the accounting expert systems in use arguably fall inside the category defined by Dreyfus and Dreyfus above. This view is underlined by the fact that these systems occur to a large extent within the precisely formalised domains of auditing and taxation. Secondly Edwards and Connell are mindful that (page 91):

In view of the optimism surrounding the use of such systems, particularly by the proprietors of such software, there is the need to guard against a situation in which the expert system becomes a ‘solution looking for a problem’, and domains are scoured accordingly until such a problem is found. The circumstances that produce a system in a technical or scientific domain are not necessarily likely to produce the same results in a commercial or business domain.

A survey by Wilson and Sangster (1992) revealed the use of expert systems even in the highly regulated domain of auditing essentially occurred amongst
only the very largest auditing practices. King and McAulay (1992) studied actual attempts to introduce management accounting expert system applications which failed. Their evaluation of why they failed substantiate both the sociological limitation and the unwarranted use of expert systems. Nonetheless there continues to be an interest in knowledge-based approaches. King and McAulay support this interest. According to another study by Wilson (1989, page 32):

"Expert systems have, to date, had a negligible effect upon the practice of accountancy. Though most UK accountants will be aware of expert systems it is doubtful if more than 1% have yet had any substantive contact with the technology."

It is now clear that expert systems will not be as pervasive as many enthusiasts first thought and that the reproduction of expert level behaviour is a more complex task than was first envisaged. It is unlikely that accounting will be substantially affected by expert systems in the foreseeable future. Having said that, it is also probable that, in the long term, expert systems will affect accountancy more substantially than other forms of computer applications. It is the only form of computer application which has the potential to reproduce the forms of behaviour which are central to expert level accounting.

Gillett (1991) continues to call for better knowledge engineering tools in accountancy. Wilson and Chua (1988), and Buckner and Shah (1989) maintain that expert systems are of valid interest to the management accountant. Humpert and Holley (1988) believe the criticisms relate to current events only and thereby overstate artificial intelligence’s incapabilities. They add (page 99):

"... that considerable improvements can be expected within the next five years. The critical reader should also consider the fact that most of the presented expert systems must be regarded as prototypes where a lot of emphasis was placed on artificial intelligence problems and less on an optimal and complete representation of the domain knowledge."

Bainbridge (1989) argues similarly in response to the above stated criticisms of the 1981 British Nationality Act. Although Humpert and Holley’s time horizon for the improvements are ambitious, they reflect the underlying

8 Humpert and Holley list and discuss expert systems in finance planning.
contention of not abandonment but improvement. This position may be reconciled with the sceptics provided such phrases as ‘optimal and complete’ in the above statement are read as ‘the best possible’. The critics would then accept the necessity of all future technological advancement provided the sociological limitations are recognised suitably in any such exercise. This is, in fact, occurring in general systems theory where the criticisms are allayed somewhat in the soft systems methodology theorised by Checkland (1981) and practiced by Checkland and Scholes (1990).

Hence a possible way forward exists for the knowledge-based approach. Keravnou and Washbrook (1989) continue to view these limitations in terms of overcoming the technical inflexibilities within the various expert system formalisms. Gammack (1991) resigns himself to accepting that a straightforward decision support system, aiding rather than replacing the human expert, instead of an expert system is the ‘usual compromise’. Klein and Methlie (1990) see the solution more positively as a matter of developing a decision support style of expert system. Gill (1988) argues convincingly for a user-centred approach in the research and development of expert systems. Technically-centred expert systems attempt to capture every element of knowledge that makes up expertise. Gill contends this will never be achieved because of the great difficulty in modelling the continually changing tacit and implicit parts of knowledge. The model of the user in this scenario is the non-expert. In contrast, user-centred expert systems capture only those easily modelled parts of knowledge and relies on the expert user to provide the remaining more difficult parts of knowledge when interacting with the expert system.

Executive information systems exemplify the decision support system equivalent of Gill’s approach. Wilkinson (1989) defines executive information systems as providing the information needed for the strategic planning undertaken by top level management. According to Coffey (1988) and Martin and Clarke (1990), by concentrating on an easy to understand user-centred approach they are breaking down the senior executive’s traditional resistance to computers. However a review by Holtham (1989) warns that an over-emphasis on this approach may result in systems which end up being too simplistic for the executive. Indeed he or she may well endure an over-simplistic system rather than appear as being too stupid through not being able to operate a more complex one.
Another danger in following a strictly user-centred mode is to under-emphasise such technical advancements that otherwise could be made. Executive information systems are not knowledge-based because they do not attempt to model the senior executive's strategic knowledge, yet George (1990) reveals how expert systems ideas can be extended conceivably into such categories. Liang (1990) views the user-centred approach as a necessary but temporary stage towards fully fledged expert systems in the multi-disciplinary area of management. Given the discussion throughout this section, it is very unlikely that such fully fledged systems are going to supplant human experts. Therefore, a new ideal for expert systems must be struck. Part of this ideal would be the deployment of a powerful enough form of knowledge representation which captures faithfully the more difficult elements within knowledge. The other part must be how to best support the human expert because the most difficult elements within knowledge will continue to remain outside the machine.

To clarify the issues discussed in this section, a second small case study forms the next section. This is a synopsis of the critical treatment by Lilienfeld (1978), Bloomfield (1986) and Leith (1990) on the systems dynamics work conducted by Professor Jay Forrester (Forrester 1961, 1969, 1971). After the next section, which comments on the issues raised by this case study, the concluding remarks section reviews the points raised throughout this whole chapter.

1.07 Case Study 2: Systems Dynamics

Jay Forrester embraced the principles of general systems theory. This meant he believed the technique could be applied to any kind of problem in any discipline. Computers were of particular interest to Forrester. This is because these machines could process the more formal models in systems theory, namely the feedback loop found in cybernetics, by far greater orders of magnitude than the human mind. The flow diagram in Figure 1.04 is an elementary feedback loop. This loop happens to depict a thermostat in a heating system. Taking the heating system as an example, the principle behind a feedback loop model is stability by the maintenance of equilibrium, here a particular temperature, by corrections of disturbances caused by, say, a temperature drop. Employing this shallow model Forrester turned his
attention to the complexities of the social sciences domain, starting with those in industry (Forrester 1961, page 1):

Our most challenging intellectual frontier of the next three decades probably lies in the dynamics of social organisations, ranging from growth of the small corporation to development of national economies. As organisations grow more complex, the need for skilled leadership becomes greater. Labour turmoil, bankruptcy, inflation, economic collapse, political unrest, revolution, and war testify that we are not yet expert enough in the design and management of social systems.

Hence he launched industrial dynamics, the first form of systems dynamics. Forrester saw such complex problems could be modelled and solved through linking together feedback loops to give a larger flow diagram structure. From this structure, a computer program could then be conveniently devised and run. This program would then be a simulation of reality over time. Forrester believed the model’s parameters should be based on the development of capitalism. Forrester preferred ‘experimental systems analysis’ in place of empirical data analysis. In effect this meant he kept altering the program until it gave him the answers he wanted to see. Forrester published his results without any evaluation of his technique in actual industry, justifying that it was in its early stages (Forrester 1961). Nevertheless there would occur (1961, page 46):

the general recognition of the advantage enjoyed by the pioneering managements who have been first to improve the understanding of the interrelationships between separate company functions and between the company and its markets, its industry, and the national economy. Competitive pressures will then lead other managements to seek the same change.
Next, Forrester attempted to apply the same modelling technique to the urban problems in cities. This became *urban dynamics*, the second form of systems dynamics. At this point Forrester stated (1969, page 113–114):

> In the social sciences failure to understand systems is often blamed on lack of data. The barrier to progress in social sciences is not lack of data. We have vastly more information than we can use in an orderly and organized way. The barrier is deficiency in the existing theories of structure. The conventional forms of data-gathering will seldom produce new insights into the details of system structure. Those insights come from an intimate knowledge of the actual systems. Furthermore, the structuring of a proper system theory must be done without regard to the boundaries of conventional intellectual disciplines. One must interrelate within a single system the economic, the psychological, and the physical. When this is properly done, the resulting structure provides nooks and corners to receive fragments of our fabulous store of knowledge, experience and observation.

From the above statement, Forrester recognised correctly the deficiencies in existing data and information processing techniques. Forrester’s own solution, however, turns out to be at least as questionable. His computer model simulated life in an archetypal city over an astonishing 250 year period. Such a long time period for cities in his model was needed, in Forrester’s view, to take account of the ‘delay’ variable before the effect could be seen. This delay factor featured in the reasons for his pre-emptive publication of industrial dynamics above. Forrester's model basically blamed the disturbances as attributable to the poor whom he referred to as the ‘underemployed’. To achieve equilibrium his model entailed that the underemployed should be discouraged from further entering the cities and thereby re-encourage the managerial-professional class back into the cities attracted by new enterprises to replace declining industry. Where else the underemployed should go, or how they could improve their lot, was conveniently overlooked in the narrow model drawn up by Forrester. The output from Forrester's model of the development, maturity and stagnation of cities is illustrated in Figure 1.05.

In addition to the above-raised reservations, Forrester was also criticised for not surveying previous studies on urban problems. He retorted by stating (In Leith 1990, page 125):
Actually the book comes from a different body of knowledge, from the insights of those who know the urban scene firsthand, from my own reading in the public and business press, and from the literature on the dynamics of social systems for which references are given.

![Figure 1.05: Life cycle of an urban area](image)

In fact *Urban Dynamics* (Forrester 1969) contained only six references of which five were to himself. Forrester rejected the intuitive human solution for providing low-cost houses, state-funded education and welfare in favour of the harsh solutions of his model. According to Forrester such judgements were gained from misleading social experiences or, as he preferred to call them, *negative feedback loops*. Forrester, in fact, insisted that if city authorities were unwilling to implement his 'objectively' derived recommendations they should be made to do so by higher authorities. Yet Forrester's model was also based on human perception of social reality, except that he accepted another viewpoint. Namely this was the reactionary view to the urban ghettos and race riots happening in US cities in the 1960's. That view was basically 'clear them out and make our city clean again'. The parameters illustrated in Figure 1.05 reflect this attitude. Forrester by this time was convinced that his naive computer model knew more than the human mind.
The ultimate challenge for Forrester was to model the world. Employing his previous techniques in the above two domains he presented *world dynamics*, the last in the trilogy of systems dynamics, at the 1970 Swiss Conference of the 'Club of Rome'. The Club of Rome were a group of international dignitaries. According to Forrester the world faced chaos (Forrester 1971, page 2):

Population, capital investment, pollution, food consumption, and standard of living have grown exponentially throughout recorded history. Man has come to expect growth, to see it as the natural condition of human behaviour, and to equate growth with "progress". We speak of the annual percentage growth in gross national product (GNP) and in population. Quantities that grow by a fixed percentage per year are exhibiting "exponential" growth. But exponential growth cannot continue indefinitely.

Forrester's exponentially growing quantities are based on biology, a subject which is fundamental to general systems theory. Many biological concepts had been transferred by the systems scientists into other disciplines. Here, the transfer occurred from doubling of biological cells when they reproduce. Like the feedback loop this 'multiplier' was easy to understand and thereby model, yet grossly underestimated the true complexity of the real world. Figure 1.06 shows the structure of 'World 2', one of Forrester's world models. From the
beginning Forrester's model had used the multiplier, and the whole world was no exception (Forrester 1971, page 5):

Within one lifetime, dormant forces within the world system can exert themselves and take control. Falling food supply, rising pollution, and decreasing space per person are on the verge of combining to generate pressures great enough to reduce birth rate and increase death rate. When ultimate limits are approached, negative forces in the system gather strength until they stop the growth processes that had been previously in control. In one brief moment of time the world that the apparent law of exponential growth fails as the complete description of nature. Other fundamental laws of nature and the social system must and will rise far enough to suppress the power of growth.

Presuming the above Forrester again selected his own parameters. Again items that should have been of interest to him such as scientific advancement, better use of the world's resources, and clearing up the pollution were ignored. Furthermore he chose to ignore himself. In both industrial and urban dynamics, his models were based on capitalism of which growth is the pillar. However, in world dynamics growth was rotten. Forrester, in fact, was consistent in that he formalised not neutral phenomena but differing human viewpoints that so characterise society. Industrial dynamics modelled capitalism, urban dynamics modelled reactionary opinions to social strife, and world dynamics modelled the Club of Rome's intellectualising of the popular hippy-like 'opt out' attitude in 1960's society.

1.08 A Positive Comment about Systems Dynamics

Although termed as systems dynamics, Forrester had demonstrated the power of a graphical flow structured knowledge-based system. This can be seen in Figure 1.04 and Figure 1.06. The earlier Figure 1.01 also represented knowledge in this kind of way. Such a representation had given Forrester the ammunition to make his far-reaching claims. As it turned out Forrester's representation was a gross caricature because it relied on over-simplified ideas, namely the feedback loop and the multiplier, to model the world over and above human opinion. Yet, setting this aside, system dynamics did manage to capture and model qualitative social and political opinions, albeit unwittingly.
Chapter I

The advantages of the graphical flow structured model is one of the tenets of this thesis. This subject is elaborated in the next chapter. The following concluding remarks section brings out the other salient points arising from this case study.

1.09 Concluding Remarks

As stated towards the beginning of this chapter, management accountants are free from the superfluous shackles of external legalistic constraint. What matters now to these professionals instead is to furnish the best possible analysis demanded by internal senior managers. The computer spreadsheet has provided a significant benefit in this direction. However, given that such managers are increasingly concerned with solving ill-structured strategic problem situations, management accountants are going to have to tear themselves away from purely numeric and data-based analysis and become involved with knowledge-based paradigms. Figure 1.01 illustrated models which can actually capture knowledge and therefore computer-manipulate it.

Hence, the appeal of artificial intelligence which is intended to do precisely this task. Yet practical experience from expert systems has shown this lofty appeal has caused a general tendency of those involved with this subject to fall into the missionary trap which presumes that everything can be modelled. The system dynamics case study demonstrates the fallacy of this mentality.

It was pointed out that mathematical decision-making techniques such as operational research made over-simplified assumptions about the complex parameters within which businesses operate. The same can be said of knowledge-based systems. Systems dynamics offered a powerful but simple to understand mapping of reality yet it was based on parameters which appealed to particular social and political viewpoints. Its outcome proves the key point made in this chapter. That is there remains a case for knowledge modelling research, but with a new ideal. Part of this ideal would be to discover knowledge representations which capture faithfully the more difficult elements within knowledge. The remainder would be to find the best way this representation can best support the human expert. This is because the machine's model of knowledge is always subject to an outer context of continually changing tacit and implicit knowledge that will remain inside the
human mind. Quite apart from its over-simplistic basis, system dynamics was an unfeasible model because it rejected rather than accepted human review.

The subject of 'artificial intelligence' has become an emotive one. To some it serves only to fuel two untruths. The first is to put the fear into humans that they will be supplanted by machines. The second is the belief by its researchers, such as Forrester, that they can become missionaries that will do this. Nonetheless, provided it adopts the essence of these criticisms, artificial intelligence can continue to progress. Management accountants, in their role as strategic analysts, should be looking towards a refreshed knowledge modelling discipline for a meaningful knowledge-based decision support tool so they can adequately help senior decision-makers.
2 The Usefulness of Conceptual Graphs to the Strategic Management Accountant

2.01 Introduction

Given the critique in Chapter 1 about the value of knowledge-based systems to the management accountant in strategic analysis, this chapter seeks the direction in which the most appropriate solution might be found. The usefulness of graphical representations of knowledge to these strategic management accountants is discussed, and in particular the advanced knowledge-based technique of conceptual graphs. The discussions are supported with examples, including two case studies in an advanced area of management accounting.

2.02 The Worth of Problem Structuring Aids

It may seem initially that production rule systems hold the answer because, as indicated in the previous chapter, they appear to be easy for the user to follow. However, there are both technical and usability limitations on this approach. Alty (1987) argues that rule-based systems are an insufficient paradigm of human mental models of problem solving. Some knowledge could not be captured and, worse, users had also to cope with 'irrelevant knowledge' about how the computer dealt with their expert knowledge. Harrison (1989), in reviewing the Leonardo expert system shell, realises the human mind is patently superior to the set of any given 'if-then' rules that a Leonardo model would embody. The usability limitation also affects procedural deduction systems, discussed in Chapter 1. This is because they too impose an additional computational overload, such as depth-first searching in prolog, on users who are forced to know how the system conducts its search strategy.

King (1989), in an empirical study, discovered that the limitations occurred because rule-based systems cannot be guaranteed to be easily understandable by the human end-user and, moreover, do not enforce any coherent structure. King's study involved four problem domain experts, named 'A' to 'D', developing their own expert systems using a rule-based expert system shell. Experts A and B were engineers, and experts C and D were accountants. King concluded (page 133):
All four found difficulty in mastering the concepts inherent in the package. Once experts B and C had surmounted the initial hurdle they found the package straightforward but grew irritated at its limitations. The other two experts struggled with the package throughout the project, one in terms of the concepts and the other in terms of using the package efficiently. This suggests that there is still considerable development needed before these tools are suitable for use by experts without a flair for computing. However, the more fundamental problem faced by experts A and D was the difficulty of structuring their knowledge to fit the package representation. All four cases demonstrated the importance of logically structuring the object knowledge first in a modular and hierarchical form and considering the relevant meta-knowledge before starting to engineer the knowledge base to suit the shell.

With respect of the most successful outcome of the four, who modelled personal taxation problems, King’s following comment substantiates the expert end-user support argument for expert systems (page 133):

Case C demonstrated a different mode of operation where the system supported the expert and helped to guarantee his consistency and accuracy. The most advanced prototype in this case was close to an operational system.

It became clear to King that end-users had to know how to structure their own knowledge before attempting to enter it into the above expert systems. The hierarchical aspect was initially discussed under structured object representations in the last chapter and will be elaborated further, as will the modular aspect, in this thesis. King’s study also suggests an expert system should be designed that further supports human experts by helping them to perform this structuring.

Practical expert systems are thereby indifferent to knowledge-based systems. This is because even rule-based versions evidently turn out to lack the distinguishing user-friendly interaction capability that, as identified in the last chapter, makes the system’s reasoning transparent to the user. The next

---

1 Note the overcoming of the shell’s usability hurdle only to reveal a fundamental technical limitation is a vivid irony which well highlights the discussion so far.

2 Meaning knowledge about the knowledge that relates directly to the given problem.
section commences a route towards overcoming all the obstacles raised so far, through *structured diagram* techniques.

2.03 The Need for Structured Diagram Approaches

Pictures, sketches, figures or diagrams are included in documents, like this thesis, to bring vividly across points or arguments which are more difficult to do so by text means. While this is accepted as a fact, it has been impossible to attest categorically why this is so psychologically. Nevertheless, studies such as Fish and Scrivener (1990) keep demonstrating that human understanding is improved by visual representations.

The use of structured diagrams are no exception. These diagrams, through their visuality, establish a structure on whatever situation they model. This is evidenced by DeMarco (1979) in *structured systems analysis*, Chen (1981) in *entity-relationship diagrams*, Ashworth and Goodland (1990) in *structured systems analysis and design method* (SSADM), and Checkland and Scholes (1990) in *soft systems methodology*. Not surprisingly therefore, structured diagrams have been augmented with knowledge-based dimensions. For instance Lazimy (1991) introduces the *extended* entity-relationship model. Murray and McDaid (1993) in their review of visual modelling techniques show how structured diagrams better facilitate the domain experts' building and reviewing of their knowledge-bases directly. As regards typical applications, King and McAulay (1991) found that 'and/or graphing' allowed the easier end-user modelling of a statement in a standard costing problem that could not so easily be achieved by a text format. In financial planning, Mays, Apté, Griesmer and Kastner (1987) described 'FAME', and Mui and McCarthy (1987) described 'FSA' by how they were better developed using structured diagrams.

Furthermore, structured diagrams are a tool that is familiar to accountants, who employ them for precisely the purposes as just discussed. Woolf (1990) demonstrates how flowcharts are a required tool for auditors to gain insights into how an organisation works, where its strengths and weaknesses lie, and what improvements might be made. Sizer (1989) similarly accredits their use in a decision support role for management accountancy. An advanced knowledge-based derivative that retained this familiarity and respect stands to benefit the strategic management accountant greatly.
Chapter 2

2.04 The Limits and Scope of Structured Diagram Approaches

Nonetheless there remain certain impediments in the way of using structured diagrams. In the structured diagram techniques discussed so far, the resulting diagrams still need to be re-coded manually into text form for the computer. Attempts to develop software that allows the end-user to input flowcharts directly into the computer have met with limited success. For example, Girvan (1990) reviews 'Matrix Layout' which is such a package. Girvan acknowledges the visual advantage that Matrix Layout imparts but criticises it for being too cumbersome for other than elementary models. Murray and McDaid (1993) state the trouble with graphical representations is that they can easily become too large for both a) the computer screen's size, and b) the human user to follow. They remark that the graphical objects in any representation need to be broken down, as King recommended earlier, into smaller manageable modules whilst retaining the same meaning.

In any event, it does not follow that any structured diagram technique is always appropriate for every problem. This is evidenced by the augmentation of such diagrams with knowledge-based dimensions to handle qualitative problems. A study by Reagan-Cirincione, Schuman, Richardson and Dorf (1991) evaluated the worth of using Forrester's systems dynamics approach, discussed in the last chapter, in a practical strategic problem solving situation. As also discussed in that chapter, they discovered that the approach's too-simplistic nature did not ultimately concur with the knowledge of their domain experts. The problem modellers had employed the systems dynamics-based simulation software STELLA (Richmond, Peterson and Vescuso, 1987) that facilitates the direct manipulation of the systems dynamics diagrams on computer.

To provide an illustration of the kind of flowcharting technique to which accountants are familiar, the next section focuses on a small case study. The case study employs the flowcharting technique in one pertinent part of management accounting that cannot be modelled purely on a numerical or data-based spreadsheet. The subsequent section discusses a particular problem with the technique. The domain referred to is "Relevant Values for Decision Making".
2.05 Case Study 3: An Accountant's Flowchart for Relevant Values

Claret (1991a) agrees that an entity's value does not necessarily equate to its cost. This is because, as Horngren and Foster (1990) advise, that all decisions involve predictions. Nothing can be done to alter the past. Any historical costs or benefits are thereby irrelevant, although they may be the best available basis for future values. Bierman, Dyckman and Hilton (1990) add the costs or benefits determined by accounting information systems cannot replace judgement in choosing the best course of action. Papers by Coulthurst and Piper (1986) recognise the relevant values domain should rely less on techniques and be based more on concepts. They call for a clearly structured conceptual framework. According to Coulthurst and Piper if a cost or benefit is 'avoidable', meaning that it can be altered by the alternatives in a decision, then that value is relevant to the decision at hand. Otherwise it is irrelevant, as it is 'unavoidable' anyway, and can thereby be excluded from the alternatives.

Figure 2.01 illustrates a typical flowcharts depicting relevant values.
flowchart that would be drawn by an accountant for a relevant values problem. Diagram (a) in Figure 2.01 goes through each decision alternative and selects which ones are relevant. The Is the alternative avoidable? diamond, which is shown as shadowed, is elaborated in diagram (b). Similarly (c) is a breakdown of the shadowed diamond Does it satisfy policy? in (b). Diagram (c) depicts an instance where the management accountant's organisation has an environmentally friendly policy for the sale or other disposal of its items. In essence any environmentally hazardous item will not be released by the organisation to a destination that does not also support the environment. In so doing the organisation has further restricted its options, as the above policy means that in some cases otherwise relevant alternatives become irrelevant.

2.06 The Particular Problem With the Accountant's Flowchart

Note that the diagrams in Figure 2.01 are drawn without reference to good systems analysis practice, as shown by standard program construction texts such as Stone and Cooke (1987), even though conventional flowchart symbols are used. In Figure 2.01 there are no clearly defined if, repeat or while loops for example. As Stone and Cooke would argue, such building blocks are considered not only essential to properly encoding the problem on a computer but to facilitate human understanding as well. Prolific evidence of this bad modelling can be seen in both of the above-referred seminal texts by Woolf and by Sizer. It is not the accountant's fault but that of the flowcharting technique itself as it, like if-then rules, does not enforce a sufficiently coherent structure. A better visual model should automatically direct these users into producing and maintaining models based on sound practice.

2.07 Comments in Respect of All the Above

The last problem above exemplifies the reservations of Vitalari (1984) about structured analysis. According to Vitalari such analysis offers little help in problem definition, diagnosis, formulation and planning. These are the essence of any useful knowledge-based decision support system from what has been expressed already. As can be seen from the above case study, visuality on its own does not guarantee adequate problem structuring. Seeking to give the accountant a better quality of structured diagraming method is nevertheless worth pursuing, not least because of the technique's
above documented popularity amongst this professional group. Another reason is that most practical studies such as Hammer and Janes (1990) and Kuehn and Fleck (1990) continue to emphasise the above discussed advantages, rather than the disadvantages, of structured diagraming techniques in senior-level management decision making.

2.08 The Candidate Solution: Conceptual Graphs

Therefore, given all the matters discussed up to this point, a candidate solution that is based on an advanced knowledge-based structured diagram paradigm is proposed for the strategic management accountant. At first sight that candidate, conceptual graphs, appears to deal with the problems whilst promoting the advantages. As well as explicating the nature of conceptual graphs, the validity of this selection is extensively examined throughout the remainder of this thesis. To begin with, the nature of conceptual graphs are introduced.

2.09 Introducing Conceptual Graphs

Conceptual graphs were devised by Sowa (1984) from philosophical, psychological, linguistic, and artificial intelligence foundations in a principled way. Knowledge in conceptual graph form is highly organised by modelling specialised facts which can be subjected to generalised reasoning. Work carried out by a) Garner and Tsui (1985) and Garner (1987) in auditing; b) Billo, Henderson, and Rucker (1990) in engineering analysis; and c) a comparative study carried out by Slagle, Gardiner, and Han (1990) in business liquidity analysis all demonstrated encouraging results for conceptual graphs as a knowledge specification tool. Dick (1992) found that conceptual graphs were an insightful expert end-user modelling tool for structuring legal knowledge.

In a comprehensive comparative study carried out by Webster (1988), conceptual graphs were acknowledged as the richest means of representing and manipulating expert knowledge in a computer model. However, Webster also warned that conceptual graphs seemingly extracted a too severe penalty in terms of the complexity encountered in using them. This can be seen in Figure 2.02 which relates the technologies examined by Webster. The

---

3 The comparatives were AGNESS (Slagle, Wick, and Poliac 1986) and KEE.
'processable expressiveness' in Figure 2.02 equates to conceptual graphs' richness, and the 'conceptual complexity' to their intricacy. Webster's findings would explain the relatively limited effect of conceptual graphs in any meaningful application areas, although there continues to be an active interest in this way of modelling knowledge⁴.

The usability criticism arises because Sowa's work leaves many loose ends: It was basically a stimulus for further research and development rather than a meticulous programmer's manual. Furthermore, subsequent conceptual graphs work has tended to reinforce Webster's complaint by introducing extra devices which endow yet more complexity⁵. This is because such efforts have leaned towards the technical issues alone, whereas the last chapter has shown

Figure 2.02: Processable expressiveness versus conceptual complexity (taken from Webster (1988), page 20).

the potency of knowledge-based systems depends ultimately on the problem domain experts' ability to review any computer paradigm of their knowledge in accordance with their own mental models.

---

⁴ For instance, there are the annual workshops on conceptual graphs. In addition, there is the conceptual graph electronic forum on "cg@cs.umn.edu". New subscribers are included by sending a request to "tjan@cs.umn.edu".

⁵ For example, see Gardiner, Tjan and Slagle (1989) and Tjan, Gardiner and Slagle (1990).
Therefore the following sections attempt to redress the apparent complexity by clarifying the salient aspects of the accepted conceptual graphs theory. It commences by examining the ‘concepts’ and ‘relations’ from which conceptual graphs are built. Next the ‘hierarchical’ nature of conceptual graphs is discussed. Together with the following discourse on the ‘projection’ and ‘joining’ of graphs, all this should reveal how the specialised and generalised knowledge in conceptual form co-relate to one another. The discussion then ends with subject of ‘inference’, which depicts how the above properties are employed to derive further knowledge. These sections are illustrated by examples similar to those found in Sowa.

2.10 Concepts and Relations

Conceptual graphs are based upon the following general form:

\[
\text{CONCEPT}_1 \rightarrow \text{RELATION} \rightarrow \text{CONCEPT}_2
\]

This may be read as: "A RELATION of a CONCEPT_1 is a CONCEPT_2". The direction of the arrows determines the direction of the reading. If the arrows were pointing the other way, then the reading would be the same except that CONCEPT_1 and CONCEPT_2 would exchange places.

As an alternative to the above 'display' form, the graphs may be written in the following 'linear' form which permits input and output using a text-based computer terminal:

\[
[\text{CONCEPT}_1] \rightarrow (\text{RELATION}) \rightarrow [\text{CONCEPT}_2].
\]

The full stop '.' signals the end of a particular graph.

---

6 Based on Polovina (1992a).
7 The remaining areas of ‘Schemata and Prototypes’ (Sowa 1984, pages 127–137) and ‘Actors in Dataflow Graphs’ (Sowa 1984, pages 187–195) have been excluded on the grounds that they are peripheral to the thrust of this thesis. Actors are introduced in Polovina and Delugach (1993).
Consider the following example:

\[ \text{[Mammal]} \rightarrow \text{(part)} \rightarrow \text{[Trunk]}. \]

This reads as "A part of a mammal is a trunk". This may create readings which may sound long-winded or ungrammatical, but is a useful mnemonic aid in constructing and interpreting any graph. It is easier to state "A mammal with a trunk".

All concepts have \textit{referents}, which refers to a particular individual of that concept\textsuperscript{8}. For example consider the concept:

\[ \text{[Mammal: Clyde]} \]

This reads as "The mammal known as Clyde", and also happens to show that a conceptual graph can consist of only one concept. The referent is a \textit{conformity} to the \textit{type label} in a concept. This example shows that Clyde conforms to the type label 'Mammal'.

A concept that appears without an individual referent has a \textit{generic} referent. Such ‘generic concepts’ should be denoted as \[ \text{[Type Label]: *]} \]. Writing \[ \text{[Type Label]} \] is merely a convenient shorthand. Generic concepts may take up an individual referent. For instance Clyde would have a particular trunk. A unique number referent can suffice to make a concept distinct. Thereby the generic concept \[ \text{[Trunk]} \] might become \[ \text{[Trunk: #1234]} \] with respect to \[ \text{[Mammal: Clyde]} \]. This would yield:

\[ \text{[Mammal: Clyde]} \rightarrow \text{(part)} \rightarrow \text{[Trunk: #1234]}. \]

\textsuperscript{8} Sowa discusses other kinds of referents such as ‘measures’ and ‘sets’. Measures are discussed in Sowa 1984, pages 89–90, whilst sets are to be found in Sowa 1984, pages 115–119. The use of sets, measures and other items as referents have tended to raise many complex issues resulting in controversy amongst the conceptual graph community (For example see Kocura, 1992). These other kinds of referents are not discussed, as their currently contentious nature precludes their part in an accepted theory of conceptual graphs.
Larger graphs may be constructed such as the following (Sowa 1984, page 78):

```
(Spoon) -
  (instrument) <- [Eat] -
    (object) -> [Walnut] -> (part) -> [Shell: *x]
    (agent) -> [Monkey],
    (material) -> [Shell: *x].
```

"A monkey eating a walnut with a spoon made out of the walnut's shell".9

This would be impossible to reproduce in linear form. Therefore additional devices are employed to overcome this:

The hyphen ‘-’ shows the relations of a concept are listed on a subsequent line. As the arrows both to and from a relation must flow in the same direction in a conceptual graph, no ambiguity is introduced. The comma ‘,’ terminates that part of a graph which relates to the last hyphen. Any part of the graph following the comma relates directly back to the hyphen before the last hyphen and so on, except there is no need to add a ‘,’ to pair up with the very first hyphen. Hence that part of the above graph from ‘(object)’ to ‘[Monkey]’ relates to ‘[Eat]’ whilst that from ‘(instrument)’ to ‘(material) -> [Shell: *x]’ relates to ‘[Spoon]’. Indenting the above graph clarifies the layout but has no computational effect. The ‘*<whatever>’ signifies a coreferent link between concepts. This shows that each such concept has the same referent. Here, if one of the ‘[Shell: *x]’ concepts

---

9 The verbose translation may be stated by reference to the conceptual catalogue given in Sowa 1984, Appendix B, pages 405-424. The section on relations (B.2, pages 415-419) explains the terms 'agent', 'object', 'instrument', 'part', and 'material'. Sowa does not distinguish along the lines of, say, nouns as type labels and verbs as relations. It might be argued that constructors of graphs would be aided if they were to think in terms of these distinctions. If such guidelines do help particular graph constructors build and clarify their graphs to themselves and others, then they should be used accordingly.
were to acquire an individual referent, then so would the other one. The '.' terminates the whole graph as before.

Linear graphs may be written differently but remain equivalent. For instance the above graph may have been written perhaps more clearly as:

\[
\text{[Eat]} \rightarrow \text{[Monkey]}
\]
\[
\text{(agent)} \rightarrow \text{[Walnut: *y]}
\]
\[
\text{(object)} \rightarrow \text{[Walnut: *y]}
\]
\[
\text{(instrument)} \rightarrow \text{[Spoon]}
\]
\[
\text{(material)} \rightarrow \text{[Shell]}
\]
\[
\text{(part)} \leftarrow \text{[Walnut: *y]},
\]

2.11 The Type Hierarchy

In conceptual graphs, type labels fall into what is known as a type hierarchy. Thereby:

Mammal < Animal.

This means Mammal is a more specialised type of the type 'Animal'. i.e. Mammal is a subtype of Animal. Alternatively, this can be stated as Animal is a supertype of Mammal. Similarly, the remainder of the hierarchy may be:

Animal < Entity.

Trunk < Entity.

Entity < T.

The type denoted as 'T' means the universal supertype. It has no supertypes and is therefore the most general type. Arrangement of each type into a hierarchy is a powerful feature. Knowledge is built up from degrees of commonality as each subtype inherits the sum of all its supertypes. Furthermore, a subtype can have more than one immediate supertype. For example, consider the concept ['Pet Cat']. This concept has a type label which may be defined as the subtype of both 'Pet' and 'Cat'. Therefore the graphs that make up Pet and Cat would both be inherited by 'Pet Cat'.

A neat representation of the type hierarchy is a form known as the type lattice. The type lattice gives a clear multi-inheritance structure. Each pair of type labels has an immediate minimal common supertype and maximum common subtype in a perfect type lattice. Practical knowledge bases developed
in conceptual graph form, however, are far more likely to fall somewhere in between a lattice and a 'tree' hierarchy into degrees of *acyclic graph* structures. A diagram illustrating the three hierarchy structures can be seen in Figure 2.03.

A referent of a type label in a concept is also a referent of that type label's supertypes. Continuing with the illustration, as Clyde conforms to Mammal then Clyde would also conform to:

- Animal: Clyde
- Entity: Clyde
- T: Clyde

These supertypes say less about Clyde than Mammal does. This relationship can be represented in *type definitions*. A subtype of Mammal, namely 'Elephant', may be introduced by the following type definition:

```plaintext
type Elephant(x) is
[Mammal: *x] -> (part) -> [Trunk].
```

"An elephant is a mammal with a trunk".

This also means:

Elephant < Mammal.

The variable "x" specifies that any individual referent would be coreferent for both Elephant and Mammal. The "<Type_Label>(coreferent)" heading part in a type definition is the *specialising concept* and the graph below that header is the *defining graph*. A specialising concept can replace the defining graph and vice versa. As Clyde is such a mammal it is now possible to assert:

[Elephant: Clyde] -> (part) -> [Trunk: #1234].
This graph cannot be contracted further because of the individual referent in the 'Trunk' concept\(^{10}\).

In addition, the form "\(<\text{Type}\_\text{Label}\> :: \text{<individual>}" lists explicitly the most specialised type label conformity for an individual referent. In the example this would be:

Elephant :: Clyde.

It is similarly possible to have definitions for relations. For example:

\[
\text{relation Grandparent}(x, y) \text{ is } \\
[\text{Person: } *x] - \\
(\text{parent}) \rightarrow [\text{Person}] - \\
(\text{parent}) \rightarrow [\text{Person: } *y].
\]

"A grandparent is a parent of a parent."

The type hierarchy might include:

Person < Animal.
Parent < Person.
Grandparent < Parent.

2.12 Projection

Projection is the 'pattern matching' operation for conceptual graphs. Every graph becomes more specialised when it:

a) increases the number of concepts, types and relations within it,

b) acquires a non-generic referent, or

---

\(^{10}\) Unless given an individual definition (Sowa 1984, pages 119-121). These are essentially type definitions but with an individual referent such as:

\[
\text{individual Elephant(Clyde) is } \\
[\text{Mammal: Clyde}] \rightarrow (\text{part}) \rightarrow [\text{Trunk: #1234}].
\]
c) substitutes subtypes for particular types.

Therefore 'inside' such a graph there exists the 'original' general graph. A general graph is likely to have many specialised variations. Thus the general graph projects into any of these specialised graphs. A projection common to more than one graph is said to be a common generalisation.

Projection plays an important role in the inference for new knowledge with conceptual graphs. If a graph happens to project into another, then a particular pattern may have been identified. From this discovery a new graph may be asserted. Inference is discussed in detail later. Projection also features in the combining of graphs to form larger graphs. This subject is discussed first and will serve to illustrate projection.

2.13 Combining Graphs

The joining of graphs into bigger structures provides a coherent structure to the knowledge base. Furthermore, it facilitates inference because more projections can be made into larger graphs.

Maximal join attempts to define the optimal method by which graphs are joined and occurs when graphs are joined on the largest, or maximally extended, projection that is common to them. The act of joining the graphs has the effect of specialising this projection accordingly. That projection now becomes a common specialisation from which the remainder of the original graphs can then be 'attached' to give a new graph. This is illustrated in Figure 2.04 which is based around "A man loves a woman", "Simon is located in Loughborough", and "A woman has blonde hair".

Having stated all this, such joining of graphs may lead to invalid results. This obstacle arises because Sowa treats the generic referent as theoretically equivalent to the existential quantifier, '∃', in predicate logic. ∃ means a conditional "there exists an item such that ..... ". For instance, a graph which

---

11 The relational catalogue given in Sowa 1984, Appendix B, pages 415–419 explains the terms 'experiencer', 'location', 'characteristic', and 'object'.

12 Conceptual graphs are, in fact, an existential notation. This is demonstrated in Sowa (1984, pages 86-87) during a direct mapping between conceptual graphs and first order predicate logic.
has [Person], [Person: *], or [Person: *x] in it means 'There exists a person (or some person) such that the other concepts and relations which make up the attached graph are valid'. However this comparison is violated when graphs are combined because any item will be suitable as a referent in a generic concept provided it conforms merely to the type label in the concept. This pays no regard to the conditional statement given by any concepts and relations attached to it. [Person] is treated as any person when it should mean an unknown person.

This can be illustrated in Figure 2.04 where it is quite possible that a male other than Simon loves Clare, Simon loves a blonde who is not Clare, Simon loves Clare and another blonde, Clare loves Simon and another male, Clare is not a blonde, or any suitable combination of these. Due to such lack of knowledge, joining should only occur when the referents are known to match in the graphs to be joined or when all of one graph can project directly onto another.

2.14 Inference

Before proceeding with the examination of the accepted form of conceptual graph inferencing it should be noted that Fargues, Landau, Dugourd and Catach (1986) reject Sowa’s accepted method of inferencing new knowledge in conceptual graphs. They instead prefer their 'easier to follow' inferencing mechanism than Sowa’s, by placing graphs in 'if-then' rules. Dick’s representation also replaces the inferencing part of conceptual graphs by 'if-then' rules (Dick 1992). Apart from thus causing the already discussed
limitations about 'if-then' reasoning to be imposed on conceptual graphs in both of these cases, it was decided to remain with the original accepted form of inference for further reasons that should become evident as this thesis progresses.

Inference in conceptual graphs theory is based upon the existential graphs logic of Charles Sanders Peirce. This 'Peirce logic' is developed by Sowa in the same principled way to provide a comprehensive inference capability in conceptual graphs. Peirce logic, cited by its founder as 'the logic of the future', is seen by Sowa as an enhancement of the traditional propositional and predicate logic of Peano, Russell, and Whitehead.

Consider the following example:

\[ \text{if Graph 1 then Graph 2.} \]

This may be read as "If Graph 1 can project into any graphs in the knowledge base, then Graph 2 can be asserted".

Logically, 'if-then' can be rewritten as:

\[ \text{not (Graph 1 and not Graph 2).} \]

This can be written graphically in Peirce logic as:

```
Knowledge-Base Graphs
    Graph 1
    Graph 2
```

This visual form illustrates the contextual domination of graphs over other graphs. A graph is dominated by another graph if the dominated graph is ringed by what is known as a negative context, whereas the dominating graph is outside of that ring. Here, the Knowledge-Base Graphs dominate Graph 1

13 Pronounced as 'purse'.
which dominates Graph 2. Any graph which projects into a graph which dominates it may be 'rubbed out' or deiterated. To assert Graph 2, Graph 1 must project into the Knowledge-Base Graphs. Should this occur, Graph 1 can be deiterated leaving two rings around Graph 2. The term 'not(not Graph 2)' equates to the term 'Graph 2' so the empty outside ring and the inside ring cancel out, or double negate. This frees Graph 2 out of the context and thereby means it has been asserted as a new graph. A true antecedent in an 'if-then' rule means its consequent is always true. This is the general inference rule of *modus ponens* and has been demonstrated here using Peirce logic.

The linear equivalent of the above is:

\[-[\text{Graph 1} \quad \neg\{\text{Graph 2}\}].\]

The symbol '−' means 'not'. A '−[ . . . ]' forms a negative context ring. For reference purposes, negative contexts may be numbered by an integer series according to how deeply they are nested. Graphs which are not inside any negative contexts are deemed to be at level 0 (zero). Graphs are deemed to be at level 1 if within a negative context, 2 if within a nested negative context, 3 if within a nested nested negative context, 4 if within a nested nested nested negative context, and so on. The general form is \(n\) if within an \(n-1\) nested negative context. The following diagram should help to illustrate this:

![Diagram](https://via.placeholder.com/150)

As indicated above '−[{<A Graph}>]' would mean that a graph is not true. The term 'not true' in this sense equates to that graph being *false*. Therefore it is also possible by the appropriate use of negative contexts and nested negative contexts to build a knowledge base consisting of both true graphs, false graphs, and various inferences of those graphs. For the sake of clarity, this discourse will substitute '( . . . )' in place of '−[{ . . . }] ' to denote negative contexts written in the linear form.

Consider the next example:

Graph 1 and Graph 2.
This is merely a case of adding Graph 1 and Graph 2 to the knowledge base because they both are true. Say, however, the example was:

if (Graph 1 and Graph 2) then Graph 3.

In Peirce logic form this would be:

(Graph 1 Graph 2 (Graph 3)).

Assuming that Graph 1 and Graph 2 existed in the knowledge base, then they can be deiterated and Graph 3 double negated thereby asserting it as a new graph at level 0. Now say that the knowledge base happened to include the graph (Graph 3) instead. This states that Graph 3 is false. Regarding the above rule, (Graph 3) can be deiterated from it leaving:

(Graph 1 Graph 2).

This shows that because Graph 3 is false then both Graph 1 and Graph 2 are false. It is still possible for either Graph 1 or Graph 2 to be in the knowledge base but not both. If they were or some derivative that would state they were, this would show there is an inconsistency in that knowledge base. Return to the first ‘if Graph 1 then Graph 2’ example:

(Graph 1 (Graph 2)).

Should (Graph 2) be in the knowledge base, then (Graph 1) would be asserted. This demonstrates another general inference rule of modus tollens, or that if the consequent of an ‘if-then’ rule is false then so is its antecedent. The illustration also shows that if the antecedent is false, then the consequent cannot be determined from it. If (Graph 1) was in the knowledge base to begin with, there is no possible way to assert either Graph 2 or (Graph 2) from (Graph 1) alone.

It should be noted that if a graph is false then so will be any of its specialisations. If it is false that “Clyde is an elephant”, then “Clyde is an elephant who lives in a wildlife park” must also be false.

As a final example, consider:
Chapter 2

Graph 1 or Graph 2.

Logically, 'or' can be rewritten as:

\[ \neg (\neg (\text{Graph 1}) \land \neg (\text{Graph 2})). \]

This maps to the Peirce logic form:

\[ ((\text{Graph 1}) \land \text{Graph 2}). \]

By the above discussed Peirce logic operations, if either Graph 1 or Graph 2 was false then Graph 2 or Graph 1 would be true respectively.

Figure 2.05 is an example using actual conceptual graphs. Graphs which include negative contexts are known as compound graphs. This example shows that before any projecting graph can be deiterated, it must first be specialised and any referents passed onto all other coreferent concepts. In the display form the coreferent link may alternatively be shown as a dotted line, which can be seen in Figure 2.05. This figure uses the same 'elephant' example as before and reveals the relationship between type definitions and inference. For the inference to become a type definition there would need to be another compound graph with ‘[Mammal:*x] \rightarrow (\text{part}) \rightarrow [\text{Trunk}]’ at level 2 and ‘[Elephant:*x]’ at level 1. Then the inference would work both ways as either the defining graph or the specialising type label could be asserted as appropriate. This applies similarly to relation definitions. The additional compound graph required to turn the example inference into a type definition is illustrated by the following linear representation:
The semi-colon, ‘;’ indicates the end of a particular graph other than the final graph. The ‘.’ is used only to terminate the whole compound graph.

In summary, Peirce logic shows contexts of knowledge elements visually dominating others and inference is performed by attempting to reduce those contexts.

The discussion now moves to considering a case study in an advanced aspect of management accountancy. This area was introduced earlier: “Relevant Values for Decision Making” and was stated as one could not be modelled purely on a numerical or data-based spreadsheet. The case study is the example of “Scrupulous Chemicals Company”.

2.15 Case Study 4: Scrupulous Chemicals Co. (Relevant Values)

Scrupulous Chemicals Company uses the chemical “GGS” to make one of their products. The manager in charge of the department involved provides the following comments:

“As regards the GGS in stock, the accounts show we bought it for £20,000. It would cost £30,000 to replace, and if we don’t use it, it’s environmentally hazardous, but we can sell it for £16,000 to Mr. Wisenheimer”.

On which figure should the Scrupulous Chemicals management base its production decisions?

This query is passed to Scrupulous Chemicals' management accountant who is expected to evaluate such problems. The management accountant thinks this problem can benefit from using conceptual graphs, so decides to structure it accordingly. To begin with, the graphs representing the query, Figure 2.06, and the statement, Figure 2.07, are drawn. The type lattice, Figure 2.08, is

---

15 Based on Polovina (1991b).
Figure 2.06: Query graph:

GGS : #7568

relevant measure

Figure 2.07: Statement graph:

Figure 2.08: Initial type lattice:

GGS < Tradable Entity. GGS is an entity which can be traded.

Sole Trader < Legal Person. In law, a person includes incorporated bodies such as limited liability companies. The type 'legal person' extends person to include this characteristic of organisations.

£ < Measure.

The statement graph, Figure 2.07, may be read as "The GGS #7568 has:

a) the disposal buyer who is the sole trader Mr. Wisenheimer,
b) the characteristic of being an environmental hazard,
c) the historic measure £20,000,
d) the replacement measure £30,000,
e) and the disposal measure £16,000".

The last figure is enclosed in parenthesis in the graph because it is income rather than expenditure. This accords with accounting convention.

The query graph, Figure 2.06, may be read as "The relevant measure of the GGS #7568 is some £". It is this latter concept for which a referent must be found.

These graphs may not appear to match precisely the statement or query sentences. The reason is the structure necessary in forming the graphs requires the tacit and implicit knowledge in the sentences to be made explicit. Mr. Wisenheimer is a referent to a type label. This could have been left as defined for those types not a direct subtype of the universal supertype for the purposes of this example.
'legal person', but the management accountant was aware Mr. Wisenheimer was a sole trader. Therefore, the conformity is to the most specialised type label because it contains *more* knowledge. £20,000 is related by historic measure to the GGS because the management accountant *knew* accounts figures are always historic.

Of course, relationships such as between 'accounts' and 'historic' could be modelled as well. Once a definition is made explicit, the user can refer to its singularly defining subtype or sub-relational label in an implicit but *not* ambiguous way. The management accountant is free to design the graphs in whatever way, as user, is thought appropriate. Subsequently, although not shown in this example, such items may in fact be included. Should such actions happen to cause an inconsistency, the user may wish to re-think his or her own mental conceptual model. The overall point is that the conceptual graphs model has demanded the user gives an *immediate* structure to the implicity and ambiguity of natural language interpretations. The user's model of the problem is not prescribed, rather the user clarifies his or her understanding of the problem. A rule-based representation could neglect these inconsistencies by, say, allowing the user to write some 'patching up' rules. Conceptual graphs make this much harder to do. This ensures the user is not lead astray.

To continue, an attempt is made to see if the query graph, Figure 2.06, can be projected onto the statement graph, Figure 2.07, and thereby assert its truth and discover the relevant value. This cannot be achieved. As the graphs are in their most specialised form, type or relational expansion may be possible. This will reveal underlying explicit knowledge from where there is perhaps common ground for projection to occur. As it happens, the relevant measure relation in the query graph, and the three measure relations and the disposal buyer relation in the statement graph can be expanded to give the redrawn query graph, Figure 2.09, and statement graph, Figure 2.10. The extended type lattice, Figure 2.11, is drawn accordingly.

The expanded query graph, Figure 2.09, asks if GGS is the subject of a relevant value which has a measure in units of £. Should the query succeed, this value will have a unique referent. The £ measures in this example happen to be unique, but in any realistic scenario this would be unlikely. The individual relevant value referent allows these cases to be distinguished.
Examining the expanded statement graph, Figure 2.10, which happens to be now in four parts, also reveals detailed underlying knowledge. The graph could be left as one large structure, but this makes it difficult to read. The graph remains valid in this form because they are all coreferent on the concept \( \text{GGS} : \#7568 \). Projections can span over more than one graph accordingly. The ‘% <words>’ are simply comments.

For each sub-relation of measure in the original graph, Figure 2.07, the expanded graph, Figure 2.10, shows that value is determined by an event called a ‘GO-poss’, an acronym for change of possession\(^\text{16}\). The value has a £ measure. The earlier ‘allows to be distinguished’ discussion of value’s subtype, relevant value, is also valid for value.

For the replacement and disposal cases, the GO-poss is part of a replacement and disposal transaction event respectively. The GO-poss concepts are part of transactions because a transaction

\(^\text{16}\) Although incidental to the discussion here, the rationale behind the use of ‘GO_<term>’ acronyms are detailed by Kocura (1992).

---

Figure 2.09:
Query graph (expanded):

![Query graph](image1)

Figure 2.10:
Statement graph (expanded):

![Statement graph](image2)

Figure 2.11:
Extended type lattice

Types not linked to supertypes and / or subtypes are direct subtypes of \( \top \) and supertypes of \( \bot \) respectively. Explanations of type labels are provided, as appropriate, in the course of the main text.

![Extended type lattice](image3)
would include some other GO_poss, which would reflect what is exchanged for the GGS. The subject of this GO_poss is usually banknotes, a cheque, a creditor or debtor obligation. The historic case transaction does not appear from the historic measure expansion, but if it did the supertype concept [Transaction] would be attached with the particular GO_poss via a part relation from the former to the latter. This is because historic measure occurs from any type of transaction.

The subject of the GO_poss is the GGS, which is the value subject too. The graph has these relationships in its shape because the management accountant is aware that change of possession realises value. A tradable entity does not have a value in its own right. It acquires value solely because some person or organisation is willing to give up something else for it. Once again, conceptual graphs makes the implicit knowledge in a statement such as "GGS has a value of £X" to be made explicit.

Lastly, the GO_poss happens at a past or future point in time. The disposal buyer sub-relation shows the disposal instance's GO_poss subject destination as Mr. Nasty. This establishes the destination is that of the GGS. Simply linking the destination direct to the GGS would give graphs that would show GGS has multiple destinations. Clearly, this is not so. The destination relates to the appropriate GO_poss. Therefore, the relation is correctly drawn.

Another attempt at projection is made, but is unsuccessful as the [Value] concepts in the new statement graph is specialised to ["Relevant Value"] in the new query graph. As mentioned earlier, a relevant cost or benefit is avoidable by the alternatives in a decision. Including the implicit knowledge this becomes, "a relevant cost or benefit is a value determined by an 'avoidable GO_poss', an avoidable change of possession". Hence, the management accountant draws the appropriate type definition for 'Relevant Value', shown by Figure 2.12.

The expanded type definition replaces relevant value in the query graph, Figure 2.09, to give the more expanded query graph, Figure 2.13. Another attempt at projection is made. Still, it is unsuccessful as the [GO_poss]
concepts in the expanded statement graph, Figure 2.10, are specialised to [ 'Avoidable GO_poss' ] in the more expanded query graph of Figure 2.13.

Type expansion of avoidable GO_poss is not possible because a GO_poss becomes avoidable when one, or more, prevailing contexts are successfully evaluated. This point is elaborated next.

The compound graphs, Figure 2.14, depict two such avoidable cases. These graphs represent Scrupulous Chemical's environmentally friendly policy. The top graph shows the company can avoid the GO_poss in the future of a tradable entity provided it is false that the tradable entity has a characteristic which is an environmental hazard. The bottom graph shows the company can nevertheless avoid the GO_poss of a tradable entity which is an environmental hazard provided it is true that the GO_poss subject destination is to an environmental supporter.

Taking the top graph, the simple graph in the odd outermost context can be projected into the statement graph, Figure 2.10. This means insertion and joining can take place such that deiteration can occur in the odd outermost context. A simpler way of stating this is the aforesaid simple graph can project into the statement graph and hence be deiterated. However, it is not possible to deiterate the false environmental hazard. In fact, the opposite characteristic of GGS happens to be true. Therefore, no progress is made here.
As for the bottom graph, the outermost graph can be deiterated completely in respect of the concept \[ \text{GO}_\text{poss} : \#ii \] which relates to the replacement transaction subgraph in the statement graph, Figure 2.10. The own company type label of Scrupulous Chemicals is a subtype of environmental supporter thereby revealing, amongst other features which distinguish it from being an environmental supporting company, the company's care for the environment. The co-referent avoidable \text{GO}_\text{poss} concept is double negated and hence asserted as:

\[ \text{[ 'Avoidable GO}_\text{poss'} : \#ii ] \]

Considering the bottom graph again, the outermost graph could additionally be deiterated for the disposal measure, but the management accountant does not know about Mr. Wisenheimer's environmental consciousness. After further investigation, the management accountant confirms that Mr. Wisenheimer is no environmental supporter. Hence, the following graph is drawn:

( [ 'Environmental Supporter' : Mr. Wisenheimer ] ).

i.e. \¬ [ [ 'Environmental Supporter' : Mr. Wisenheimer ] ].

This also means Mr. Wisenheimer cannot conform to Environmental Supporter. Therefore, the deiteration does not take place.

Incidentally, if Mr. Wisenheimer did happen to be an environmental supporter the following graph would have been asserted instead:

[ 'Environmental Supporting Sole Trader' : Mr. Wisenheimer ].

The query graph, Figure 2.12, is then re-projected into the statement graph, Figure 2.10. The projection succeeds for the replacement subgraph because the appropriate \text{GO}_\text{poss} concept conforms to avoidable \text{GO}_\text{poss}. This yields the result, Figure 2.15. This can be contracted to give the graph, Figure 2.16.
This may be contracted down further to the original query graph size. However, the management accountant may wish to make the relevant measure relation more suitably distinct to take account of the relevant value referent, #2. The relational definition ‘relevant measure / #2’ may be drawn. Using the new definition, the graph, Figure 2.17, may be produced.

The management accountant now can present the results of this evaluation to the company’s management. Any queries on the result itself can be explained by going through the graphs, and any further ‘what if’ queries can be modelled by the management accountant without immediate recourse to rewriting the whole problem. Such restructuring that may occur will serve not to patch holes in, but further refine the management accountant’s present conceptual framework.

2.16 Comments Arising from the Scrupulous Chemicals Case Study

The above small example may appear to be trivial, but illustrates how even the subtleties in a qualitatively based strategic management accounting problem could be captured in a model that can be directly computer manipulated. A further criticism of taking a conceptual graphs approach might be that they have transformed what appears to be a simple problem modelling exercise into something horrendously complex. This accords with the above-stated reservations of Webster (1988). The complexity arises due to the need to explicate presumptions and background knowledge. Although the above problem had to make all the appropriate implicit knowledge explicit, such an exercise should essentially need to occur only once. Given an increasing set of graphs, larger problems are expected to require relatively fewer graphs as a graph library is built up by the user. Certain graphs would
need re-defining, but this would be unsurprising as part of the earlier arguments stressing constant user review.

2.17 Concluding Remarks

Rule-based approaches cannot capture sufficient knowledge. They also suffer, as do procedural deduction formalisms, from burdening the human expert end-user with extraneous computational knowledge. After duly considering their reservations, which include not necessarily guaranteeing to enforce a sufficiently coherent structure, structured objects hold more promise in the form of advanced knowledge-based structured diagrams that still facilitate end-user review.

In the above regard, conceptual graphs were introduced. This advanced knowledge-based technique can be presented to the strategic management accountant as emerging from an already respected methodology, namely structured diagrams. Thus conceptual graphs may pave the way for these management accountants to have on hand a familiar decision support tool that can adequately model their problem domain. This success of conceptual graphs would assure the combination of technical advancement with the user-centred aspects that knowledge-based systems need to succeed. This initial evidence needs to be established further, and forms the remaining part of this thesis.
3 Conceptual Graphs and Cognitive Mapping

3.01 Introduction

Chapter 2 provided initial evidence that conceptual graphs are a suitable knowledge-based decision support tool for strategic management accountants. This chapter starts to pursue a more substantial confirmation of conceptual graphs' technical capability in the strategic analysis problem domain, by comparing them with the cognitive maps of Eden (1991). Eden's mapping technique, which is a leading knowledge-based structured diagram technique for strategic planning, is based on the advanced 'personal constructs' methodology begun by Kelly (1955). The cognitive mapping technique both a) employs a highly structured approach, and b) is designed as a practical human expert end-user support tool. Given all the discussions so far, it thereby offers a valuable comparison with conceptual graphs. Should conceptual graphs sufficiently enrich Eden's cognitive maps then the choice of conceptual graphs will be further strengthened accordingly. As its basis, the examination employs the realistic office location problem that Ackerman, Cropper, and Eden (1991) choose in highlighting the benefits of cognitive mapping. An analysis of the same problem is performed using conceptual graphs.

3.02 The Example Problem

The example itself is as follows (Ackerman et al. 1991, page 41):

"We need to decide on our accommodation arrangements for the York and Humberside region. We could centralise our service at Leeds or open local offices in various parts of the region. The level of service we might be able to provide could well be improved by local representation but we guess that administration costs would be higher and, in this case, it seems likely that running costs will be the most important factor in our decision. The office purchase costs in Hull and Sheffield might however be lower than in Leeds. Additionally we need to ensure uniformity in the treatment of clients in the region and this might be impaired by too much decentralization. However we are not sure how great this risk is in this case; experience of local offices in Plymouth, Taunton and Bath
in the South West\textsuperscript{1} may have something to teach us. Moreover current management initiatives point us in the direction of greater delegation of authority."

### 3.03 The Cognitive Map for the Example Problem

Ackerman et al. cognitively map the above problem and produce the diagram in Figure 3.01 as a result. This figure illustrates two essential elements underlying this cognitive mapping interpretation. Namely these elements are 'concepts' and 'links'. Each concept is represented as one emergent 'pole', which describes one side of the problem, and a 'contrasting pole' which is meant to focus the concept by a meaningful contrast to the first pole. Poles may lead to other poles by means of directed links. All this is clarified further by examining the map as it appears in COPE, which depicts these cognitive elements.

Figure 3.01: Cognitive map for offices location problem
(Source: Ackerman et al. 1991, page 47)

\textsuperscript{1} The passage actually states 'South East' but this is a typographical error as confirmed by the references to 'South West' later in the article.
maps in computer software\textsuperscript{2}. To begin with, the map's concepts are represented by the following table in COPE:

1. open local offices...centralise services at Leeds
2. local representation...[not] local representation
3. increased running costs...[not] increased running costs
4. higher administration costs...[not] higher administration costs
5. improve level of service...[not] improve level of service
6. too much decentralisation...[not] too much decentralisation
7. risk of impaired treatment of clients...ensure uniformity of treatment
8. lack of understanding about risk...[not] lack of understanding about risk
9. use experience of s w local offices...[not] use experience of s w local offices
10. lower purchase costs of local offices...higher cost in Leeds
11. greater delegation of authority...[not] greater delegation of authority
12. follow current management initiatives...[not] follow current management initiatives

Note COPE adds a sequential number to signify each concept entered by the user, and automatically adds the prefix '[not]' to create a 'default' contrasting pole for any concept where a contrasting pole was not entered. The links are entered by taking the two appropriate concept numbers and placing a '+' symbol between them. The concept before the symbol leads to the concept placed after it. For example, '10+1' shows that 'lower purchase costs of local offices' leads to 'open local offices'. This also stipulates the contrasting pole 'higher cost in Leeds' leads to the contrasting pole 'centralise services at Leeds'. The cognitive mapping methodology also happens to stress that it is important the emergent pole should always represent what the user can best identify with. However, this is likely to create problems when it comes to making links as this constraint means poles may lead to poles of the other kind. The '-' symbol thus replaces the '+' symbol to combat this problem. This is illustrated by the following links table in COPE:

\begin{tabular}{cccc}
1 & $>+6$ & $+3$ & $+2$ \\
2 & $>+5$ & & \\
3 & $>+4$ & & \\
6 & $>+7$ & & \\
8 & $>1$ & & \\
9 & $>-8$ & & \\
10 & $>+1$ & & \\
11 & $>+1$ & & \\
12 & $>+11$ & & \\
\end{tabular}

The above table reads from left to right. The number before the '>' symbol signifies the concept that leads directly to all the concepts signified by number after the '>' symbol. Each '+' shows a pole to pole, and contrasting pole to

\textsuperscript{2}COPE, an acronym for Cognitive Policy Evaluation, runs on an IBM PC or compatible and is available from the Strategic Decision Support Research Unit, University of Strathclyde, UK.
contrasting pole, link. The '-' is a pole to contrasting pole, and contrasting pole to pole, link. The link '9-8' thereby means 'use experience of s w local offices' leads to '[not] lack of understanding about risk', and '[not] use experience of s w local offices' to 'lack of understanding about risk'. Another link, '.', is a 'connotative' link and is employed when the user knows there is an insufficiently definable yet somehow valid connection between concepts. This link is applied to the relationship between concepts '1' and '8' as overcoming 'lack of understanding about risk' may lead to either operating centralised services or opening local offices.

As an option the user can direct COPE to draw the map in 'graphical' form, as shown by the COPE screen below for the concepts linked to concept '1':

Concepts which are preceded by '**' indicate there are other concepts attached to them. These other concepts can be brought up on screen in this modular form as required, and there are other commands that similarly show particular direct or indirect diagrammatic relationships between concepts. As can be seen from the above, the graphical form also suppresses the default '[not]' contrasting poles. The above problem is now explored using conceptual graphs. The conceptual graphs representation of the above problem is based on the same cognitive map as identified above. This approach should ensure a common comparative basis, yet highlight vividly any distinguishing features between the two representations.
3.04 Modelling the Poles in Conceptual Graphs

Starting with the poles themselves, they appear to fall into two categories. The first category has user-defined contrasting poles whilst the second’s contrasting poles remain undefined. Concentrating on the defined concepts to begin with, these may be modelled initially by the conceptual graphs in Figure 3.02. In this figure the pair of poles become a conceptual graph by placing each pole into a separate conceptual graph concept and together surrounding them within a negative context. These negative contexts signify that whatever is contained within them, taken as a whole, is false. Therefore each graph provides contrast by stating that it is false that both poles can exist simultaneously. As elaborated below, if one of the concepts is true then the other becomes false.

Figure 3.02: The initial conceptual graphs for the defined contrasting poles

Take the middle graph in Figure 3.02, which refers to the poles 'centralise services at Leeds, open local offices', as a representative instance. Let's say we decide to see what happened if 'centralise services at Leeds' was chosen. As a conceptual graph this could be shown initially as in Figure 3.03.

Figure 3.03: centralise services at Leeds

The true graph of Figure 3.03 dominates its matching concept inside the aforesaid middle graph in Figure 3.02. Hence this inside concept can be removed, or deiterated, to yield Figure 3.04.
Figure 3.04 shows that 'open local offices' is false. This occurred because 'centralise services at Leeds' is true. Should the decision be 'open local offices' instead then 'centralise services at Leeds' would be false accordingly. The whole picture for this scenario is shown in Figure 3.05. Unlike the earlier cognitive map that only passively records the poles, a computer-based conceptual graph processor could make these new assertions automatically as the appropriate new graphs are added to its base of knowledge. The important repercussions of these inferences will become evident later.

Note that the above 'true-asserts-false' form does not assert one pole as true should the other be false. For example the poles 'centralise services at Leeds' or 'open local offices' cannot be asserted as true from their contrasting pole being false. To do this would require the additional 'false-asserts-true' graph shown in Figure 3.06. In this figure there are nested negative contexts. Remember that in conceptual graphs, a whole negative context and its contents can also be deiterated provided it matches an appropriately dominating negative context and its contents.
This removal can be illustrated from the graph in Figure 3.04 (‘open local offices’ is false). This graph dominates the matching part in the false-asserts-true graph of Figure 3.06 because it is surrounded by a lesser number of negative contexts. Hence its matching graph can be removed from the latter figure to yield the result shown in Figure 3.07.

Figure 3.07:

centralise services at Leeds

This result has left two negative contexts around ‘centralise services at Leeds’. These double negate to give the same graph as in Figure 3.03 (‘centralise services at Leeds’ is true).

In the present cognitive mapping technique the ‘false-asserts-true’ aspect is insufficiently clear. The present approach may prefer the user to assume if one pole is false then the other is true, yet it is quite possible that the decision maker may for instance do nothing or decide to open mobile offices instead. In this case the above ‘false-asserts-true’ graph would be incorrect. The bipolar nature of the present method cannot cope with this scenario. Even worse, it could provide a too narrow framework which stifles originality of thought: The model does not lend itself to decision makers realising other alternatives, such as mobile offices. In view of this deficiency, the ‘false-asserts-true’ aspects cannot be transposed to the conceptual graph representation in a manner which guarantees validity.

3.05 Refining the Graphs

Moving on, it is possible to leave the conceptual graphs in this ‘true-asserts-false’ two concept form and manipulate them as elementary propositional logic statements\(^3\). Ackerman et al. stress that the sentences should remain as they are because the decision maker can identify with what he or she stated directly. With conceptual graphs the above concepts could be

\(^3\) The details of propositional logic and predicate logic can be found through any seminal text on logic (For instance, Kowalski 1979a). Sowa (1984, Appendix A.5: “Symbolic Logic”, pages 384–391) also discusses these matters.
refined nonetheless to the more powerful predicate logic level\textsuperscript{4} thereby capturing more about the problem, yet arguably remain human expert readable. The refinement is demonstrated by the graphs shown in Figure 3.08, which refine the graphs in Figure 3.02.

The graphs in Figure 3.08 now include relations, referents and coreferent links as well as essentially more proper hierarchical type labels. The left-hand graph inside the nested negative context of Figure 3.08(d) (or "8(d)" for short) may be read as "The characteristic of an office is a higher purchase cost" for example. The referent ‘Leeds’ conforms to the type label ‘central office’ and ‘Hull, Sheffield and Harrogate’ conforms to ‘local offices’. Part ‘8(a)’ is merely a shortening of one of the concept’s phrases. This graph could easily be refined further, as indeed may all the graphs throughout the entire offices example, hence ‘8(a)’ may be viewed as an example of an intermediate step in model development. The greater degrees of refinement are demonstrated by ‘8(b), ‘8(c) and ‘8(d). In ‘8(b), ‘Leeds’ is an instance of a central office in that Leeds will have its own peculiarities but shares the same characteristics as any central office in general. This would permit inferences to be made about Leeds from both what is known about central offices in general and Leeds in particular.

3.06 Generalising the Model

The above shows that a knowledge-base can be built up based on the appropriate degree of generally applicable knowledge. This also prevents

\begin{itemize}
  \item impaired client treatment risk
  \item ensure uniformity of treatment
\end{itemize}

\begin{itemize}
  \item central office: Leeds
  \item local offices: Hull, Sheffield and Harrogate
\end{itemize}

\begin{itemize}
  \item lower purchase cost'
  \item 'higher purchase cost'
\end{itemize}

\begin{itemize}
  \item local offices
  \item offices
  \item characteristic
  \item higher purchase cost
  \item lower purchase cost
\end{itemize}

\begin{itemize}
  \item central office
  \item office
  \item characteristic
  \item higher purchase cost
\end{itemize}

Figure 3.08: Refined conceptual graphs

\textsuperscript{4} See footnote 3.
unnecessary duplication when the same knowledge applies to more than one particular concept. The degree can be appreciated by developing the Leeds example in a little more detail. It may be that certain things are applicable to Leeds in its own right, Leeds as a Yorkshire central office, as a northern central office, or an English central office as well as a central office. The same principles apply to the local offices. Taking the central office case as representative, the type hierarchy would then include (where subtype < supertype):

central office < office.
English central office < central office.
Northern central office < English central office.
Yorkshire central office < Northern central office.

The most specialised conformity for ‘Leeds’ is ‘Yorkshire central office’. This means Leeds conforms to all of the above central offices, but not to say ‘Southern central office’ (Southern central office < English central office). Thereby any inference in respect of Southern central offices would not apply to Leeds but any for Yorkshire, Northern, central office and office would.

The graphs in ’8(c) and ’8(d) concern the purchase costs of the offices. Examining ’8(c), the left graph shows that if a purchase cost is higher then it cannot be lower and vice versa. The right graph shows that if one is false the other is true. The coreferent link in both cases establishes that they refer to the same cost. These graphs are therefore so general in nature that they can be used beyond the offices example.

Turning to ’8(d), these graphs imply that a central office is an office which has a higher purchase cost whilst local offices are offices with a lower purchase cost. Should ‘central office’ or ‘local offices’ dominate these graphs respectively, the appropriate inference would be made accordingly. This is demonstrated in Figure 3.09.

Conceptual graphs thereby also raise the user’s awareness through their inherent structure: As the user refined the graphs so they become more and more based on hierarchical type labels and specific instances within those labels, the user would have to think about the appropriate degree of relevance. The graphs as they currently stand apply to any local or general office. Alternatively they may be written to infer about Yorkshire offices only,
Chapter 3

The graph:

Yorkshire central office: Leeds

matches concepts in:

<table>
<thead>
<tr>
<th>central office</th>
</tr>
</thead>
<tbody>
<tr>
<td>office</td>
</tr>
<tr>
<td>characteristic</td>
</tr>
<tr>
<td>higher purchase cost</td>
</tr>
</tbody>
</table>

giving:

<table>
<thead>
<tr>
<th>central office: Leeds</th>
</tr>
</thead>
<tbody>
<tr>
<td>office: Leeds</td>
</tr>
<tr>
<td>characteristic</td>
</tr>
<tr>
<td>higher purchase cost</td>
</tr>
</tbody>
</table>

resulting in:

<table>
<thead>
<tr>
<th>office: Leeds</th>
</tr>
</thead>
<tbody>
<tr>
<td>characteristic</td>
</tr>
<tr>
<td>higher purchase cost</td>
</tr>
</tbody>
</table>

Figure 3.09: An inference involving hierarchical relationships and referents

in which case 'central office' and 'local offices' in the appropriate dominated graphs would instead read 'Yorkshire central office' and 'Yorkshire local offices' respectively.

From the limited information in the Ackerman example it would not be safe at this point, however, to include the graphs '8(d)' within the knowledge-base. Therefore the restricted forms shown in Figure 8.10 are included in their place. This figure prevents deiteration and its consequent assertion occurring unless the referent is specifically 'Leeds' or 'Hull, Sheffield and Harrogate' respectively.

3.07 Modelling the Undefined Poles in Conceptual Graphs

Continuing further, the concepts with undefined contrasting poles can be modelled initially in conceptual graphs as shown in Figure 3.11. The concepts 'greater delegation of authority' and 'follow current management initiatives' are not included in this figure and will be dealt with after the discussion on links below. Where the contrasting pole is not defined on input, COPE creates it by prefixing the term 'not' to the input emergent pole as stated earlier. As COPE embodies the present cognitive mapping methodology, this step poses a question. That question asks if this is most accurately replicated in conceptual graphs by taking the
input pole and encircle it with a negative context, as denoted in Figure 3.11. An alternative might be to write, without using a negative context, 'not emergent pole' instead. The graphs would then be based on the form given by Figure 3.12.

![Figure 3.10: Restricted graph to account for lack of knowledge in existing problem](image)

Unlike their defined counterparts, the 'false-asserts-true' scenario can be seen to be valid for undefined contrasting poles, hence the graph form in Figure 3.13 would also need to be added into the knowledge base for each alternative. This reveals the alternative is superfluous because the term 'emergent pole is false' clearly equates to 'not emergent pole is true'. Another question is thus raised asking if there is any need to include such poles in conceptual graphs anyway. For instance given 'local representation' was true or false, this would merely assert 'local representation' is true or false respectively. This tautology shows such concepts in fact turn out to be meaningless. Therefore they can be excluded from the conceptual graphs representation.

![Figure 3.11: Conceptual graphs for concepts with undefined contrasting poles](image)
3.08 Modelling the Links in Conceptual Graphs

The cognitive map links may be modelled initially as implications in conceptual graphs as shown in Figure 3.14. The nature of these graphs are explained by Figure 3.15. As can be seen from these figures, without worrying about the graphs affected by double negation for the moment, the 'leads from' pole becomes a concept which is enclosed in a negative context. This context also encloses another negative context that encloses the concept of the 'leads to' pole. The 'negative' link found in COPE becomes redundant because the order in which the poles are drawn are irrelevant in conceptual graphs. The user could still retain the visual order through arranging the shape of the graphs according as to what, say, that user would like to see at the top or bottom part of his or her graph drawings. The concept 'use local office experience' has been refined to 'use local office experience: #256' as it describes a particular office experience identified by the serial number '#256'. This number may be a reference to the relevant documentation on this issue for example.

As for the double negated graphs, the effect in the case of the graphs describing the false 'local representation', 'increased running costs', and 'too much decentralisation' implications of 'central office: Leeds' is they now appear to be like existing cognitive mapping concepts instead of its links. Hence these graphs show there are links that emerge to be additional contrasting poles. Conceptual graphs have yielded this fact explicitly and drawn it to the user's attention, whilst it remains unnecessarily implicit and thereby easily undetected in the existing cognitive map.
Chapter 3

General form of implication (illustrated by $X$ asserts $Y$, not $Y$ asserts not $X$):

![Diagram of implication]

Say $X$ in fact is equal to not $Z$. This would result in:

![Diagram of implication]

Similarly, say $Y$ was equal to not $W$. This would give:

(Before double negation) (After double negation)

Figure 3.15: Modified implication in conceptual graphs

Figure 3.14: Conceptual graphs denoting the links in the cognitive map

All the above of course highlights another question as to whether the present cognitive mapping technique should indicate that all default contrasting poles are accurate enough to attach other poles logically to it. There is the distinct possibility that such links could be erroneous, with the result of the user being mislead by the model. For instance, picking up on the mobile office dimension discussed earlier, it seems that ‘local representation’ [not] local representation’ leads to ‘improve level of service’ [not] improve level of service’ respectively. However it may be possible to have ‘[not] local representation’ and ‘improve level of service’ through ‘use mobile offices’. It would then be false that ‘[not] local representation’ implies ‘[not] improve level of service’. The dubiety of this whole premise is heightened somewhat when COPE apparently contradicts itself by suppressing such links in its own graphical output form. All this reveals a potentially serious flaw in Eden’s cognitive mapping method.
3.09 Modelling the Other Knowledge in the Example Problem

The concept 'use local office experience' has some background information relating to it about the source of that information from some actual offices in the South West. This may best be described by the graph in Figure 3.16. This graph can be added to the knowledge base and then called upon as necessary. If the earlier graphs in Figure 3.08(d) had not been restricted to those in Figure 3.10 then the graph in Figure 3.17 would be asserted from the background South West offices graphs in Figure 3.16. This instance underlines how useful the powerful hierarchical nature of conceptual graphs can be, as brought out earlier.

At this point all the concepts and links have been discussed apart from 'greater delegation of authority', 'follow current management initiatives' and the relationship between 'lack of understanding about risk' and the choice of office. The first two concepts are modelled by Figure 3.18. This figure reflects that these two concepts are a statement of fact, whereas the previous concepts depend basically on which office type is chosen. Furthermore, adding the usual implication graph shown in Figure 3.19 would imply, from the above preference graph that 'local offices: Hull Sheffield and Harrogate' would be true when it is really undecided. This situation can be avoided by making the statement apply to local offices in general. Although Figure 3.20 states there exist some local offices, it cannot purport that those local offices are in Hull, Sheffield and Harrogate.
It may be argued this is merely a convenient device on the part of conceptual graphs to get round this problem. Whilst this would not be denied, the inherent nature of conceptual graphs would demand eventually a more exacting analysis of the problem as part of the continuing overall refinement of the knowledge-base. Ensuing invalid inferences for instance would draw the user into finding out more about the situation or, if that is not possible, to apply expert judgement. The appropriate graphs could then be devised. In the above instance it may be wise to determine if delegation really must mean local offices. This can perhaps be provided somehow in a central office environment, or by mobile offices. Despite the existing cognitive map's link, reference to the office text does not sufficiently clarify this. Therefore this link, even in the above 'convenient' form, cannot be included in the conceptual graphs knowledge base.

The above convenient argument also applies to the concept 'office' in Figure 3.21 that describes the 'lack of understanding about risk' relationship. In COPE this implicit relationship in the cognitive map was refined into a connotative link, which the figure is intended to reflect. Even though there is an element of convenience in the graph, it is less ambiguous than the first one and therefore can be included in the knowledge-base. This, of course, is subject to appropriate user-initiated revisions as discussed already.
Chapter 3

A and B must be true before C is asserted as true (If C is false then so is A and B):

![Diagram with nodes A, B, and C]

Only one of A or B need be true before C is asserted as true (However if C is false then A and B is still false):

![Diagram with nodes A, C and B, C]

Figure 3.22: Capturing the 'and/or' distinctions omitted by cognitive mapping

The existence of the above 'lack of understanding about risk' graph would have an effect on the graphs which describe 'lack of understanding about risk. Use local office experience: #256' in Figure 3.14, in that it would cause 'use local office experience: #256' to be asserted as true. In view of this assertion 'use local office experience: #256' could appear as a true graph to begin with. However, this is not done for two reasons. The first is to show that as new knowledge is added, this has a dynamic effect on the present knowledge. The second is that it shows a line of reasoning that the user may wish to know about.

One other general point worthy of discussion arises when a pole has more than one link to it. This is whether all the 'leads from' poles have to occur, or just some of them, for that 'leads to' pole to occur. The present cognitive mapping approach does not bother to differentiate. In the cognitive map for the offices, the example has three concepts leading to 'centralise services at Leeds. Open local offices'. As it happens, only one of these, 'lower purchase costs ...... higher purchase costs ......', ends up as an implication in the conceptual graphs model as it stands. However, if there had been more a decision would have to be made as to how many of them would need to be true. This is illustrated by Figure 3.22. Even though it does not arise as yet in the offices problem, this 'and/or' distinction is yet another area where the present cognitive mapping technique may well mislead the user.
3.10 Allowing Inferencing

Now the conceptual graphs knowledge-base can sensibly start to infer new knowledge. For example, given 'central office: Leeds' is true then 'impaired client treatment risk' is asserted as false. This is shown in Figure 3.23. In this figure there are coreferent links to concepts labelled as 't', which may best be thought of as 'it' instead of the full concept's name, to aid readability by avoiding repetition. Furthermore, unlike conceptual graphs, Eden's cognitive mapping technique does not lend itself to proving false 'antecedents' from false 'consequents'. For instance, if 'ensure uniformity of treatment' was an over-riding consideration then 'local offices: Hull, Sheffield and Harrogate' would be false. This can be seen in Figure 3.24.

3.11 Visual Cues

Like the cognitive mapping graphical display illustrated by COPE earlier, the graphs can be arranged by the user to highlight particular parts of the knowledge-base. Such a user-initiated visual cue was mentioned earlier in respect of the negative links in cognitive mapping. Both Figure 3.23 and 3.24 may be viewed as visually arranged examples. Another example is Figure 3.25. This figure shows the direct links arising from the choice of offices. However, unlike the existing cognitive mapping technique, it can be

---

5 The 't' essentially equates to the universal supertype, but with an attached coreferent link that immediately specialises it.
seen that the conceptual graphs model shows a much richer set of interrelationships whilst remaining user-readable.

![Diagram of interrelationships between the local offices and treatment uniformity]

Figure 3.24: Interrelationships between the local offices and treatment uniformity

3.12 Merging Graphs

A feature of cognitive mapping is that maps can be broken down into separate 'clusters' to focus on sensible subsets of the problem. Of course conceptual graphs also supports such modular substructures provided, like cognitive mapping, the appropriate concepts are correctly duplicated. Another cognitive mapping feature is that separate maps, depicting different perspectives about a problem situation, can be merged. Like clusters, conceptual graphs does not hinder this process. In fact they can improve it through the type hierarchy.

For example, one individual may have mapped the offices problem as already discussed. Another may have mapped the contrast as one between 'mobile offices' and 'free telephone support'. Assuming there are not other concepts that may suitably relate the two maps, there is no apparent connection between these maps unless there is a compromise on the very terminology itself. However, by a hierarchical approach, each party may retain their own perspective but be able to agree at some other conceptual level. Here, they may concur for instance that 'offices' (as a supertype of both 'local offices' and 'mobile offices') would lead to 'face-to-face service'. Of course merging would still require graphs to be redefined, but within the much wider scope that the hierarchy allows.
Figure 3.25: Directly linked interrelationships surrounding the choice of office

3.13 Comments About Cognitive Mapping in Conceptual Graphs

In summary, from the offices example the conceptual graphs representation has managed to reveal:

a) Like the present cognitive mapping methodology, the concepts and relations in conceptual graphs can be based on a language that the decision maker identifies with. Conceptual graphs can also be arranged to retain the visual cues the end-user may require.

b) Unlike present cognitive maps, conceptual graphs allow the further refinement of the problem through, say, the interrelation of generalised and specialised knowledge. Initial conceptual graph models may start by being a literal paradigm of cognitive maps at the existing level. Subsequently they may be refined by graphs which, as illustrated by the
office purchase costs, break down these phrases along increasingly greater expressive dimensions.

c) Through its bipolar limit, which conceptual graphs overcome, the current cognitive maps could stifle creative thought by the decision maker.

d) The 'not] emergent pole', which is the default contrasting pole in the present maps, turn out to be meaningless when modelled as a conceptual graph.

e) Conceptual graphs do not need any additional devices to show the 'negative' links unlike the present cognitive mapping technique. Any user visuality element therein need not be compromised by this link's absence.

f) By always implicitly linking concepts with default contrasting poles current cognitive maps obfuscate the distinction between legitimate and potentially damaging relationships. Conceptual graphs, on the other hand, remove this arbitrary situation by focusing the user's mind on what in fact are valid and invalid contrasting poles, including default ones.

g) Even though conceptual graphs may permit convenient devices to 'quick-fix' obstacles in the mapping exercise, they ultimately cause the user to improve the quality of the knowledge itself and refine the graphs accordingly.

h) The present cognitive maps fail to distinguish between any 'and / or' relationships between knowledge elements. This relationship arises when a pole has more than one link to it and there is a need to determine whether all the antecedent poles have to occur, or just some of them, for the consequent pole to occur.

3.14 Concluding Remarks

Clearly conceptual graphs can enrich cognitive mapping. Though it may successfully elicit knowledge through its contrasting poles and links, cognitive mapping cannot extract properly the genuine impact of these relationships nor put the user on enquiry to seek for further dimensions that may affect the problem. Moreover it can be wrong, as the references to mobile
offices have revealed for example. A conceptual graphs processor could automatically recognise and deal with the contrasting aspects of the cognitive mapping technique. This would occur as a direct part of the negative contexts upon which conceptual graphs inference is based. As well as inference, the processor would also be able to check for any inconsistencies as they are entered into the knowledge-base. All this should free the user to declare merely what he or she believes and then review that mental model, or its computer paradigm, in the light of the processor's output.

Although critical of the current approach, this chapter does not seek to dismiss it. As Eden states, the present cognitive maps can be drawn quickly and thereby get an immediate handle on the problem situation at hand. Therefore it remains a valuable initial modelling tool. However as a more permanent building block of knowledge, its limitations are simply too significant to ignore. Conceptual graphs supplies a similarly visual but much more highly principled basis from which more meaningful knowledge can be eventually built.

Much has been made in this chapter about the generality of conceptual graphs. This is further examined in the next chapter.
Chapter 4

4 Conceptual Graphs and Economic Accounting

4.01 Introduction

This chapter seeks to determine how general the applicability of conceptual graphs could be for strategic management accountancy. As will be elaborated below, accounting is an abstraction of economic theory. The currently numeric-based and data-based accounting models, however, capture only the more straightforward parts of economics. The reasons for such limitations were dealt with in earlier chapters. The challenge to conceptual graphs examined now is whether they can provide more modelling power to the strategic management accountant, through eloquently capturing more of accounting's economic bases. The discussion starts, by focusing on one fundamental economic concept in an illustrative case study, how economic concepts might be modelled in conceptual graphs. The discourse then moves on too see how conceptual graphs may extend the above-mentioned existing accounting models, and thus reveal links with strategic knowledge like that shown by cognitive mapping in the previous chapter.

4.02 The Essence of Economics

Hyman (1992) and Lipsey (1988) show that economics is known to be far more extensively based upon the highly elusive dispositions of people rather than definitive physical laws. Rhodes (1985) illustrates how economics' precarious scientific basis has caused the economist to be the subject of much ridicule. Nonetheless Rhodes also shows that the economist's knowledge remains consequential, because economics is a reality that affects us all. The teaching style of Pool and La Roe's amusing text on understanding economics underscores this actuality, by drawing considerably on the formative, yet imprecise human instincts that their novice readers would already possess (Pool and La Roe, 1985). The indomitable conceptual bases that characterise economics thereby provide an excellent proving ground for conceptual graphs. By way of illustration we focus on one economic concept, namely that knowledge comprising what is popularly known as 'the law of supply and demand'.

1 Based on Polovina and Delugach (1993).
4.03 Supply and Demand

Supply and demand is an aspect of fundamental microeconomic knowledge that is a key element in modeling essentially any overall economic theory. As indicated above, it is also a good example of an economics that is well-known by the general public. It therefore represents not only a class of domain-specific knowledge, but also a corpus of 'common sense' general knowledge that should well suit conceptual graphs.

Put simply, the law of supply and demand holds that the price of some resource depends upon relationships between the supply of the resource and the demand for the resource. A resource might be some person’s time, an unmarried person available for dating, or anything valued by another person. In a supply and demand context, that resource becomes an economic resource. Ijiri (1975) clarifies economic resources as objects that (a) are scarce and have utility, and (b) are under the control of an enterprise. A recent compilation of American cultural information by Hirsch, Kett, Joseph and Trefil (1991) describes supply as “the amount of any given commodity available for sale at a given time,” and demand as “the amount of any given commodity that people are ready and able to buy at a given time at a given price.” The law of supply and demand is described as follows (Hirsch et al., page 434):

In classical economic theory, the relation between these two factors determines the price of a commodity. This relationship is thought to be the driving force in a free market. As demand for an item increases, prices rise. When manufacturers respond to the price increase by producing a larger supply of that item, this increases competition and drives the price down. Modern economic theory proposes that many other factors affect price, including government regulations, monopolies, and modern techniques of marketing and advertising.

This description is the starting point of the following case study because it concisely characterises a) the important parts of supply and demand, and b) by extension, the very essence that economics purports to be a view of popular culture based on 'common sense'. The case study describes supply and demand using conceptual graphs.
4.04 Case Study 5: Supply and Demand in Conceptual Graphs

The conceptual graphs in Figure 4.01 begin to describe supply and demand using conceptual graphs. Figure 4.01(a) shows that if there is an economic resource then it has both a supply and a demand. Figure 4.01(b) shows that if there is an economic resource whose demand quantity $x$ is less than its supply quantity $y$, then that economic resource is in a state of over-supply. Similarly, in Figure 1(c), if there is an economic resource whose demand quantity $x$ is more than its supply quantity, then it is in a state of over-demand.

Figure 4.01 reveals how conventional terms are used. For example, the terms under-demand and over-supply would have essentially the same meanings. The spatial arrangement of these graphs is also significant. The '(supply)' links are shown in the '*<symbol>' marker form instead of the dotted line alternative. Using markers instead of lines may be preferable where, say, drawing lines may result in highly cluttered graphs.

2 Out of interest note that, apart from Figure 4.01(a) which illustrates both, any coreferent links are shown in the '*<symbol>' marker form instead of the dotted line alternative. Using markers instead of lines may be preferable where, say, drawing lines may result in highly cluttered graphs.
relation and the ‘(demand)’ relation are both in the same spatial location in both Figure 4.01(b) and Figure 4.01(c). The term ‘less than’ could have been used instead of ‘more than’ in the right hand graph but there would still remain that visual similarity. The interrelationship between the (less than) and (more than) relations is similarly evident from our own tacit common sense knowledge. Figure 4.02(a) thus captures the natural meaning of ‘If measure x is less than measure y then y is more than x’. Likewise, Figure 4.02(b) captures ‘If y is more than x then x is less than y’.

The above is further illustrated by Figure 4.03 which shows the relationships between the notions of over-demand and under-supply, so that the economist may use any terms that are convenient. Figure 4.03(a) represents “If over-demanded economic resource a then not over-supplied economic resource a and vice versa”. Figure 4.03(b) represents “If under-demanded economic resource a, then not under-supplied economic resource a and vice versa”. Figure 4.03(c) represents “If over-demanded economic resource a, then under-supplied economic resource a”. Figure 4.03(d) represents “If under-demanded economic resource a then over-supplied economic resource a”.

The above would chain together backwards, forwards, or appropriate combinations thereof. The above simple examples vividly show that Peirce logic makes all these relationships much more evident to the domain expert than if-then rules. It may also suggest alternative viewpoints to the economist, once he or she is shown some alternate representations that are relevant. For instance, Peirce logic explicates to the user even the simple modus tollens relationships obscured somewhat by the procedural nature of an ‘if-then’ implication. Hence, say, what if it is false that quantity ‘x’ is less than quantity ‘y’, then quantity ‘y’ is more than quantity ‘x’ would be asserted as false. Careful examination of the
graphs presented in this case study will reveal such and more involved 'what if' interrelationships not evident from if-then rules alone.

Figure 4.04: Representing over-supply  Figure 4.05: Representing over-demand

Figure 4.04 reveals that if an economic resource is in over supply then its price decreases or less units of it are produced. Note that the term 'price' is used instead of 'money'. This is because price may be derived from a non-monetary exchange, and that money itself is anyway fluid through exchange rate movements and inflation. Of course even this seemingly clear-cut point should not be treated as a completely hard and fast rule, for the reasons of user tacit awareness already indicated. Figure 4.04 shows how the state of having an over-supplied economic resource causes the price and quantity to decrease toward their equilibrium amount. Figure 4.05 shows a similar result for the over-demanded economic resource. As with over-supply there is the effect on price or units produced due to over-demand, albeit now an increase not a decrease.

The above graphs form a basis from which more refined graphs would need to be constructed perhaps according to particular situations. Not surprisingly, there is much more to supply and demand problems than as discussed. For example, for the over-demand case already denoted by Figure 4.05, a slightly more complex situation is described in Figure 4.06. From Figure 4.05, it can be seen that over-demand leads to either an increase in price or units produced. However it may be that these two parameters are fixed, say by government policy. Therefore a third possibility can be introduced that states a certain amount of demand for that economic resource could remain unsatisfied instead.
Moreover there are instances where supply and demand do not follow the expected pattern, such as the decreasing of the price of an economic resource may turn out to decrease its demand. This may be because dropping the price of a product may cause customers to believe it is now second-rate and thereby avoid buying it. Hosking (1992) reports on the concern of the luxury fragrance houses that large discount health and beauty retail chains wish to sell those houses' perfumes at a substantial discount of up to 50%. Hosking notes that the advertising media take the houses' attitude so seriously that they will not accept advertisements by the discount retailers, for fear of losing the lavish advertising business of the houses. Becker (1991), and The Economist magazine (1992, February 8, page 85), describe in a similar vein the highly-fashionable, or "in" commodities which, as their price increases, their demand increases. However this only occurs if demand is never satisfied, otherwise the price and quantity demanded drops again.

The above dynamic well exemplifies how the pervasive yet elusive tacit/implicit knowledge in a problem domain, here in terms of what is presently in fashion, affects the fundamental methodology that attempts to model that domain. It underpins the importance of models that can be built by the domain experts directly. Figure 4.07 is presented lastly as one thought-provoking way towards how such scenarios might eventually be captured by an economist using conceptual graphs.

4.05 Comments Arising From Supply and Demand, & Case Study 5

The last case study has shown how some basic common knowledge, namely the law of supply and demand, may be represented naturally through conceptual graphs. Negative contexts, unlike if-then rules, specifically
explicated less apparent relationships between economic facts. Although the case study has only touched upon the economics of supply and demand, let alone economics in general, the discussion reveals how conceptual graphs can meaningfully allow the human expert to tackle this very difficult problem domain. With all the above in mind, the discussion now moves away from illustrating economics in general to the more detailed relationship between economics and accounting.

4.06 The Essence of Accounting

Remember that the major benefit of adopting a structured model of a problem is so that such a model, by its inherent nature, draws out all the problem's relevant parameters from which a solution can be investigated fully. Contrast this with a written or spoken text discussion of the problem where it is well known that ambiguity and obfuscations can occur easily. This 'natural language' interpretation of problems may be the most flexible and easily followed, but without at least a basis in some structured form it can be dangerously erroneous. Hence the emergence of disciplines such as accounting that attempt to model the dynamics of economic activity in a structured way. The model, of course, must also be structured on a suitably principled basis. Otherwise it will omit or misinterpret the salient issues of the problem situation.

---

3 Based on Polovina (1992c).
4.07 The Double Entry Bookkeeping Model

As noted by Lee (1986), it was with the above in mind that the traditional model of accountancy, the bookkeeping model, was developed in the Middle Ages. The principle behind this model is economic scarcity. In other words for every benefit a sacrifice has to be made. For example, the benefit of a business owning its office is sacrificing £1,000,000 that could be employed elsewhere; a book prepared by its author researching a new exciting area in semantic understanding may have involved that author deciding against many complex yet important alternatives, such as the costs of, say, not participating in his or her growing family. These ‘transactions’ occur because the decision-maker makes a value judgement that the benefits outweigh the costs.

The bookkeeping model appears simple but rigourous. Fundamentally, instead of recording one amount per transaction it records two: A ‘debit’ and a ‘credit’. Moreover these amounts are complementary to one another, hence they ‘balance’ against each other. An accounting ‘balance sheet’ is merely the aggregate of all these debits and credits. The rigourousness derives from this principled ‘double entry’ structure so that each benefit is accounted for by a cost and vice versa. Hence every gain is matched to a sacrifice.

4.08 Problems with the Bookkeeping Model

However on deeper investigation the double entry bookkeeping model is unlikely to capture all these economic value trade offs. Say the business in the first example above decides to sell its office. This transaction can be recorded easily by the elementary bookkeeping entries “DEBIT Cash £1,000,000, CREDIT Fixed Assets £1,000,000”. The second, preparing the book, is simply too qualitative to be recorded by the bookkeeping model yet the author may want to know clearly about all the actual costs and benefits of such a transaction. This neglect on the part of the bookkeeping model is elaborated on below.

The threshold where the bookkeeping model may break down is perhaps lower than may be thought. Reconsidering the first example about the office, the value of selling the current office may be the purchase of cheaper offices for £500,000. In this case the bookkeeping is basically “DEBIT Fixed Assets £500,000, CREDIT Cash £500,000”. Now say, by spending the remaining
£500,000 elsewhere, the business generates a revenue of £600,000. On aggregate in the balance sheet the business's money worth then increases by £100,000 (Represented essentially as “CREDIT Profit and Loss Account £100,000, DEBIT Assets £100,000”). However if the value of the current office is retaining key employees through a comfortable work environment then, as in the author example above, the bookkeeping model is inappropriate. Therefore the double entry bookkeeping model is easily liable to make significant errors of omission.

Furthermore the bookkeeping model could mislead. Reconsidering the 'preparing the book' example the value may be viewed as the more easily quantified cost of the author ceasing to conduct consultancy work at £2,500 a time instead. This revenue would have been recorded by the bookkeeping model on an ongoing basis. However the book might bring its author satisfaction of a deep desire for an enhanced reputation amongst peers. Unless this can be translated into a cash benefit the bookkeeping model would not record these judgements.

Figure 4.08(a): The events accounting model in entity-relationship form.

Figure 4.08(b): Transformation of the events accounting model into supertype conceptual graphs.

Figure 4.08(c): Transformation of the events accounting model into subtype conceptual graphs.
and thereby leave a 'loss' of £2,500. By choosing to author the book the
decision-maker *qualitatively* has to justify, *against the grain of the*
bookkeeping model's *assessment* of *value*, why that £2,500 has been forsaken
even though this may the lesser value item. Therefore the double entry
bookkeeping model, taken too literally, can also readily lead to significant
*errors of commission*.

4.09 The Events Accounting Model

The above problems are familiar to most accountants. The *Economist* (1992,
February 8, page 104), for instance, summarises the difficulties accountants
have in attaching a monetary value to 'intangible' assets such as product
propose the *events accounting* model. Unlike the bookkeeping paradigm,
events accounting attempts to capture the qualitative dimensions of
economic scarcity. The model is shown by the diagrams that make up
Figure 4.08.

Figure 4.08(b) and Figure 4.08(c) will be discussed below. Figure 4.08(a) shows
the events accounting model as an *entity-relationship diagram* (Chen, 1981).
In line with the earlier discussion about the desirability of structured models,
Chen (1985) argues that the pictorial nature of entity-relationship diagrams
are particularly useful in structuring problems qualitatively stated in natural
language. Given this actuality, events accounting represents a powerful
means of recording scarcity as more than a monetary measure. Setting aside
its 'dotted' part for the moment, Figure 4.08(a) reveals the fundamental links
between an 'economic resource', which means some exchangeable item of
value, and the parties which create the 'economic event' that causes the
economic resource to be exchanged.

4.10 The Events Accounting Model as Conceptual Graphs

Sowa (1985), in response to Chen's argument (Chen, 1985), adds conceptual
graphs can further structure the dimensions of natural language-based
problems. Sowa (1990) demonstrates that conceptual graphs extend
entity-relationship diagrams by adding the capacity of first-order logic in a
pictorial way. It is therefore sensible to transform the events accounting
model into conceptual graphs form. The conceptual graph in Figure 4.08(b)
thereby represents this transformation. This graph is basically a conceptual graphs reproduction of the general supertype parts of Figure 4.08(a), except that the arguably more definitive term 'event subject' is substituted for 'sale line item'\(^4\). Figure 4.08(c) basically expresses the specialised subtypes of the top diagram. It also refines 'party to' into 'source' and 'destination', and thereby shows the route by which the economic resource changes possession. For both the conceptual graph diagrams, certain aspects in the entity-relationship model are not reproduced to focus on the salient nub of events accounting.

---

\(^4\) The relation '(party to)' illustrates a 'triadic' relation, as it is also linked to a third concept, instead of the previously discussed relational form which are of a 'diadic' nature. Such \(n\)-adic relations (where \(n \neq 2\)) are uncommon in the accepted form of conceptual graphs, but the theory does allow for them (See Sowa, 1984, pages 72 & 114).
4.11 Duality

The conceptual graphs within Figure 4.08 do not yet fully capture events accounting. To achieve this the duality principle of economic scarcity needs to be considered. Duality is defined by the statement "for every benefit there has to be a sacrifice made". The events accounting model indicates duality in the 'dotted part' of Figure 4.08(a). This 'cash receipt pays for the sale' is really a shorthand to make that top diagram concise. For instance 'party to' should also connect to 'cash receipt' because it is also part of the exchange. Figure 4.09 reveals this duality in full at the supertype level. In Figure 4.09 both sides of duality are shown by linking the economic events to the same transaction. As already indicated, the 'Economic Resource' concepts may be specialised to any quantitative or qualitative concept describing an item of value. As one value describes the benefit in the transaction, the other value depicts what had to be sacrificed for that benefit. An example scenario is portrayed in Figure 4.10.

4.12 Intangible Values

Figure 4.10 develops the earlier example of a business wishing to realise cash by selling its expensive offices, with a view perhaps to purchasing cheaper ones. The diagram shows a £1,000,000 cash benefit to 'OurCo Ltd.' for the loss of their high-class offices at '1, Prestige Plaza'. As such, it may easily be recorded by the bookkeeping model. However this may not be the full picture. In addition to the loss of the present offices, there may be an intangible loss due to disgruntled key employees leaving the corporation as a consequence. As discussed earlier, such intangible values remain outside the bookkeeping model. Nevertheless the business could require such factors to be included. Claret (1992a) points out an intangible liability may, for example, determine the very survival of the business itself. The conceptual graph of Figure 4.11 manages to record this qualitative cost.
4.13 Comments on the Events Accounting Model

The events accounting model, in summary, does not restrict the recording of transactions to only those that can be captured by numeric monetary measures. Thereby its wider scope enables qualitative value judgements to be modelled and hence be gauged along with the quantitative elements. In databases, such parameters are defined in data fields which can be then subjected to some selection criteria. Within any knowledge-base, in addition to this 'lookup' facility, there should exist the ability to make adequate inferences for new knowledge. Hence if accounting transactions can be presented in a knowledge-based form they can be subjected to this inferencing. Peirce logic, which is the inference mechanism associated with conceptual graphs, will be seen to be a powerful part of the pivotal area that performs this role. This area, strategic planning, is addressed in the following sections which begin by introducing into the current discussion the cognitive mapping technique of the last chapter. To preserve the flow of argument, some of this introduction necessarily repeats on that of the previous chapter.

4.14 Introducing the Cognitive Mapping Dimension

Cognitive mapping is a practical technique derived by Eden (1991) from the personal constructs theory of Kelly (1955), and is used by strategic planners to structure the highly subjective problems that characterise strategic planning. The cognitive map element shown in Figure 4.12 continues the office relocation example discussed so far.

In Figure 4.12 the left-hand concept contains the emergent pole 'buy down-market offices at 13, Sidestep Row'. The contrasting pole, separated from the emergent pole by '...', is 'retain high-class offices at 1, Prestige Plaza'. Where a contrasting pole is unspecified, as in the case of 'disgruntled key employees', then the contrasting pole is determined to be 'not <emergent pole>'. Hence the contrasting pole for the latter concept becomes 'not disgruntled key employees'. The link leads
from the 'buy cheaper offices at 13, Sidestep Row... retain current offices at 1, Prestige Plaza' concept to the concept with the emergent pole 'disgruntled key employees' and default contrasting pole 'not disgruntled key employees'.

From the above the down-market offices lead to disgruntled key employees whereas the current offices lead to these employees not being disgruntled. After refining this statement from the elementary form illustrated by Figure 4.12 into the more logically advanced conceptual graphs, the choice of offices and its consequence may be remodelled as given by the conceptual graphs of Figure 4.13.

In Figure 4.13(a) the pair of specified poles denoting the choice of offices are restated as a conceptual graph by placing each pole into a separate conceptual graph concept and together surrounding them within a Peirce negative context. The single specified disgruntled employee pole and its unspecified 'not' contrast is also restated in Figure 4.13(a) although, as explained in the previous chapter, this graph turns out to be a tautology. Figure 4.13(b) demonstrates, in conceptual graphs, the cognitive map links as two implications based on the Pierce logic operations of deiteration and double negation. The conceptual graphs show both implications to be generalised, assuming this step is valid, to show that for any down-market offices the key employees located there are disgruntled whilst in any high-class offices they
are not (modus ponens). Similarly if such employees are not disgruntled then they are not located in down-market offices (modus tollens).

Conceptual graphs provide the contrast in cognitive map concepts by stating that one pole must be false if the other is true. However when both poles are specified the converse, that if one is false the other is true, cannot be determined. This distinction, which is not clear from the elementary cognitive mapping model, is intended. The business (referred to earlier as ‘OurCo Ltd.’) may, in the event, dispose of 1, Prestige Plaza and obtain offices other than at 13, Sidestep Row or even not any other down-market offices for that matter. Hence it may be quite wrong to assert that they will be at either 13, Sidestep Row or 1, Prestige Plaza. Furthermore conceptual graphs allow each pole to be generalised or specialised to differing degrees so enabling a potential continuum of contrast. Not mentioned in the previous chapter but now evident from the present example, this contrast happens to accord more precisely with Kelly’s original personal construct theory (Kelly, 1955). There may be, for example, a more general contrasting pole to a more specific emergent one. Ultimately the most general contrasting pole is the ‘not <emergent pole>‘. As for the cognitive mapping links, the modus tollens inference is arguably not evident in the elementary form given by Figure 4.12.

As well as the ability to a) generalise and specialise problem scenarios to the appropriate degree and b) vividly express modus tollens, conceptual graphs further reveal that c) there is no real distinction between the concepts and links of cognitive mapping anyway. As should be evident from a careful examination of Figure 4.13, these links turn out to be like emergent and contrasting poles. Hence conceptual graphs remove the arbitrary choice as to what a link or a pole. Moreover strategic planning problem situations can now be captured alongside the accountancy domain through the single medium of conceptual graphs. This opens the way towards suggesting a bridge between the two techniques.

4.14 A Bridge Between Events Accounting and Cognitive Mapping

An accounting methodology that incorporates the events model, rather than merely the bookkeeping model, enables the more qualitative elements of transactions based on economic scarcity to be captured along with the more
quantitative elements. Conceptual graphs enhance the entity-relationship model and include a visual method of inference in the form of Peirce logic. It thereby widens an organisation's numeric-based financial record systems into a corporate knowledge-base that can now include the parameters of strategic planning.

Given the discussion so far, it is not expected that inference will play a major role for the qualitative elements in accounting transactions. Such inference becomes significant instead within strategic decision making where, as also discussed, it in fact plays the major part. Yet presenting accounting events as conceptual graphs because of the inference capability remains worthwhile. This is because their qualitative aspects can then be interrogated and reviewed by the conceptual graphs derived from strategic analysis. For example, should the graph $-\text{[Disgruntled Key Employee]}$ be asserted as a parameter of a strategic decision then, given the graphs so far for the office location example (and OurCo Ltd. Disgruntled Key Employee $<$ Disgruntled Key Employee), the transaction disposing the offices at 1, Prestige Plaza would be blocked.

From all the above, conceptual graphs have revealed the fundamental difference between accounting and strategic planning is the former is essentially descriptive in nature and can thereby be captured by knowledge representation alone whereas the latter is prescriptive and requires the dynamics of inference. However the benefit is not one way: Accounting could offer strategic planning a firmer conceptual basis from which its dynamic models might be built.

4.15 Concluding Remarks

Through facilitating strategic management accountants to build and review conceptual graphs familiar to them, even some rather esoteric accounting notions may thus be effectively modelled. Furthermore the general applicability of conceptual graphs allows eloquent bridges to occur between accounting, economics and strategic planning. Nonetheless the understandability of conceptual graphs still needs to be evaluated amongst actual domain experts to either vindicate or deny the technique's worth. The thesis now turns its attention to this subject, beginning in the next chapter.
That chapter examines what kind of conceptual graph environment might best facilitate the human domain expert.
5 Determining the Methodology to Test the Conceptual Graphs Approach

5.01 Introduction

The previous chapters indicated the potential benefits of conceptual graphs to the strategic management accountant. The subject matter of this chapter refines those general arguments into the practical environment in which conceptual graphs may best be presented to the strategic management accountant. The discussion starts by examining the practical considerations involved when the domain knowledge is transferred between the domain expert and the computer, moving onto the particular appropriateness of conceptual graphs. The need for conceptual graphs as computer software is then identified, together with the inadequacy of existing such software. From all these bases the Conceptual Analysis and Review Environment, or CARE for short, is devised. The CARE software employs the more promising conceptual graphs processor, JEHCGP, which is also discussed. CARE is the vehicle through which conceptual graphs are user evaluated in the following chapter.

5.02 The Practicalities of Eliciting and Reviewing Knowledge

With any knowledge-base system there must be some means of sufficiently transferring domain knowledge between the domain expert and the computer. As mentioned in Chapter 1, this practical process is known as knowledge elicitation. Jackson (1990) summarises the experiences with an early knowledge elicitation technique, protocol analysis\(^1\), as follows (page 23):

Newell and Simon generated a kind of knowledge representation known as *production rules* \(^2\). They also pioneered a technique known as *protocol analysis*, whereby human subjects were encouraged to think aloud as as they solved problems, and such protocols were later analysed in an attempt to reveal the concepts and procedures employed. This approach can be seen as a precursor of some of the knowledge elicitation techniques that

---

2 Discussed in Chapter 1.
knowledge engineers use today. These psychological studies showed just how hard the knowledge representation problem was, but demonstrated that it could be addressed in a spirit of empirical enquiry, rather than philosophical debate.

The knowledge acquisition stage has also been referred to as the 'bottleneck' in knowledge-based applications development (Feigenbaum, 1977). Hence knowledge elicitation is itself no trivial exercise, and thus needs to be properly considered. Its importance cannot be overlooked: This thesis has already demonstrated the fundamental dangers arising from the domain expert's knowledge being inadequately computer-encoded.

In their study of failed management accounting expert systems, King and McAulay (1992) suggest two key properties that an expert system should possess to overcome the knowledge elicitation bottleneck. Their first proposal is for a pre-installed library consisting of an established academic body of domain knowledge, from which the domain expert could build a tailored model according to his or her particular circumstances. King and McAulay explain that such knowledge is more thorough than the relatively 'patchy' knowledge gained by the expert in the field. Their second proposal is that the domain expert should somehow be able to enter his or her knowledge into the computer via a 'checklist', as this is the method by which human experts present their knowledge. In essence, the expert would communicate to the system simply via a list of domain knowledge elements considered as relevant to some problem of interest. Thus, apart from any familiarity it may have with the domain expert, the checklist approach portrays a practical means of freeing the domain expert to concentrate purely on domain knowledge. The domain expert is thereby not burdened with the added computational overload hurdle first identified as fundamental in the early part of Chapter 2. However those checklist-based expert systems must not be too shallow, as we know from Chapter 1. As a reminder, Leith (1991) aptly makes the point (page 190):

... the systems which are being called expert systems are no more than checklists, and the explanations they give, no more than reading off the ticks or crosses on the checklist.

---

3 The role of the 'knowledge engineer' was mentioned in Chapter 1.
Bearing Leith’s criticism in mind, King and McAulay’s proposals are further discussed later. As already established by this thesis, Sangster (1991) discusses the need for human domain expert review in management accounting. Sangster typifies the warning that management accounting knowledge does not lend itself easily to formal models. Thus many studies, such as Back-Hock (1991) and Levary and Madeo (1991), suggest knowledge-based systems should be adaptive in nature. In this portrayal the knowledge in the computer is built up from continual iterations of activity with the human expert, rather than that expert having to accept an outright formal model at the outset. Back-Hock explains one such instance, the ‘product life cycle’ model. The formal knowledge inherent in this model needs to be constantly tailored by the real-world user, so leading to what Back-Hock calls the ‘integrated product life cycle concept’.

From all the above it can be seen that there appears to be a need for both relatively static and dynamic elements of knowledge. However, The Economist magazine (1993, March 20) lucidly reveals that there is little agreement about what business strategy is. The strategic problem domain thereby renders truly static domain knowledge as significantly inappropriate, so the scope for King and McAulay’s first proposal above about a body of established academic knowledge becomes severely limited. All this heightens not only the need for an interactive environment, but also one that the domain expert can be comfortable with given its consequently expected heavy use.

The discussion in this thesis has shown domain experts working directly with expert systems. O’Brien, Candy, Edmonds and Foster (1992) reveal the advantages of this situation. O’Brien et al. refer to such systems as ‘End-User Knowledge Manipulation Systems’, or EUKMS for short. They state (page 4):

Minimally, the definition of EUKMS is the provision of software support which enables a knowledge worker\(^4\) to encode, refine and manipulate domain knowledge, in a machine-usable form, without the continuing mediation of the knowledge engineer. EUKMS diminish the conceptual gap between a knowledge worker’s expertise and a machine-tractable representation of that knowledge, by providing a medium of communication which is founded upon domain concepts and terminology. Our concern

\(^4\) ‘Knowledge worker’ is their term for domain expert.
extends beyond the provision of tools which knowledge-workers can use, but do not otherwise augment the cognitive communication between user and system. A general goal for the design of EUKMS is to provide an interactive knowledge acquisition tool or environment with which knowledge can be elicited, encapsulated and encoded, using domain compatible external knowledge representations which are meaningful to the user. The computational representation required by the target system is independent of the external knowledge representation, and should be concealed from the knowledge worker.

The processes of eliciting and reviewing the kinds of knowledge that the strategic problem domain demands means that, in practice, an expert system would best support the strategic management accountant by such systems being highly interactive with that domain expert. This integration between human user and computer could best be achieved by allowing domain experts to directly state and query elements of domain knowledge in user-familiar terms. As the interaction developed, the human-computer dialogue could extend beyond superficial checklists of knowledge. Therefore King and McAulay's second proposal above, about the advantages of a domain expert entering a checklist of that expert's knowledge, would thus also be effectively satisfied without making the expert system too trivial. This is especially because the user would be interacting with domain knowledge alone.

5.03 Clarifying The Appropriateness of Conceptual Graphs

Although the matter has already been referred to extensively in this thesis, we now turn to emphasise how conceptual graphs could assist in the above practicalities of eliciting and reviewing knowledge. To begin with, a psychological study by Novak and Gowin (1984) demonstrated how the structured diagrammatic nature of their simple 'concept maps' elicited key concepts and relationships even to young children. Examples, illustrating the essential contours of these maps, appear in Figure 5.01.

Sowa (1991a) explains however that such concept maps, unlike the equally diagrammatic conceptual graphs, cannot express all of logic and natural language for serious analysis. Given Sowa's argument, the tough problem domain of business strategy would certainly require such levels of analysis. There is a particular significance in the above psychological basis for strategic problems. This is because, as already identified, the model must also enable
strategic management accountants to build and review their own knowledge bases in view of the high level of continually shifting tacit knowledge that strategic problems embody. Together with their advanced technical power, conceptual graphs' above argued user-readability should best enable the domain expert to integrate with knowledge-based systems beyond the checklist level. The following discussions expound upon the dimensions of this relationship.

To start with, consider the following factual illustration. During the formulation of Polovina and Delugach (1993), its authors debated the choice whether to employ 'quantity' or 'units' in that paper's conceptual graphs modelling 'supply and demand'. These same graphs appeared in the early part of Chapter 4, where supply and demand was also dealt with. It was discovered that a case could be made for either, merely according to individual inclination. As a result the authors decided that a deep semantic analysis of 'quantity' or 'units' outright mattered substantially less than what those terms symbolise psychologically to each user through his or her own conceptual graphs models. As corporate knowledge bases evolve there could nonetheless be some standardisation effort amongst the users to gain the benefits of wider scale knowledge. This would, as a consequence, involve standardising terms as well as graphs. From the arguments about strategic knowledge above, this would present a real challenge to those participants. Nevertheless, all would still be as agreed by a consensus of users, not as dictated by some well meaning knowledge
engineer. In this way, the benefits of King and McAulay's proposal about the benefits of an established academic library of knowledge may emerge.

Then there is the scope in conceptual graphs for hierarchical analysis, as already explained throughout this thesis so far. Although conceptual graphs are modular anyway, the use of the hierarchy allows even more compact graphs at a range of abstractions. As a result, the hierarchy should yield more user-readable structures. Further support for the success of hierarchical analysis is evidenced by the existing commercial software 'IdeaFisher', which is essentially based on this form of analysis alone. However, the tricky nature of strategic knowledge may preclude sufficiently neat hierarchies. Thus how effectively the hierarchy aids user-readability will be of particular interest during the user evaluations of Chapter 6.

In Chapter 2 the accountant's flowchart problem, allowing the accountant to draw badly structured diagrams, was discussed. Reference was made accordingly to standard program construction texts such as Stone and Cooke (1987). It will be found from such sources that the problem arises because those flowcharts are inherently procedural in nature. The declarative nature of conceptual graphs, however, causes the accountant to avoid drawing badly structured graphs automatically by simply sidestepping this obstacle. Furthermore the route of reasoning in the accountant's flowchart has to start at the very beginning and stop at the very end. Should a given query fall exactly into this control pattern, then arguably the accountant's flowchart can be seen to be satisfactory. Should the query, however, start or end somewhere in between then this procedural limitation becomes apparent. Hence the domain expert's attempts at interpretation using the accountant's flowchart with such scenarios can become extremely difficult. Again the indomitable nature of strategic knowledge is highly unlikely to be routed so narrowly.

Regarding the matter of inference, we saw that conceptual graphs can be more succinct than deploying if-then rules. As explained from Chapter 2, inferencing in conceptual graphs employs negative contexts, adapted by Sowa

---

5 IdeaFisher runs on the Apple Macintosh, and is available from 'Camelot' (Tel: 0800 565656).
6 The flowchart is an algorithm. Kowalski (1979b) shows that "algorithm = logic + control". Thus it is the control provided by the accountant's flowchart that would be found wanting in respect of strategic knowledge.
(1984) from the logic of Charles Sanders Peirce. As a reminder, the rule "If $P$ then $Q$" could also be expressed as "not($P$ and not $Q$)". This transformation may appear to be initially counter-intuitive to non-logicians, but Peirce relied on converting other logical relationships into AND and NOT form to obtain the visuality of his logic. To bridge the gap, Sowa (1984) proposed explicit relations in conceptual graphs such as one for 'if-then'. Therefore in the linear notation:

$$[P] \rightarrow \text{(implication)} \rightarrow [Q].$$

Here the graph elements '-> (implication) ->' denote the antecedent $P$ and the consequent $Q$, hence 'If $P$ then $Q$'. In Peirce-based form the above implication essentially reads:

$$\neg([P] \rightarrow [Q]).$$

or, in the more convenient '(...)' form discussed from Chapter 2:

$$([P] ([Q])).$$

As noted in Chapter 2, Farques et al. (1986) go further and dismiss Peirce logic completely, replacing this aspect of conceptual graphs with 'if-then' rules on the grounds that Peirce logic is unnecessarily complicated. However, as already shown, Peirce logic uncovers even the simple 'modus tollens' relationships obscured somewhat by the shallow procedural nature exhibited from chains of 'if-then' implications. We can further see that even Sowa's 'compromise' form shown above does not truly help, yet such explications can be seen to be vital for modelling strategic knowledge.

Furthermore, as well as conceptual graphs basis in domain expert-familiar structured diagrams, the method of negation in Peirce is arguably similar to that in the accountant's bookkeeping model, where figures are negated by surrounding them in brackets. For example the complementary double entry of £3,000' is '(£3,000)' (Lee, 1986). The conceptual graphs inferencing advantage over if-then rules also reveals benefits in the conceptual graphs-based system's output, because of the potentially greater number of ways a system could output an if-then based explanation than conceptual graphs. Thus there is less risk of such software replying in a sequence that the user might not expect.
Lastly, structured diagrams are further demonstrated by Hammer and Janes (1990) and Schwartz (1992) to be notably useful in the type of interactive human-computer integrated environment detailed above. The particular usefulness of conceptual graphs in that integrated environment are encouraged by Slagle, Gardiner and Han (1990), Loucopoulos and Champion (1990), and Champion (1991). In employing protocol analysis during the knowledge elicitation phase of their study, Slagle et al. note (page 37):

> While protocol analysis is difficult, conceptual structures\(^7\) offer a good mechanism to facilitate that analysis and create a knowledge specification.

Although we have mainly come across the above discussions beforehand in this thesis, these extra arguments have now focused and further underpinned the potential value of the conceptual graphs-based interface to the strategic management accountant. The remainder of this chapter details the interface’s design in computer software.

5.04 The Need for Conceptual Graphs Software

Although accepting that conceptual graphs should be software-based from all the arguments presented so far, it is briefly worth stating why this is so explicitly. The case really centres around whether it is sufficient simply to draw graphs manually and the human user seek insights merely by inspecting the result. After all, this is how the accountant’s flowchart and the many other graphical notations discussed in this thesis are employed in practice.

In answer to the above it can be seen that the particularly complex nature of strategic decision making causes this problem domain to lie well beyond human cognitive limits, as explained in Chapter 1. Thus a software knowledge-based system tool would be highly beneficial to the strategic management accountant. Apart from the strong evidence suggesting that this tool should be conceptual-graph based, manually having to convert any notation into machine-comprehensible form can only serve to introduce a delay that can significantly undermine the human-computer integration

\(^7\) i.e. conceptual graphs.
requirement stressed earlier. This is because, as Murray and McDaid (1993) indicate, in the visually-based environment the system and the user would work together in real-time for integration to be ultimately effective.

5.05 Existing Conceptual Graph Software, e.g. CAMES

Now that the need for conceptual graphs in software has been set out, attempts were first made to seek out any such existing suitable software. As The Department of Computer Studies, Loughborough University, had an active group working on this very software, it was decided that the most sensible search should be one that started locally. Hence the existing conceptual graph processor, written in prolog, of Smith (1988) and then Heaton (1989) was first examined. This software was simply called 'cgp', which stands for 'conceptual graphs processor'. However cgp was dismissed on the fundamental grounds that the user had to instruct the software as to which graph operations actually needed performing. For instance, the domain expert would have to tell cgp that graph 'a' is to maximally join with graph 'b'. Then the result, graph 'c', would be instructed by the expert to try being projected into by graph 'd'. Should graph 'd' have successfully projected, then 'd' would be told by the expert to deiterate from the compound graph 'e'. Then the expert would tell the result 'f' to double negate itself to assert the result 'g'. Clearly this represented a too-high level of computational overload on the domain expert, as established above. In the instance, the expert is merely interested in the domain knowledge of whether 'g' can be asserted from the the graphs 'a' and 'b' given 'e'. It is up to the software, not the human user, to determine the computational knowledge remainder.

The conceptual graphs-based expert system CAMES, which stands for Client Admin Expert System, was then investigated (Smith, 1991). This expert system, a subsequent incarnation of Smith's work on cgp above, was in commercial use at the blue-chip news company 'Reuters'. As part of its activities, Reuters supplies financial information to dealers, banks and brokers worldwide. Reuters also market to their subscribers complete trading room systems and interfaces to Reuters' own computers, as well as services and equipment supplied by various competitors. It is in this role that CAMES
is employed in that it checks, when those subscribers order further equipment, that the new kit is properly compatible with their existing setup\(^8\).

However, CAMES did not offer the full facilities of conceptual graphs theory. Specifically, it excluded the Peirce-based inferencing. Instead it relied on *refutation* which, as implemented in CAMES, meant that if a false graph could project onto a graph representing a subscriber's setup with the new equipment then that setup would be rejected as invalid. As we know from Chapter 2, if any graph is false then automatically so are all its specialisations. We saw that if it is false that Clyde is an elephant, then anything involving Clyde the elephant will also be false. For instance, given that falsity, then 'Clyde is an elephant and works in a circus' must also be false. Hence small false graphs can falsify any larger graph that contains that false small graph as a generalisation. In CAMES small false graphs, called 'schematic rules', are projected onto potentially vast subscriber's resulting configuration graphs. The alternative of checking for just valid configurations would involve potentially enormous projections, as all of the client's graphs would have to project into ranges of true configuration graphs. Therefore CAMES can be seen to be computationally efficient.

To illustrate the above, a Reuters marketing policy might be that "No two products shall supply the same service"\(^9\). This stipulation ensures that a subscriber does not pay for two products that would only supply the same service. The schematic rule reflecting this is:

\[
\begin{align*}
\text{[Service]} & - \\
\quad (\text{support}) & \leftarrow \text{[Product: *a]} \\
\quad (\text{support}) & \leftarrow \text{[Product: *b]}.
\end{align*}
\]

Now given the subscriber graph:

\[^{8}\text{Thus it is somewhat like the expert system 'RI' that configured VAX computers (McDermott, 1982).}\]
\[^{9}\text{The illustration is adapted from Smith (1991).}\]
Chapter 5

[Subscriber: UK000011]-
  (has) -> [MONR+FF: #1]
  (support) -> [Money-Rates: GGMONR]
  (support) -> [Financial-Futures: CGFF],
  (has) -> [FFA: #2]-
  (support) -> [Financial-Futures: CGFF].

and the appropriate hierarchical relationships, the above stipulation would successfully project into the client graph and result in:

[Financial-Futures: CGFF]-
  (support) <- [MONR+FF: #1]
  (support) <- [FFA: #2].

Thus it can be seen that 'FFA: #2' is a superfluous piece of equipment. In Peirce-based form the equivalent would be:

([Service]-
  (support) <- [Product: *a]
  (support) <- [Product: *b]).

giving:

([Financial-Futures: CGFF]-
  (support) <- [MONR+FF: #1]
  (support) <- [FFA: #2]).

and thereby:

([Subscriber: UK000011]-
  (has) -> [MONR+FF: #1]
  (support) -> [Money-Rates: GGMONR]
  (support) -> [Financial-Futures: CGFF],
  (has) -> [FFA: #2]-
  (support) -> [Financial-Futures: CGFF]).

From all the above, it may seem that CAMES appears more attractive than rule-based systems. However, CAMES still does not sufficiently encode the human expert’s knowledge. This is because CAMES assumes that any configuration is true by default. In the complex domain of business strategy this assumption might simply be too rash. Even stating the default as unknown could be expected to offer no help. The strategic management accountant may likely want to distinguish between truth, falsity and unknown. Where CAMES is placed evidences its limitation in the strategic sphere, as it can be seen to be in one of those conveniently well-defined
problem domains criticised in Chapter 1. Added to these significant doubts were proprietorship obstacles in obtaining CAMES, as it was Reuters' own in-house software. In view of all these concerns, it was decided to reject the CAMES expert system.

Surprisingly there was little else that could be considered as suitable, even further afield. However during the course of the above efforts I developed a close working relationship with a fellow researcher, John Heaton, who had enhanced cgp as mentioned above (Heaton, 1989). John was now devising a new cgp as part of his own PhD research in the Computer Studies Department, Loughborough University. That software looked more promising and, because of my close contact with John, there was the opportunity to exercise some influence in respect of that new cgp's design.

5.06 JEHCGP: A More Promising Conceptual Graph Processor

To begin with, the new cgp software still retained the name 'cgp'. It was decided, however, that for this thesis' purposes the new software should be referred as something that distinguished it from the earlier cgp. Hence I gave it the working title 'JEHCGP'. This term was arrived at by simply prefixing cgp with John Heaton's initials.

JEHCGP had automated the computational overload that cgp had suffered from, as discussed above. The user merely told and asked JEHCGP in terms of conceptual graphs. Upon the entry of each graph, JEHCGP evaluated it. Apart from syntactic checks, in essence JEHCGP compared the newly entered graph with those graphs in its knowledge-base. For user-asserted graphs, JEHCGP would report that either the graph was unknown to it or that it thought the opposite was true. In the latter case the knowledge-base had determined that a true graph was false or vice versa. For a user-queried graph JEHCGP would

10 Excepting for any conceptual graphs processor that was being kept completely secret, this state of affairs was confirmed by events at the conceptual graphs workshop user evaluation that appears in the next chapter.

11 To state that the knowledge-base, rather than the inference engine, determined the result may be seen as conflicting with the traditional model of expert systems. This statement, however, is deliberate. It has already been shown that conceptual graphs render the distinction between facts and rules as artificial. Hence, by extension, so can be the distinction
explain why it was true or false. Alternatively if the graph could not be
determined as either true or false, the software would reply that the graph
was simply unknown. All in all, JEHCGP therefore offered solutions to the
issues raised throughout this chapter.

Although the details of JEHCGP remained unpublished for some while, its full
functionality eventually appeared in Heaton (1992). JEHCGP’s various
functions are further discussed, as appropriate, throughout the remainder of
this thesis. For now, JEHCGP’s essential commands can be illustrated by:

!graph. Add graph to the knowledge-base.

?graph. Query the truth of graph in the knowledge-base.

~graph. Retract graph from the knowledge-base. Omitting a graph
retracts the whole knowledge-base\(^{12}\).

<filename. Load a knowledge-base.

>filename. Save the current state of the knowledge-base.

*.. List the current state of the knowledge-base.

^. Quit JEHCGP.

Examples of commands with graphs are:

![central office': Leeds].

!(['central office': *x]; ([office: *x]-
(characteristic)->['higher purchase cost'])).

?['improved service level'].

between the knowledge-base and the inference engine. As far as the user is concerned it is the
very graphs’ structure that causes the reasoning. Thus yet another aspect of computational
overload is made transparent to the user.

\(^{12}\) Subsequently changed to ‘-graph.’.
Note that, in the above, the more convenient ‘(...)' are used in place of ‘−(...)’ to denote negative contexts in the linear form. To see how far JEHCGP would be usable by the domain expert, a pilot study was conducted. The details, which also elaborate on the above, and outcome of that study are discussed next\textsuperscript{13}.

5.07 Events at a Usability Pilot Study Based on JEHCGP

This pilot study examined the usability of conceptual graphs employing JEHCGP, which at that time was implemented on Loughborough University Computer Studies Department's Hewlett-Packard UNIX computer known as 'indigo'. The pilot study was conducted on the two days March 11, 1992 – March 12, 1992. The study's subjects were Susan Heggie, Chris Hinde and Chris Messom.

The subjects were given and asked to understand the paper "Enriching Cognitive Mapping: A Technical Comparison between COPE and Conceptual Graphs based on an 'Office Location' example" (See Appendix A/01) and the introductory instruction sheet "Pilot Study based on the attached 'Office Location' paper" (See Appendix A/02) the day before (on the 11th).

At the start of the study itself (on the 12th), the subjects were given the paper "Conceptual Graphs Pilot Study: Instructions and Tutorial" (See Appendix A/03) and asked to work through it using JEHCGP.

The results were:

1) All the subjects managed to get to grips with the problem and the way it was modelled in conceptual graphs. They were able to both appreciate the

\textsuperscript{13} That discussion is also duplicated in Polovina (1992b).
deficiencies in the COPE, and find ways of improving the existing conceptual graphs knowledge-base using conceptual graphs.

2) However all the subjects expressed severe reservations about the quality of the interface itself. In particular JEHCGP's interface, which could only handle the linear form of conceptual graphs, came in for harsh criticism. Essentially the subjects felt the linear form was too difficult to follow. The subjects agreed the display form of conceptual graphs, in a suitable WIMPS environment, would be a great improvement. They felt that the results would be much more meaningful if this study was re-conducted using the display form throughout instead.

3) One subject criticised the 'UNKNOWN' outcome of certain queries. This arose from the fact that cgp would produce this same outcome whether the graph a) existed in the knowledge-base but could not be proved, or b) did not exist in the knowledge-base at all. For instance, given that neither a local or central office was chosen beforehand, the query '?['improved service level']', which existed but was not provable, would produce the same 'UNKNOWN' answer as the query '?['office site']', which is totally absent from the knowledge-base. The subject felt a decision-maker would be asking 'what do I do now?'. That subject felt a more interactive response than merely 'UNKNOWN' was needed, to include looking for similarly spelt concepts.

Therefore it became evident that, to test properly the usability of conceptual graphs, a graphical interface in an interactive WIMPS environment would be required. One subject suggested that the problem presented for the study was sufficiently trivial that it need not be modelled in conceptual graphs but in prolog alone. However such triviality was necessary to commence the study at a reasonably user-understandable level from which further details could be built. Moreover the ultimate purpose underlying this study is to evaluate if the familiarity of pictorially-based structured diagrams respected by business information professionals such as accountants (See, for example, Sizer 1989 and Woolf 1990) can be combined with the power of conceptual graphs. Overall, the work was therefore worth continuing.

Initially, as John Heaton was also at the above session as a silent observer, he subsequently ameliorated the above arising 'UNKNOWN' problem by
enhancing JEHCGP so that it would explicitly output why the query graph was unknown. The output now showed the graphs that would be needed to prove the query. Hence the above query ‘?{'improved service level’}’ would now not produce the same ‘UNKNOWN’ answer as the query ‘?{'office site’}’.

In the latter case JEHCGP would repeat the graph back to the user, thus stating that it had no knowledge about that graph whatsoever. In the former case it would show graphs for both a local or central office, thus depicting that JEHCGP would need to know about the existence of these offices beforehand.

The other issues about ‘similarly spelt’ concepts, more meaningful examples, and the graphical form were to be examined thoroughly during the remainder of this thesis. In the next section, a methodology to aid significantly in these tasks is proposed.

5.08 The Conceptual Analysis and Review Environment, CARE

Given all the matters raised in this thesis up to this point, a human-computer integrated framework called Conceptual Analysis and Review Environment, or CARE, is proposed as the most suitable means by which conceptual graphs can be employed by strategic management accountants. Through the software-based CARE approach, the strategic management accountant could integrate with a computer knowledge-based system. For the reasons already identified, CARE should be conceptual graph-based. In summary, these reasons were:

a) we identified that strategic management accountants work with a particularly involved highly qualitative problem domain that requires a technically advanced knowledge-based tool.

b) the advanced technical tool specified in ‘a)’ above must also be user-comprehensible. Thus conceptual graphs could be particularly suitable because:

i) business information professionals such as accountants use structured diagrams in the general course of their work
ii) negative contexts happen to be similar to the way the accountant's bookkeeping model negates numeric values by enclosing them within rounded brackets.

It was discovered from the pilot study above that the linear form of conceptual graphs was too abstruse for users. The thereby discovered 'command line' usability limitations exemplified by the linear form augments the reservations of Schneiderman (1987), who asserts the considerable superiority of employing graphical human-computer interfaces instead. The user's dislike of the linear form, as compared to the graphical display form, of conceptual graphs was also identified in the findings of Slagle, Gardiner and Han (1990). Hence CARE should employ conceptual graphs in their graphical form.

In addition, points of the theory itself must be as clear as possible. Ideally, as expressed throughout this thesis, intricate parts of theory should be handled by the machine and be transparent to the domain expert without making the power of any CARE software too trivial. Allied to this, and in line with a study by Reason (1990), the CARE software should prevent the domain expert from constructing graphs that have any incorrect syntax.

Through the very spirit of human-computer integration, CARE itself should also be adaptive in the light of experiences gained from its exposure with the domain expert. The nature of such changes would be determined by whatever occurs during that interactive process. An example might be where a user models, say, that 'A' affects 'B', 'B' affects 'C', 'C' affects 'D', and 'D' affects 'A'. In some cases this may be a valid recursion but in others it may not. Amey (1968) and Mepham (1981) discuss whether circular arguments occur within the management accounting domain of opportunity costing. Such dilemmas might become significant in conceptual graphs. This is because knowledge in conceptual graphs form can be generalised and specialised at many levels, thus rendering a recursion insufficiently obvious. Should this problem arise from user exposure, it may be appropriate to include some kind of user-warning aid.

Further to the single user-computer interaction scenario, CARE should allow integration to occur amongst many domain experts. A typical CARE modelling situation might begin with the initial graphs being drawn by one strategic
management accountant. Subsequently, that expert’s results could then be consolidated with graphs drawn by other such accountants. CARE could be used to check for inconsistencies between the models. This would draw out differing opinions which could then be resolved to obtain a more comprehensive model. Such activity, as it expanded into capturing other problem domains of interest to the relevant organisation, would evolve into that organisation’s corporate knowledge-base. In this regard, CARE need not ultimately be restricted to generalising over strategic problem domains alone. Linking conceptual graphs to databases, spreadsheets, or other knowledge-based systems could be another avenue for CARE. As supported by Parsaye, Chignall, Khoshafian and Wong (1989), such computer-computer integration could significantly elucidate all manner of business problems.

As already stated, CARE was anticipated to be a direct end-user modelling tool without the domain expert requiring to refer matters constantly to the knowledge engineer. In practice, for the user evaluations of the following chapter, it could not be expected that the domain expert should be left completely alone with the CARE software. This is because those experts could not be reasonably asked to supply their valuable time simply to learn about CARE, as well as apply it to some appropriate domain problem. Therefore there would be on hand, for the purposes of the user evaluations at least, a human facilitator to assist in the rapid understanding of CARE by the domain expert. The cognitive mapping approach, as modelled by COPE and discussed in Chapter 3, also employs human facilitators in its practical implementations (Eden, 1991). Given that the facilitator would need to be intimately familiar with CARE, it would be incumbent upon myself to fulfil this role.

Lastly, CARE was about bringing conceptual graphs in their most clear yet candid form to the domain expert. It was therefore decided that facilities such as translating between graphs and natural language would not be added to CARE. After all, CARE was ultimately being devised to test conceptual graph’s usability. That meant finding out about the direct interrelationship between conceptual graphs and strategic management accountants. As already suggested in this thesis, natural language could easily obfuscate this particular desire despite the view that conceptual graphs translate naturally to and from natural language (Sowa, 1991b).
5.09 The CARE Software, and its Design Issues

We now turn to how the above general bases of the CARE methodology are to be applied. As expected, a major part of the CARE methodology is its software implementation. Devising that software would require the above ideas to be more precisely defined. Hence this section not only discusses the technicalities of the program itself, but CARE’s actual design issues too.

The CARE software was written in prolog using ‘LPA MacProlog’, a prolog compiler for the ‘Apple Macintosh’ computer. This compiler was selected because it offered extensive graphical libraries that took advantage of the Macintosh’s acclaimed WIMPS interface. Furthermore CARE was to be designed so that it provided a human-computer interface for JEHCGP, which was also written in prolog. This was because JEHCGP was criticised essentially for its linear graphs interface alone in the pilot study above. Thus JEHCGP would basically be retained to perform all its conceptual graphs theory operations, as this aspect was generally liked by the pilot study’s participants. As JEHCGP was written in prolog too, it was natural to combine it with the prolog of the CARE interface to provide an arrangement that could be expeditiously prototyped and enhanced cyclically.

The CARE interface program is listed in Appendix A/04. As agreed, over-explicit references to confidential JEHCGP code have been edited in Appendix A/04 accordingly. Appendix A/05 imparts a flavour as to the changes that were necessary to JEHCGP itself. As can be seen from Appendix A/05, the CARE interface code was essentially written as a module that was simply linked to JEHCGP. This avoided too many delicate changes being made to JEHCGP itself, thus assuring its integrity remained intact. Appendix A/06 illustrates how the interface, which will be discussed further below, looks. The interface is also shown as part of that discussion. It is not proposed to discuss the program clauses directly, as this low level of detail can

14 NB: Appendix A/04 shows the program code as at version ‘1.3’ rather than as it started, namely version 1.0. Version 1.3 was how the code ended up by the time that all the user evaluations of Chapter 6 were completed.
be gleaned from *Appendix A/04* and *Appendix A/05*. Rather the operation
of the program is described as it affects the user.

The CARE software, meaning the combined interface and JEHCGP engine, is
fired up by mouse double clicking on its icon. As it was being devised as a
user evaluation prototype, the software was not a 'stand alone' program.
Hence the LPA MacProlog application files also needed to be present. Once the
CARE software is loaded, the user is presented with the screen as shown by
Figure 5.02. This figure also appears as Figure A/06.01 in *Appendix A/06*. The

![Figure 5.02: Example CARE screen (version 1.0).](image)

only differences between the start up window and Figure 5.02 are that the
‘offices leads to employees’ window contains no graphs, and that window is
entitled ‘untitled drawing sheet 1’ instead. This shows that CARE currently
contains no conceptual graphs. It is upon such windows, or ‘drawing sheets’

---

15 To provide a reference point, however, the program commences with the clauses under the
‘START’ heading. Each heading denotes a MacProlog program window. Hence ‘START’ depicts
such a window, ‘BROWSING’ and ‘INPUT’ are others. Further details about LPA MacProlog
can be obtained from the programmer’s manuals or ‘Logic Programming Associates’ (LPA)
directly (Tel: 081 871 2016, Email: lpa@cix.compulink.co.uk).
that graphs are drawn by the user. The functionality of CARE is enacted through the 'tool pane' and the 'menu bar'. The drawing sheet contains the tool pane along its left hand side, whilst the menu bar resides at the very top of the screen. In line with the standard WIMPS interface of the Macintosh, the drawing sheet windows can be scrolled and re-sized. The tool pane operates as follows:

**Arrow Tool (shown activated).** The arrow tools functions are as follows. To begin with, the arrow tool selects a conceptual graph drawing by the user dragging a marqui over it as required. The arrow tool also drags a selected drawing by positioning the cursor over the upper left-hand corner of the appropriate concept, relation or outer negative context and then holding down the mouse button. A 'hand' cursor appears together with a rectangular 'outline' of the entire selected drawings to show they are being dragged. To prevent any confusion, graphs cannot be dragged such that they remain on top of other graphs. Hence graphs that are so dragged simply return to their original location\(^{16}\). Such tacit error checking, that is without an explicit message, also feature elsewhere in the CARE software. In these instances, this approach was decided to be the most succinct method of illustrating the error to the user. Above selected drawings can be cut, copied or cleared via the 'Edit' menu. Incomplete graph elements, such as a relation attached to two concepts, cannot be cut or cleared as this would leave a syntactically incorrect graph. This time an error is signalled, as the user would be less certain about what had precisely happened with the drawing. The cut error message is "Sorry, but nothing cut because at least one incomplete structure was selected." The clear message is the same except that the term 'cleared' is substituted for the term 'cut'\(^{17}\).

\(^{16}\) This alternative was decided upon, rather than positioning the graph in the nearest 'free space' for example, because it was the simplest to program without creating conceptual graphs syntax difficulties for the user. This was deemed acceptable as the user evaluations discussed in Chapter 6 could also be relied upon to highlight any deficiency in this respect, and thus require adoption of the appropriate alternative.

\(^{17}\) Again this choice offered the simplest, yet syntactically uncompromising, option in programming terms prior to user evaluation outcomes revealing the contrary. Another
Eraser Tool. This tool erases a drawing with this tool by positioning the cursor over that drawing, in the upper left-hand corner if appropriate, and clicking once. Should an incomplete graph element be erased, its remaining elements are also erased. Again, this prevents incorrect syntax graphs remaining on the drawing sheet.

Concept Edit Tool. Creates a new concept by positioning the cursor on the desired starting point on the drawing pane and, by holding down the mouse button, dragging a marqui to an estimated size of the concept. Then the appropriate text string is typed in. Once complete, a click anywhere outside the concept automatically re-sizes the concept box to fit around the text string neatly. Previously created concepts can be edited by clicking within the upper left-hand of that concept. Any links, such as relation links or negative contexts, attached to that concept are automatically erased. This ensures that, as the edited graph is now essentially a new one, a conscious decision is made by the user as to whether those original links were still valid.

Relation Edit Tool. Performs for relations the same as the concept edit tool above.

Directed Link Tool. Creates a new directed link between concepts and relations by positioning the cursor on the upper left-hand of the first concept and, by holding down the mouse button, dragging a line to the upper alternative, for instance, could have been to cut or clear all complete structures and leave the incomplete ones behind.
left-hand of the relation. Then the mouse button is released and a second line is dragged to the upper left-hand of the second concept. Tacit error checks, in that no link is drawn as a result, prevent these links from being drawn wrongly by the user. For instance, directed links cannot be drawn i) from one concept directly to another without a relation in between; ii) between relations alone.

(...)

Negative Context Tool. Creates a negative context by dragging a marqui over a drawing as required. This procedure is illustrated by Appendix A/06. The context is not drawn if it only partially covers a graph, or if it surrounds an incomplete graph element such as an unattached relation. Thus syntactically incorrect conceptual graphs are avoided, in the tacit manner discussed earlier.

Structure Assert Tool. Asserts a conceptual graph drawing into the knowledge-base of JEHCGP by dragging a marqui over a drawing as required. Like the negative context tool above, a partially covered or incomplete graph is not asserted. Unlike the tacit error handling of the negative context, however, the user receives an error message. The message, “Sorry, but nothing entered because at least one incomplete structure was selected.,” thereby removes any ambiguity in the user’s mind with respect to what may or may not have entered into the knowledge-base.

The text string of attempted concept or relation assertions is also checked for syntax errors. Basically this involves existence of too many ‘colons’ in that string. For example, JEHCGP could not determine where the type label ended and the referent begins in the concept:

Management Accountant: Strategic: S. Fred

Hence, on attempting to assert this graph, the error message “Sorry, but nothing entered - a concept had more than one colon (:)” would occur. Similarly, as the accepted form of conceptual graphs does not support type
labels and referents in relations, the error message "Sorry, but nothing entered – a relation incorporated a colon (:)!" is generated. Thus the user is not erroneously lead to believe that relations can be so structured.

When asserted graphs successfully enter JEHCGP, the user is presented with the screen as shown by Figure 5.03. This figure also appears as Figure A/06.02 in Appendix A/06. This figure shows that JEHCGP was unaware of this graph, so it adds the graph to its knowledge-base. Alternatively, JEHCGP could have responded that it knew the graph already. This would happen when JEHCGP contained the graph already or could derive it by inference. If JEHCGP determined that the opposite was true, it would state this accordingly. In all these other cases, the new graph would not be added to the knowledge-base.

Figure 5.03: Output from asserting example graphs into knowledge-base.

Note that the output window, 'cgp - status report', retains the linear form of conceptual graphs. Whereas the user could simply layout his or her graph drawings to taste and then input them into JEHCGP, the programming in replying in the display form was seen to be very complex. Furthermore, LPA MacProlog would not permit graphical objects and text to appear in the same window. As a result there would be a real danger that CARE would have to output too many windows on the screen for the user to comprehend.
Therefore it was decided to keep JEHCGP's linear based output, although it was evidenced earlier that the graphical display form was preferable. The effect of this decision could expect to be confirmed or denied by events at the subsequent user evaluations in Chapter 6, and the appropriate modifications made to CARE accordingly.

The way generic referent's are handled should also be noted from the JEHCGP output, in that JEHCGP changes these "<whatever>" referents into '*' prefixed sequential numbers. This is to prevent the erroneous interpretation of such referents, as explained in Chapter 2. In essence, any graphs:

a) having the same generic referent in more than one concept and 

b) input simultaneously from the same drawing sheet

are determined to be coreferent. JEHCGP therefore gives them the same sequential number. Otherwise they are given different numeric generic referent markers signifying they are not coreferent. The user is thereby freed from having to worry about the indistinct nature of generic referents. That user would simply need to be aware that suitably marked coreferent concepts are input simultaneously from the same drawing sheet. Those graphs that were not so input would be regarded as non-coreferent even if they happened to share the same generic referent marker18.

A double click on the 'I' tool beforehand prevents the graph from being evaluated until the old knowledge is overridden in the 'CGP' menu by the menu command 'Override old knowledge...'. This matter is discussed later.

Structure Query Tool. Queries a graph in the knowledge-base by dragging a marqui over a drawing as required. The actions are similar to the 'I' tool above, excepting that the query graphs, by definition, are not added to the knowledge-base. As discussed earlier, JEHCGP responds with an explanation showing the graph to be true, false or unknown. If there is more than one

18 The practical difficulties of generic referents are not peculiar to conceptual graphs theory, as evidenced by Pilote (1989).
explanation, the user is prompted with a 'yes/no' dialog box requesting whether another explanation is required. Clicking on 'yes' displays the other explanation until all the explanations are exhausted or 'no' is selected. Double clicking on the tool lists the entire knowledge-base.

Structure Retract Tool. Retracts a graph in the knowledge-base by dragging a marqui over a drawing as required. The actions follow the pattern of the '1' and '?' commands above. The graph is not actually retracted until the old knowledge is overridden by 'override old knowledge...' in the 'CGP' menu as discussed below. Double clicking on this tool retracts immediately all that is held in the entire knowledge-base, leaving it empty.

Graphic Information Tool. Shows the internal prolog-based form of a drawing. This was of relevance more to CARE's program maintenance than domain expert use.

The 'mini viewing pane' at the bottom left hand corner of the drawing sheet enables the user to move the visible part of the drawing sheet more quickly than through using the scroll bars. The scroll bars use effectively cause the Macintosh to redraw the graphs continually, and thus slow down the visible area movement significantly.

We now move onto discussing CARE's menu items, other than where they a) do not differ from the usual Macintosh menu functions, b) are essentially irrelevant LPA MacProlog menu functions, or c) have been mentioned already. Each menu item also supported a standard Macintosh 'N' 'hot key' equivalent in line with good human-computer interaction practice. The menu items are as follows:

Under the 'File' menu:

Defaults....: The 'Graphics' option should be in 'MacProlog' format for graphs to be properly pasted to another drawing sheet.
quit: Quits CARE. The user is asked if he or she wishes to save the current state of the drawing sheets and knowledge-base. If yes, the 'Save the current state...' command, under the 'CARE' menu and discussed below, is enacted before the program is quitted.

Under the 'Edit' menu:

Paste: When pasting a conceptual graph drawing where the front drawing sheet already contains a graph, a new untitled drawing sheet is automatically generated to take the pasted drawing. Thus the user is discouraged from drawing unnecessarily large graphs. This procedure is meant to ensure the user-readability of the graphs, in accordance with Miller's seminal study which showed that a human can generally handle only a maximum of between five and nine concepts in his or her mind at any one time (Miller 1956).

Balance: In a linear output graph, by positioning the cursor within that structure and enacting this function, the area between the graph's matching brackets will be highlighted so enabling their scope to be seen more easily. The 'Balance' command was included to help overcome any user difficulty with the residual linear element in CARE, and the command's effect is expected to be determined from the user evaluation sessions. Appendix B/13 illustrates the 'Balance' command in use from the final user evaluation session discussed in Chapter 6.

Under the 'Window' menu:

Select Window...: Allows a particular window to be selected.

Under the 'CARE' menu:\n
19 As a tidying up exercise, some of these menu items were subsequently rearranged either within the same menu bar heading or moved to under the 'C6P' menu. Appendix A/04 shows their final position. There should be no difficulty in finding the appropriate code under the 'MENU HANDLING' program window of that appendix.
About CARE...: This displays details about the CARE software, such as copyright, trademarks, version number and acknowledgements.

New drawing sheet: Creates a new empty drawing sheet, giving it the title "untitled drawing sheet <n>" where 'n' signifies a sequential but previously unused number to distinguish it from other drawing sheet windows. Each drawing sheet contains its own tool bar. Clicking in the small white box at the top left hand corner of the drawing sheet causes the drawing sheet to disappear, but it is merely hidden. Thus it and any graphs it contains can be made visible through the 'Select Window...' command under 'Window' above. To completely remove a drawing sheet and its contents, 'Kill front drawing sheet' is used below.

Rename front drawing sheet...: Renames the title of a drawing sheet window. For instance, the window 'untitled drawing sheet 1' could be renamed 'offices leads to employees'. A renamed drawing sheet name that coincides with an existing drawing sheet name is suffixed by a 'e', to avoid window conflict problems. The user thus has the ability to name drawing sheets in familiar phrases.

Kill front drawing sheet: The front drawing sheet, and any graphs it contains, are completely deleted.

Open a previous state...: This opens a previously saved set of drawing sheets and knowledge-base, and also clears the prevailing knowledge-base. Any existing drawing sheets are, however, left untouched. Clearing the present knowledge-base and replacing it with the file-saved one was a feature of JEHCGP. It was decided that this should also occur with CARE, as it would prevent the user from being confused as to what lay in the knowledge-base. Like 'Rename front drawing sheet' above, window conflicts are avoided by the suffix 'e'.

Save the current state...: This saves the current state of the drawing sheets, hidden and visible, and knowledge base, according to a file name and location essentially of the user's choosing. Both this command and 'Open a previous state...' above operate via standard Macintosh file open/save dialogues.
Find a text string...: Searches the output of `cgp - status report` and then the
drawing sheets for a user defined text string, but does not search the internal
knowledge-base. Therefore, if required, the knowledge-base needs to be listed
onto the output beforehand by double clicking on the '?' tool above. For the
output part, any existence of the string is highlighted from which the user can
stop or go on to look for another instance. In the drawing sheets the string's
concept or relation appears selected, together with a 'yes/no' dialog box. The
user can continue the search by choosing 'yes' or stop it by clicking 'no'.
Should the string not be eventually found, the user is informed that "<String>
could not be found. Sorry!".

MacProlog menus: Shows, by being disabled ('greyed out') itself, that all LPA
MacProlog menu items not relevant to the operation of CARE are disabled.

Under the 'CGP' menu:

Override old knowledge...: Each previously unknown structure added to, or
retracted from, the knowledge-base is recorded as episode of knowledge.
Unless a retraction, or a structure entered immediately after double clicking
on the 'I' tool, the structure is also evaluated. If the newly evaluated structure
does not make the knowledge-base inconsistent then that structure is also
added to the knowledge-base. Otherwise it remains an episode only. However
the knowledge-base may be rebuilt to override its previous knowledge in
favour of the later contradictory knowledge, together with retracted and
double clicked 'I' items, by enacting this menu item.

Lattice...: Replaces the JEHCGP command line equivalent for entering lattice
relationships directly. Note that in JEHCGP, a lattice not only refers to the type
hierarchy but a hierarchy of relations too. Although not in Sowa (1984), the
addition of a relation hierarchy in conceptual graphs theory has met with
general acceptance. Hence there was no valid reason why it should be
excluded from CARE. JEHCGP, however, displays these lattice relationships in a
somewhat non-standard way. Namely, they are displayed as:

Subtype << Supertype. (Or Sub-relation << Super-relation.)

rather than:
Chapter 5

Subtype < Supertype.

This was so for reasons inherent to the functioning of JEHCGP. Nonetheless, it was decided that the change was so trivial that there was no need to rectify this in CARE. Another interesting feature of JEHCGP was that hierarchical relationships could be conceptual graphs too. Thus:

Management Accountant << Accountant.

could be stated as:

[type:Management Accountant]<-(subtype)<-[type:Accountant].

or

[type:Management Accountant]->(supertype)->[type:Accountant].

The well argued theoretical basis for this approach can be seen in section 2.9 of Heaton (1992). Without repeating the technical issues here, it was decided that this would be a natural way for the domain expert to deal with hierarchical relationships. This was because the above step would incorporate the hierarchy into the remit of conceptual graphs themselves, and so further reduce the computational overload on that user. Although this was an extension to the accepted conceptual graphs theory, it was worthy of inclusion in CARE.

Type definitions were entered into JEHCGP by means of the 'double implication' described in Chapter 2. Relation definitions are also entered in a similar way. For example the graphs:

![Diagrams]

also describe that:
central office << office.

These alternative methods of describing hierarchical relationships could be entered into JEHCGP via the 'I', queried by '?', or retracted by the 'crossed-out I' tool commands. Thus the user computationally overloading nature of the 'Lattice...' command was effectively rendered as redundant.

Conformity...: Replaces the JEHCGP command line equivalent for conformity relationships. As JEHCGP extracted these conformities in the course of its conceptual graph evaluations anyway, this command became redundant.

Execute a linear form script...: Replaces the JEHCGP command '%<file>.'. As this item read in linear form graphs, this facility would not be anticipated to form part of the user evaluations of Chapter 6. Hence it is not discussed further. The nature of this command is discussed in sections 2.8.2 and 3.4 of Heaton (1992).

Engage Interactives: Replaces the JEHCGP command '%user.'. This command was not anticipated to be employed in the user evaluations. Hence, like 'Execute a linear form script...', the command is not discussed further. The command is discussed in section 4.3 of Heaton (1992).

Command line...: This facility executes the remaining JEHCGP commands in their native command line form. These commands, which are essentially not relevant to this thesis, can be seen in Heaton (1992).

Lastly, in all the above functionality and design issues of the CARE software, attention was given towards the criteria of Ravden and Johnson (1989). In particular, their study of practically evaluating human-computer interfaces identified the following nine top level attributes:

- visual clarity
- consistency
- compatibility
- informative feedback
- explicitness
- appropriate functionality
- flexibility and control
- error prevention and correction
• user guidance and support

As stated by Sage (1991), the definition and meaning of these attributes can be inferred from the attributes themselves. Therefore it should be evident that CARE has primarily met the above criteria. The actual extent of conceptual graphs' achievement, through CARE, along these attributes can expect to be tested in the user evaluations of the following chapter.

5.10 Concluding Remarks

The critical limitations of transferring knowledge between the human domain expert and the computer are well known. Added to this obstacle, the strategic problem domain is known to be particularly complex. Conceptual graphs offer a potential means of tackling this impasse. This is because conceptual graphs are an advanced knowledge-based technique yet contain acute similarities with existing methods, namely structured diagrams and bookkeeping negation, respected by the accountant. From this user-familiar base, CARE, which stands for conceptual analysis and review environment, presents conceptual graphs in a user-friendly methodology for strategic management accountants. Furthermore the advanced knowledge-based decision support software tool that embodies CARE incorporates sound human-computer integration principles. Thus, through CARE, these experts might be able to employ conceptual graphs in handling their intricate problem domain.

At this point, it is appropriate to confirm all the findings discussed in this thesis so far through user evaluations. This is the subject of the next chapter, that now follows. That chapter evaluates conceptual graphs amongst domain experts who would be adequately representative of the strategic management accountant.
Chapter 6

The Usability of Conceptual Graphs by Strategic Management Accountants

6.01 Introduction

As established in Chapter 5, CARE would be the favoured vehicle through which conceptual graphs should be evaluated amongst human domain experts who were adequately representative of the strategic management accountant. This chapter is about these evaluations, starting out with subjects from the conceptual graphs community itself. The evaluations progress to key business school experts, and culminate in a session with senior practising accountants. That last session is augmented by statistical analyses. In addition, CARE is enhanced iteratively in accordance with the results of each evaluation session. The outcome and implications from each of the above sessions are discussed extensively.

6.02 Responses to CARE at a Conceptual Graphs Workshop

This survey aimed to determine to what degree CARE was acceptable to the conceptual graph specialists who essentially comprised the attendees at the annual workshops on conceptual graphs. Clearly if CARE was too difficult for this group to follow then strategic management accountants could not thereby be expected to understand CARE. In such a case the survey would show that a radical re-thinking of the whole approach would be needed before proceeding any further. With the above in mind the following experiment was performed.

On July 10, 1992 at the Seventh Annual Workshop on Conceptual Graphs, New Mexico State University, Las Cruces, New Mexico, USA the software that embodies the CARE methodology was demonstrated to the workshop’s attendees. Those attendees are listed in Appendix B/01. The demonstration occurred between 8.30 a.m. and 9.30 a.m., and was followed by the presentation of Polovina (1992c) later that morning. Example screens of the CARE software itself appeared in Appendix A/06. These were shown at the demonstration and as an overhead projector slide at the presentation. Each attendee was given a copy of two ‘quick references’ depicting the means through which the CARE software is operated. The quick references are
reproduced in Appendices B/02 and B/03. Also on hand was the latest guide to JEHCGP\textsuperscript{1}. Furthermore the attendees were given the questionnaire reproduced in Appendix B/04. However, because of the many time consuming activities at the workshop, they were not actually expected to complete the questionnaire. Instead their oral comments, which the questionnaire may have helped to focus, were recorded on audio cassette tape whilst the demonstration was being conducted.

At the demonstration, the most striking result was the very favourable response CARE received. It soon became clear that the participants at the workshop had seen nothing like CARE, let alone their accepting CARE as a valid interpretation of conceptual graphs theory. Many attendees were extremely keen on obtaining a copy of the software. The statement of John Sowa, himself at the workshop, aptly illustrated the interest by stating that he would like to take a copy to show others that there does exist an actually usable application of his theory. All this confirmed the necessity of research to be performed on conceptual graph's usability dimension, as mentioned above.

A further encouraging sign came about from a discussion with another key participant who typified a minority that were sceptical about the accepted Peirce-based method of inference in conceptual graphs theory. Like those others, this attendee favoured the more traditional if-then style of inference instead. He therefore wondered why CARE took the Peirce logic route. Rather than get into the severe limitations of rule-based systems, it was simply explained to the participant that this complexity could be less apparent to accountants. The explanation clarified the strong similarity between Peirce logic and the way accountants negated figures by surrounding them in brackets. Despite his scepticism, the attendee agreed with the merit of Peirce logic in CARE.

There were also particular comments about the software worthy of note:

1) The output of what was to the attendees an endless amount of continually scrolling linear form graphs, lattice interrelationships, conformities and

\textsuperscript{1} Unpublished at the time, but an updated version later became available as a Loughborough University Department of Computer Studies Technical Report (Heaton 1992).
explanations from elementary graph operations caused a sizable critical response from the participants. The output occurred essentially whenever graphs were entered into the knowledge-base. One participant suggested why not have an option that hides the output window completely, but most felt that condensing the output would be an initial improvement. The participants were shown that the output window could be enlarged, or a smaller sized output font selected. Nevertheless the comments revealed that if conceptual graph experts had difficulty keeping up with the output information then strategic management accountants would probably find it impossible. Therefore not only the quantity but also the quality of the output requires further careful thought.

2) There were questions as to what was the largest size of graphs that could be drawn in each drawing sheet window. It was pointed out that, although virtually any size graph could be drawn, there were subtle devices within the software that encouraged users to draw smaller graphs using more windows rather than larger ones in less. One example is that graphs may only be pasted into an empty drawing sheet. These devices are meant to ensure the user-readability of the graphs, in accordance with Miller's seminal study which showed that a human can generally handle only a maximum of between five and nine concepts in his or her mind at any one time (Miller 1956). The participants concurred with this approach.

3) One attendee was concerned about the removal of negative contexts or directed links when such a constrained concept or relation was edited. The participant thought this could be a potentially tedious process. It was explained that this was a deliberate step intended to ensure that, as the edited graph is now essentially a new one, a conscious decision should be made as to whether the constraints were still valid. This would be ensured by the user actually having to redraw them. The participants were shown how easy the redrawing process was. Nonetheless the attendee's comment was to recur throughout the demonstration, together with support from some others. In the end these participants felt that some kind of user prompt giving the choice of retaining the constraint was better than automatic removal.
4) Although it would not be added to CARE for the reasons given in Chapter 5, the ability to translate graphs into natural language statements was brought up as a property that could be added to the software.

5) One key participant, who was actively involved in business applications of conceptual graphs, saw the CARE software as a valuable computer aided software engineering - commonly referred to as CASE - tool. What interested him was the level of browsing capability the software had. In response, the participant was shown the software’s text string search facility. The participant appreciated this approach but commented that for large-scale applications a more sophisticated method was needed. A second key participant then stated that conceptual graphs themselves could be used to provide this enhanced sophistication, by means of the hierarchical interrelationships within both concepts and relations. The second participant proposed an interface in which the user could mouse-click on a concept or relation and a further window could appear showing the definition of that clicked graph element. This could continue through many levels. The first key participant responded by arguing, however, that this process would be inadequate for dynamic interactive knowledge management. Considering this argument, it may be possible that the gap could be met through Peirce logic. Hence this avenue should be explored further in any attempt to improve the browsing capability.

6.03 Comments on Workshop Responses

It was somewhat surprising that CARE managed to excite the attendees at the workshop to the extent it did. One main expectation, for example, was that there was already in existence a similarly usable tool. The interest underlines the ardent desirability for examining conceptual graphs’ overlooked usability side. The eager response at the workshop justified CARE, as it would find users now in the form of the workshop participants. However subsequent evaluations would entail taking CARE, after considering the improvements outlined above, to where it was actually targeted. This was namely amongst representative domain experts in strategic business knowledge. These experts, unlike the workshop attendees, could not be expected to be conceptual graphs specialists.
6.04 The Way Forward from the Workshop Responses

After the firm support from the demonstration at the conceptual graphs workshop, it was decided to take CARE directly to a domain expert in strategic decision making. Nonetheless it was also decided that the domain specialist should be one who is in an academic environment. This was because such a person would be able to comment, from his or her academic knowledge, on any further improvements that CARE might require before being exposed in any industrial situation. Hence the next participant would be best selected from a 'business school' type of institution.

The participants would be given an individual demonstration. The reason behind conducting the problem solving exercise as such was to ensure that the expert would have the chance to develop his or her own perspectives about the whole exercise independently from the premature influence of others. To provide a further empirical basis, the subject would be asked to produce a problem of entirely his or her own choosing. The idea is that the expert presents a problem well known by him or her to be particularly difficult to analyse by existing methods. Then I would attempt to help, in my separate role as the 'facilitator' described in the last chapter, the expert solve the problem using the CARE software. As with the conceptual graph workshop session, these sessions were to be recorded on audio cassette tape. The sessions thus resembles the protocol analysis style of technique discussed in the last chapter, but with the added emphasis on evaluating CARE itself than purely a knowledge elicitation exercise.

In the event, three key subjects as described were selected. The subjects were from Loughborough Business School, Loughborough University. Each kindly agreed to take part. They were a) Malcolm King, Professor; b) Ruth King, Lecturer; c) Paul Finlay, Professor. To preserve some degree of anonymity, the business school participants are from now on referred to simply as 'Expert 1' to 'Expert 3'. For the same reason, the masculine also refers to the feminine domain expert as appropriate. The actual studies commenced with an individual demonstration of CARE with the first chosen subject, 'Expert 1', who is an authority on the particularly relevant area of business strategic problem solving. Details about the sample problem are given in Appendix B/05. The problem was one that was of particular interest to Expert 1. Again in my aforesaid separate role as facilitator, I had been given this problem the
week before. This approach enabled the facilitator to attempt the kind of conceptual graphs models that could help structure the problem. The facilitator would then relate these graphs to Expert 1 at the session for his comments, criticisms and other ideas. The resulting initial graphs appear in Appendix B/06. Chris Hinde, Senior Lecturer, Department of Computer Studies, Loughborough University was also in attendance at the morning session.

The study took place on September 14, 1992 in two three-quarter hour sessions at 10 a.m. and 3 p.m. The significance of organising the sessions thus should become evident later. Modifications were made to the CARE software prior to this study. These changes were introduced in the light of the workshop survey and are discussed first.

6.05 Enhancing CARE from Version 1.0 to 1.1

In line with the more pressing comments arising out of the conceptual graphs workshop demonstration, enhancements were made to the CARE software. The modified software was given the version number '1.1', an incremental upgrade from '1.0', to distinguish it from the workshop version. The action taken was as follows:

a) The attendees at the workshop had criticised the endless amount of continually scrolling linear form graphs, lattice interrelationships, conformities and explanations from elementary graph operations in the output. The comments revealed that if conceptual graph experts had difficulty keeping up with all this technical output information then strategic management accountants would probably find it impossible. Hence an 'easy output' option was added that, when engaged, essentially suppressed the substantial detailed technical output whilst retaining a sufficient content for the user to follow the happenings within the knowledge-base2. Therefore the easy option made CARE more flexible by enabling the output to be followed more easily, through its reduction to user-manageable proportions, yet allowing the full technical output to be chosen if required for some reason.

2 The easy option was enacted by default. To toggle between the easy and technical output the user would select "Command line..." in the 'CARE' menu and type the letter 'u'.

- 139
b) The text string search facility was improved so that the user could now select where the search should be directed. Previously the user had to go throughout the whole output prior to examining the drawing sheets. The user now had the choice of looking directly in the drawing sheets, output or the knowledge-base itself. The latter location took advantage of an enhancement made to JEHCGP by its author, John Heaton, also from Loughborough University's Computer Studies department. This improvement was viewed as an intermediate step in addressing CARE's browsing capability highlighted at the workshop until perhaps a more adequate solution can be found.

c) An option to either remove or retain negative contexts or directed links when a concept was edited was also introduced. This appeared as 'Always remove encumbrances' in the 'CARE' menu. Although it was an easy task to realign the links automatically, the same could not be said of the contexts as it was possible that a realigned context could overlap another graph element. It was felt that the little untidy-ness of a context mis-fitting around its changed concept or relation was less of a disadvantage than the complex programming needed, with all its inherent dangers, to shift graph elements around into some suitable form.

Further to the above changes, the opportunity was taken to fix various bugs in the workshop version. The new version also incorporated the latest version of JEHCGP. We now turn to the session with Expert 1.

6.06 Responses to CARE by Expert 1, Session 1, a.m.

Using the CARE software Expert 1 was shown the conceptual graphs in Appendix B/06, together with an attempt to explain these graphs to him. However it became evident right from the start that Expert 1 could not follow these structures. Expert 1 had difficulty accepting that contexts in conceptual graphs terms was consistent with the negating brackets in the accountant's bookkeeping model. Various scenarios, using the aforesaid graphs, was explained to Expert 1 demonstrating the similarity. Expert 1 also did not like the use of non-user familiar terms such as 'deiteration', and was not even happy with substituting terms like 'removal' or 'rubbing out' in its place. Moreover Expert 1 was far more used to 'if-then' rules and constantly wished
to relate the graphs to such rules. Again Expert 1 was shown how one graph could consist of many ‘if-then’ rules and furthermore vividly show other logical relationships such as ‘and’, ‘or’, ‘not’, and ‘not consequent then not antecedent (modus tollens)’. In the end it was becoming clear that the session should be abandoned with the view of determining how to proceed, if at all. Expert 1, in fact, suggested suspending the session and he would return in the afternoon instead.

6.07 Preparing the Ground for the p.m. Session

Once Expert 1 had left, a discussion was had with Chris Hinde about the plan for the afternoon session. Chris Hinde had sat in on the morning session, as mentioned above. We agreed that the fundamental obstacle lay in touching base with ‘if-then’ rules. Once Expert 1 could see this link then he might be able to see the extra dimensions that conceptual graphs reveal. Hence it was decided that the menu item ‘Forms’ should be temporarily added to the CARE software. This menu item would contain the following:

‘If A then B’,
‘If C and D then E’,
‘If F and not G then H’,
‘If not I then J’,
‘If K then not L’.

Each item would call up the appropriate pro forma graph. Although there was insufficient time by the afternoon session to include the menu item fully, it was possible to draw each pro forma graph on individually titled drawing sheets, save them, and then call up the whole file. ‘Forms’ was added to the menu bar to show how they might be called but the links to each ‘if-then’ item was not implemented. Nonetheless enough was completed for now to get the point across. The pro forma graphs appear in Appendix B/07.

It was also suggested that another temporary menu item ‘Concepts’ should be added to CARE. The reasoning behind this item was to enable users to examine what concepts and relations were held in the software. The point was made that users should be prevented from as much keyboard input as possible due to the risk of, say, misspelling or using the wrong letter typing case. Even though CARE supported the text search facility mentioned above, it was felt that there should be also be a more ‘intelligent’ concept and relation...
retrieval mechanism. This echoed comments about browsing made at the conceptual graphs workshop. Hence ‘concepts’ was added to CARE’s menu bar and contained the following as ideas:

‘By order of recency...’,
‘Alphabetical order...’,
‘Hierarchical order...’.

The first item would order concepts and relations under the last time they were used with the more recent the higher in the list. The second is by alphabetical order and the third by relationships in the type and relational lattices. The ‘concepts’ menu item could not be made active by the time of the afternoon session. Its purpose was to draw out Expert 1’s thoughts about such a facility by its mere presence.

The basis of all the above was to help the user determine where to start with the software, to decide what to do next with it and not reject it out of hand. It was these obstacles that may well have been the reason why the morning session went so badly, as may be affirmed or denied by the afternoon session.

6.08 Responses to CARE by Expert 1, Session 1, p.m.

Expert 1 was shown the pro forma graphs, and each graph was discussed with the author. The same points were raised as in the morning session. This time, however, Expert 1 grasped the points that could not successfully be brought across in the earlier session. Furthermore Expert 1 became keen on returning back to his actual problem graphs which he could now follow competently.

As for the ‘concepts’ item, in the event it did not arise in the afternoon session. This matter could have been brought up, but it was anticipated that such a move would likely have thrown off course the constructive flow of this afternoon session. Nonetheless this item should not be overlooked. Hence this item, now with time to consider its more proper implementation, should be correspondingly introduced at some later session.

Now that the pro forma graphs had helped Expert 1 ‘get started’, and thereby vindicate somewhat the matters raised above on this point, he began to ask

---

3 This even happened to the extent that in one case where I had thought a graph was incorrect, it was in fact shown by Expert 1 to be satisfactory!
some more involved questions about conceptual graphs themselves. The salient questions were:

1) Do the conceptual graphs represent the problem more concisely than ‘if-then’ rules? Although he could appreciate the many dimensions that one conceptual graph reveals would require many explicit ‘if-then’ rules, it remained in his view that guideline ‘a’ in Appendix B/05 still required many tricky looking graphs. From the graphs modelling this guideline in Appendix B/06, it was difficult to disagree with this comment.

2) How adaptable are the graphs in the light of further knowledge? Referring again to guideline ‘a’, he asked what would be the effect of adding, say, ‘Assistant Lecturer’ to the model? Like ‘if-then’ rules, it was clear this would involve changing all the graphs. Again, this obstacle was particularly acute in guideline ‘a’.

After the above session was completed it was decided to see if the above concerns could be ameliorated for another session with Expert 1.

6.09 Preparing the Ground for Another Session with Expert 1

As suggested above, the graphs in Appendix B/06 that related to guideline ‘a’ in Appendix B/05 were given further careful thought. A way was found to model the guideline that appeared to overcome the concerns raised by Expert 1 above. These new graphs are shown in Appendix B/08, which also includes a comprehensive explanation of how these graphs are read. In addition, it was decided to show Expert 1 the essence of how many lines of ‘if-then’ rules would be needed to appreciate the inherent dimensions within guideline ‘a’, whereas the same dimensions could be captured in far fewer conceptual graphs. The voluminous if-then rules, given in Appendix B/09, would thus be shown to Expert 1 at the next session. Also some modifications were made to the CARE software, and are mentioned first.

6.10 Enhancing CARE from Version 1.1 to 1.1a

The menu ‘Forms’ was replaced by the menu item “Draw ‘If-Then’ forms”, which was added to the ‘CARE’ menu. This fully enacted the pro forma ‘if-then’ style graphs that were found to be so useful from the last session. On
the user selecting this item CARE would load, one drawing sheet per graph, the graphs shown in Appendix B/07 without affecting the state of the knowledge-base itself. Each drawing window would be titled appropriately, for example “If A then B”. The user would thereby have the pro forma graphs on hand to refer to as required yet leaving all else intact.

Also added was the item ‘Draw without’ to the ‘Edit’ menu. After further thought on the experiences with Expert 1, it became evident that a more succinct method of illustrating the effect of deiteration or double negation was needed than always having to resort to the JEHCGP engine. ‘Draw without’ achieved this by opening a new untitled front drawing window and placing within it the graph from the previous front drawing sheet. The difference would be that any properly user-selected graph elements in that previous drawing sheet would be excluded. Remaining negative contexts would be re-drawn correctly sized around whatever remained within that context. In addition there is an option to leave double negative contexts intact, rather than allowing them to double negate to true. For example, given a front drawing sheet containing just the following graph:

![Graph](image)

in which the user had selected the concept ‘[p: *x]’4, enacting ‘Draw without’ would create a new drawing window entitled ‘untitled drawing sheet <number>’ and place one of the following graphs within it:

![Graph](image) or ![Graph](image)

(allowing double negative contexts)

Clearly the above was less principled than employing JEHCGP, for instance not showing the effect of coreferent passing, but was considered to be a better quick and convenient indicator. The term ‘Draw without’ was chosen

---

4 Which would be shown highlighted as such in the CARE software.
deliberately to signify this indicative purpose rather than a completely sound occurrence of deiteration or double negation.

As these were practically small changes to CARE the version number was incremented to 1.1a rather than 1.2 at this stage. As before, bug fixes to CARE and the latest version of JEHCGP were incorporated. The responses of Expert 1 at the second session now follow.

6.11 Responses to CARE by Expert 1, Session 2

The second session with Expert 1 occurred between 10:00 and 10:45 a.m. on September 27, 1992. The session was audio tape recorded as before. Using CARE, Expert 1 was shown the items in Appendix B/08. Expert 1 also saw Appendix B/09, showing the great amount of if-then rules required to highlight the equivalent views of the same problem. Expert 1 made the following initial points at this second session:

- Smartened the representation up a lot. Fewer and clearer graphs than before.

- Liked the way the system could get to know about seniority in general.

- Can now understand that a negated graph states that "it is true that the graph inside the negative context is false".

- Appreciate how conceptual graphs are more succinct than if-then rules. Furthermore, because of the potentially greater number of ways a system could output an if-then based explanation than conceptual graphs, agree that there is less risk of CARE replying in a sequence that the user might not expect.

- Avoid conceptual graphs' technical terms like 'instantiate' or 'project'. Try the term 'specific case' instead. Similarly 'remove from consideration' is most likely to be preferential to 'deiterate'. There is a danger of over-assuming that the user would have an unconscious knowledge about conceptual graphs' technical terms. It is unlikely that the domain expert would remember such terms, as such terms ill-accord with that expert's own background knowledge. Better to try and use terms that the
strategic management accountant would be familiar with already. Thus an ideal user reaction might be "I can relate to CARE, like the spreadsheet, in a natural way".

- Further to the last point it might be best to elucidate 'removal from consideration' by "any graph within a negative context that has its specific case existing outside that context can be removed from further consideration".

Expert 1 subsequently stated that he had moved away from expert systems because they were too weak, as they could not properly handle declarative logic. He explained that in strategic planning it is difficult to know what are objectives and what are constraints. Expert 1's statement demonstrated the inappropriate nature of traditional expert systems for strategic problem solving, as already discussed in this thesis. In this regard, it was explained to Expert 1 that the nature of conceptual graphs also freed the user from worrying about having to make the 'artificial' distinction as to whether a knowledge element is a fact or a rule. As for the 'flowchart' and 'bracket' comparisons, Expert 1 stated the following:

**Flowcharts.** Expert 1 doubted that accountants really used them in practice. It was explained that flowcharts formed part of an accountant's basic training, but Expert 1 remained sceptical. Expert 1 warned that practice does not necessarily support the text-book advantages. This dislike on the part of practising accountants would thereby mean that these professionals' familiarity with flowcharting is actually insufficient to make the link with conceptual graphs.

**Brackets.** Again Expert 1 was not convinced. He saw that '(£50)' for instance meant no more to the accountant than '£50'. The idea that the accountant would see the underlying conceptual meaning of 'not £50' was too ambitious to make the appropriate link with Peirce's negative contexts in CARE.

Expert 1 was shown the 'Concepts' menu that was overlooked in the previous session, but gave no striking reaction about this matter other than vague
agreement\textsuperscript{5}. As expected, this was because the other issues were turning out to be of more prior importance.

More fundamental than all the above however was that, although the conceptual graphs in \textit{Appendix B/08} had improved upon the representation of Expert 1's problem, doubt remained as to their ultimate viability as the strategic management accountant's knowledge-based decision support tool. None of the above comments by Expert 1 could disguise the reality that he was not comfortable with the intricacy of conceptual graphs, even as presented in CARE. This was exacerbated by the fact that even the enhanced graphs could not be claimed to have captured more than the trivial elements in the problem. Expert 1 raised this very point. For instance the graphs of \textit{Appendix B/08} made no attempt to rank the funding request priorities according to some time or monetary scale. What, for example, would be the priority be if only a Probationer and a Reader each made a funding request, but the Probationer had done so \textit{after} the Reader? What if a Probationer had made ten requests, should he or she be allowed them all when perhaps it was more equitable that some of the limited money budgeted should go to a more senior staff member?

In the conceptual graphs models in \textit{Appendix B/08}, all such items were conveniently labelled under the broad label 'Other factor', but this would clearly need to be analysed further if Expert 1's problem was to be usefully modelled in CARE. \textit{Appendix B/10} shows how this may be started. Nonetheless in practice the conceptual graphs knowledge-base would face considerable difficulties, as it could only determine that no other factors existed by default. Hence it would have to keep searching its entire contents, and perform every avenue of reasoning, to discover anything that may crop up as an other factor. As discussed by Russell and Wefald (1991), the problem of default reasoning can lead to exponential computational times. Given the increasing size of the conceptual graphs model required to pursue Expert 1's problem realistically, there would soon be much to reason through. Thus CARE's interactive stipulation laid down in Chapter 5 becomes denied, as the user would lose the benefit of real-time responses from the software\textsuperscript{6}.

\textsuperscript{5} This was not recorded on tape, which had just run out at this point.

\textsuperscript{6} Tellingly, the accepted form of conceptual graphs theory contains no provision to prevent the exponential computation time problem generally. This was evidenced by the increasing time to
We could accept that such situations were technically solvable, as Russell and Wefald suggest, through some intelligent search algorithm. However, given Expert 1's aversion with the present graphs model, the even more intricate graphs detailing the other factors could only intensify that aversion. It was therefore becoming apparent that the complexity of conceptual graphs was overwhelming the problem at too trivial a level.

6.12 Comments on the Sessions with Expert 1

Given that Expert 1 was effectively a newcomer to conceptual graphs, the above demonstrations of conceptual graphs under the CARE methodology with that expert indicated that the conceptual graphs technique could be expert user-understandable. This was remarkable given the relatively short space of time that the Expert 1 sessions took, especially after the obstacles encountered at their start. Expert 1 did, however, have reservations about the similarity between conceptual graphs and a) the accountant's flowchart, and b) bracketing figures to negate their value in the accountant's bookkeeping model. He questioned the viability of drawing out this comparison to 'real-world' accountants, who he anticipated would not be as au fait with 'a)' and 'b)' to see the link with conceptual graphs. This attacked the fundamental basis of the CARE methodology itself, and thereby needed to be carefully monitored in future sessions accordingly.

The decision to let the user select a problem of entirely his or her own choosing was also seen to be the correct approach. This randomness had inherently brought out dimensions that a pre-defined problem could have easily overlooked. In particular, the apparently simple relationships that make up guideline 'a', Appendix B/05, of the user's problem really tested a potential weakness of conceptual graphs. It was revealed from the second session with Expert 1 that this aspect might be overcome as a conceptual graph knowledge-base is continually built up thus producing more generally applicable structures. As shown by Appendix B/08, such a process in its course can yield more succinct and adaptable structures but does not necessarily

---

process other than the most basic graphs in JEHCGP. Figure B/08.10 (Appendix B/08) in fact had reached the level of complexity that could not be evaluated by the software at all.
appear to make the hidden intricacies of the problem itself sufficiently user-understandable.

Therefore the study with Expert 1 reveals that conceptual graphs might be understood by strategic management accountants, yet the session also reveals a caveat. The caveat is that merely understanding conceptual graphs does not guarantee the expected result that those graphs highlight additional dimensions of a strategic problem to the domain expert. In fact the conceptual graphs appear to add complexity to the problem even when that problem is at a very trivial level. This fear of adding computational overload was mentioned in the early parts of Chapter 2 in respect of ‘if-then’ rules. This burden may turn out to be relevant to conceptual graphs as well, and signifies the area for particular attention in the further evaluation sessions with different human domain expert participants.

6.13 Enhancing CARE from Version 1.1a to 1.2

Although perhaps by now of merely incidental value, the following measures were added to improve the browsing capability of CARE should this key issue still arise in any future session:

a) When a new graph enters the knowledge-base, any new type labels, relation labels and individual referents were now compared with those already known to knowledge-base. All new items are reported to the user on the output window. Hence the user would be made aware of such as i) input graphs that contain spelling mistakes, ii) wrong letter typing case choice, or iii) inappropriately worded items. The item is output as:

   New type or relation labels:

   <list of items>.

   and has already been illustrated in this thesis (For instance see Appendix B/08).

b) From the earlier conceptual graphs workshop session, “Draw browsing forms” was added to the ‘CARE’ menu. This contained a series of query graphs, one per suitably titled drawing sheet, that can be used to intelligently interrogate what was in the knowledge-base. Thus conceptual
graphs themselves were being employed as a browsing mechanism in accord with the discussion raised at the conceptual graphs workshop. The layout is similar to that of its sister menu item "Draw 'If-Then' forms". Examples of where the graphs could be used included i) finding the hierarchical relationships between type or relation labels, ii) show true concepts or relations, or iii) show false concepts or relations. The actual graphs appear in Appendix B/11.

To reflect the reasonable significance of the above additions, the CARE software version number was incremented to '1.2'. Again bug fixes to CARE, and the latest JEHCGP, were incorporated.

6.14 The Way Forward from the Sessions with Expert 1

In response to the issues raised at the Expert 1 sessions, it was proposed that CARE should now perhaps be introduced from a simpler viewpoint, based on the idea that a knowledge-base could consist of merely a set of FALSE circumstances. The user builds up what he or she believes to be a set of unacceptable graphs, and then queries the knowledge-base as before. This time, however, instead the knowledge-base's UNKNOWN response could be treated as a statement that believes the graph is TRUE in the absence of any reason why the graph is false. To check for invalid queries the system, as mentioned in the above enhancements for version 1.2, now gives a warning when it encounters a type/relation label or individual referent in a query graph that it has never heard of. A FALSE answer shows that a true query graph matches with a graph given as false in the knowledge-base. This process is similar to the refutation strategy employed illustrated by the conceptual graphs-based expert system CAMES discussed in Chapter S. By this route the strategic management accountant would encounter much less difficulty in reviewing the conceptual graphs modelling his or her knowledge.

Although considerably restricting in power as a consequence, the above approach would still retain the following features of conceptual graphs:

a) The hierarchical object-oriented nature of conceptual graphs.

b) If a graph is false, then so are any of its specialisations. This should also i) make the user aware of the consequences of stating falsity, and ii) make for
a more humanly comprehensible, through more compact sized, knowledge-base.7

c) The essence of negative contexts.

Ultimately conceptual graphs would still be employed by the domain expert in their full capacity to be truly effective. This need has been emphasised throughout this thesis. Through initially introducing conceptual graphs to the domain expert on the above simplified basis, it was hoped that the difficulties in following the full form could be overcome as the domain expert's understanding increased.

The new subject, 'Expert 2', was both a domain expert in his own right and had conducted extensive research into the viability of accountants using intelligent knowledge-based systems directly. Thus his comments on CARE would be invaluable.

As with Expert 1 it was decided that Expert 2 should take a problem entirely of his own choosing. Expert 2 chose his department's method of selecting acceptable applicants for their Master of Business Administration (MBA) degree course. Although he had documented a solution to this problem (Finlay and King, 1989), Expert 2 stated the situation remained unsatisfactory. The intricacies of the shortcomings were intimately familiar to Expert 2. The randomness advantage that this problem selection method assured was again secured, as with the Expert 1 sessions. In the session now, however, no facilitator activities of modelling any part of the user-chosen problem beforehand occurred. Any facilitator activity would happen solely at the session itself. Hence the domain expert might identify more with conceptual graphs than shown at the previous session, if modelled in front of that expert from the very beginning.

6.15 Responses to CARE by Expert 2

The actual session was carried out for approximately 1 hour from 5.30 pm on Friday, October 9, 1992. Like before it was tape recorded. Expert 2's paraphrased

7 With a quicker response time too.
Chapter 6

comments in italics, together with the appropriate graphs and remarks, occurred as follows:

Let's look at the interaction between an applicant's age and his/her eligibility for the MBA course. In response, the following graph was drawn:

The term 'char' is short for 'characteristic'. The above graph was followed by the 'query':

The above query was made to the knowledge-base and, due to the previous graph that any applicant over 50 was ineligible for the course, was shown to be FALSE. It was clarified to Expert 2 that no matter how many other graph elements or other specialisations were attached to the above query, such a graph would still include the above pattern and thereby be identified as false. The next query was:
Here, as expected, the knowledge-base came back with UNKNOWN. Based on the earlier explained initial approach, the knowledge-base knew of no reason why the above graph was false. Hence it could be accepted as true by default. *All understood.*

Given that Expert 2 comprehended the above, the original ‘full’ conceptual graphs reasoning method was attempted. To aid events, the previously discussed ‘if-then’ pro forma “If A then B” was first shown and described to the participant beforehand. This was followed by the scenario:

```
<table>
<thead>
<tr>
<th>Applicant:*x</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
</tr>
<tr>
<td>Age:=&gt;50</td>
</tr>
</tbody>
</table>

| Ineligible Applicant:*x |
```
to give, after "x" is instantiated to 'F Bloggs':

**Ineligible Applicant: F Bloggs**

Hence, in addition to true results by default, including graphs such as the 'if-then' style one above would also show TRUE from a precise set of circumstances. Expert 2's understanding, however, stumbled when the above scenario was presented. 'Double negation' was followed satisfactorily, but 'deiteration' was not. The terms "The inside matching one can now go", and "General case in the inside matches with the specific case outside, so the inside one can go" were tried. Where do they go? The terms 'higher level' and 'lower level' were also tried, whereby higher level graphs mean their lower level counterpart can be removed out of consideration completely, were also tried. What is higher level and what is lower level? Referring to the computer screen, Expert 2 was shown that the higher level, the less 'rings' (negative contexts) the graph was bounded by. An unbound graph was at level '0' whilst a graph inside a boundary, which was itself inside a boundary was at level '2' and so on. Hence the higher the level number the lower and vice versa. By judicious use of what turned out to be the very helpful 'Draw without' facility in the 'Edit' menu added since version 1.1a to drive the ideas home, Expert 2 finally understood.

Furthermore, the subject was shown how the above 'if-then' style graph vividly revealed modus tollens, an aspect not clear from 'if-then' rules: i.e. given that a consequent is false, then so are all its antecedents. In the above if **[Ineligible Applicant: F Smith]** was in fact false, then it and its negative context could be removed from the above mentioned graph leaving:
Hence 'F Smith' cannot be over 50. Understood.

It was also demonstrated that [Ineligible Applicant] is the same as stating that [Eligible Applicant] is false. Therefore, by substituting the latter term for the former in the above rule, this gives:

After the double negation cancels out, the graph becomes:
This reveals a further insight of the interrelationships between problem elements not evident from 'if-then' rules. Understood.

The session then moved on to discuss the next scenario from which the following graph was drawn:

![Graph 1]

An applicant who is aged between 35 and 50 and who lives outside of the East Midlands is ineligible. 'Non EM' depicts a non-East Midlands location. Understood.

The graph:

![Graph 2]

would be UNKNOWN, because it is more general than the previous false combination for 35 to 50 year olds. Therefore should the default of true be accepted by the user as before, F Smith would be wrongly given the benefit of the doubt here! It was also explained to Expert 2 that it would be wrong to have the 'too general' graph:
because such a graph would falsify all applicants between 35 and 50 no matter where they lived. By this point the limitations of the refutation approach should be apparent. Understood.

Expert 2 then asked about introducing a hierarchy: There are three types of location, namely 'Close', 'Reasonable' and 'Far away'. An applicant who is aged less than 35 and lives 'Far away' is ineligible. An applicant who is aged between 35 and 50 and who lives 'Reasonable' or 'Far away' is ineligible. 'Reasonable' and 'Far away' may be seen as subtypes of 'Non EM', which the earlier false graph depicts. Imagining that the user found out that F Smith lived in a 'Reasonable' Lincoln, the answer to the following graph would be FALSE:

Without any further prompting, Expert 2 correctly saw the 'AND' relationship in the earlier false '35 to 50 age' graph. Expert 2's discovery was underlined by drawing the following graph, which conveyed the same meaning:
Understood.

To recap on and consolidate the subject's understanding, the 'If-then' forms were re-loaded and more thoroughly explained. This facility was introduced in the light of the previous Expert 1 sessions. As with Expert 1, it was explained how the visual forms revealed more than procedural 'if-then' rules such as modus tollens, NOT, and AND aspects already mentioned above.

Lastly, to indicate the limits of how far conceptual graphs theory could be reasonably understood by the strategic management accountant, Expert 2 was shown the following graph:

It was explained that, apart from reading the above as 'If not I then J' or 'If not J then I', the graph could be read as 'I or J'. This is because if one, say I, is true it can be removed to give:

Anything that shares the same context with an 'empty' negative context, known as the 'empty clause', can be removed. This is because logically the truth of any context which contains the empty clause is always false. The result is:
The double negations cancel out to simply ‘true’. The purpose of this procedure was shown to Expert 2 by the following example graph:

As well as other possibilities, the above can be read as ‘If I or J then K’. Hence should either ‘I’ or ‘J’ be true, then by the above explained operations ‘K’ would be asserted as a true fact by being released into the outermost context. Now say instead that ‘I’ was false. This would leave:

This further illustrated the extra insights not evident from if-then rules in that, irrespective of whatever an item’s OR value may be, should that item be true then the corresponding consequent is asserted as true. Here, ‘K’ would be asserted from ‘J’ irrespective of whether ‘I’ is TRUE, FALSE or UNKNOWN. Not fully understood.

We now move onto Expert 2's comments about CARE. To begin with, even the most expert system-astute domain expert, like Expert 2, was not expected to understand the last-discussed complicated ‘OR’ scenario fully in a single session. Nonetheless the session revealed that Expert 2 was following the conceptual graphs theory satisfactorily. Through this understanding he commented that conceptual graphs were too intricate to warrant the management accountant modelling strategic knowledge this way. This would even be the case despite conceptual graph's technical flexibility over if-then rules.
Expert 2 was not convinced by the argument that such an overhead was to be seen by the user as a much more accurate picture of the actual dimensions of qualitative problems. His reason struck more fundamentally at the CARE methodology itself: *Are you saying that CARE is better at solving problems because it uses a technique, conceptual graphs, that is better than the way human experts think? It seems to me that CARE is a normative model of problem solving, in which case you are on very dangerous ground. The overwhelming evidence is that this too-prescriptive approach of artificial intelligence is clearly erroneous.*

6.16 Comments on the Session with Expert 2

Examining the events at the Expert 2 session, the similar issues that occurred in the Expert 1 also arose here. Again there were the peripheral issues, like using the correct terminology. There were also the more fundamental: Expert 2's concern that the user needed to understand the conceptual graph operations would in effect represent an unwarranted computational overload on the strategic management accountant. Furthermore, Expert 2's potentially too-prescriptive nature of CARE statement meant that conceptual graphs were inherently moulding CARE into a role that directly contravened the basic intent of the CARE methodology.

Expert 1's criticisms echoed those of Expert 2. Both experts were ably qualified to represent the strategic management accountant in the user evaluations. These similarly critical outcomes from two independent experts depicted a worrying reality. It would seem that conceptual graphs might *not*, after all, be of use to the strategic management accountant.

6.17 The Way Forward from the Session with Expert 2

Given the above outcome it was decided to start determining the extent of the criticisms of the last two domain experts. Each of these experts had particular previous experience on the matters discussed throughout this thesis so far. The views of a mainstream accountant unexposed to the aforesaid debate might thereby be useful by placing the outcome from the first two experts in perspective.
With the above in mind 'Expert 3', who fitted the above requirements, was approached. Expert 3 is a fully qualified Chartered Accountant, and has an extensive knowledge of current accounting issues. His main interests lie in the financial accounting domain. His comments about CARE, as a appropriately independent domain expert would thereby be invaluable.

Given Expert 3 had no previous exposure to expert systems, it was decided that this session should concentrate on CARE itself than in tackling a domain problem. However, to ensure Expert 3 was not left completely untutored, he was provided copies of Polovina (1991-1992a, 1992c) for him to glance through the week before the session. As Expert 3 became more aware about CARE from the session, he would be asked to suggest freely what sorts of problems he envisaged where CARE could be employed. It was originally intended that such a problem might then be covered in a further session, but it turned out that Expert 3's comments made this subsequent event unnecessary.

6.18 Responses to CARE by Expert 3

The actual session itself was carried out for approximately three-quarters of one hour from 2:00 pm on Thursday, October 22, 1992. The session was tape recorded. Expert 3's paraphrased comments in italics, together with appropriate diagrams and remarks, occurred as follows.

As with the Expert 1 and Expert 2 participants, Expert 3 was shown the 'if-then' forms following the same explanation strategy as with those previous sessions. The author even discussed the effect of the 'empty clause' first brought up with Expert 2. Expert 3, somewhat surprisingly, did not express any in difficulty following throughout: A single picture can encapsulate more than one rule. The phrase "can be removed from consideration", which came about from the previous sessions, seemed more meaningful to Expert 3's understanding than the term 'deiteration'. Again the facility "Draw without" appeared to help correspondingly.
The following simple illustration was drawn for Expert 3. The graph considers the hypothetical definition of a small company:

“A Company which has a turnover of less than £1,000,000 is a Small Company”. Given:

The result is:

A realistic instance may be, say, when the user is interested in the investment of a portfolio of companies but is only interested in those that are environmentally friendly. Under traditional accounting models companies would be analysed according to the greatest return on investment irrespective

---

8 The term ‘\( \lt \)’ in the graph stands for ‘less than’. Although not shown, there is also ‘\( \gt \)’, which signifies ‘more than’. Note the referent also contains the ‘\( \geq \)’ symbol. All these items reflect an enhancement made by John Heaton to JEHCGP by the time of this session. The enhancement echoes the measure referent (See Sowa, 1984, pages 89-90). Although excluded from this thesis on the grounds noted in Chapter 2, the measure referent as implemented in JEHCGP appeared sufficiently rigorous to be made use of here. The illustration reveals the nature of this implementation.
of company background, whereas through CARE the qualitative dimensions that make up environmental aspects could also be introduced into the model. 

Appreciated.

Expert 3 wondered who the target users were: *Do you expect accountants in general to have CARE on their desks, or only those with specialist Information Technology skills? I anticipate that there may be resistance to the use of such a tool by some accountants, because most do not fall into the latter category. Even electronic spreadsheets are not used by a few accountants simply because they are too unfamiliar with computers.*

The flow diagram in Figure 6.01, regarding the *accounting of contingencies* arising from the accounting profession's Statement of Standard Accounting Practice (SSAP) 18, was produced at the session by Expert 3. *How do conceptual graphs compare with such a diagram? In response it was emphasised that, with diagrams like that in Figure 6.01, it was basically possible only to start at the beginning and follow down the routes given. This was because of that form of diagram's procedural nature. It would be very hard to start at an element somewhere in the middle and appreciate all its repercussions. The above kinds of diagram thereby suffer from the danger that they may well obscure salient aspects that should be addressed. Appreciated.*

Through the very act of producing the above diagram, Expert 3 supported the familiarity accountants did have with structured diagrams. Expert 3 furthermore agreed with the reasoning that CARE methodology exploited the accountant’s knowledge of structured diagrams to introduce conceptual graphs. However Expert 3 saw the problems similar to Expert 1, in making the comparison with bracketing: *I think that accountants just see the debits and credits of the bookkeeping model as merely something they learnt to use, without always understanding that it applies an underlying conceptual meaning of a sacrifice versus a gain. This is probably because an accountant's*
training is so intense that there is insufficient time to ponder over such academic aspects. I do not specifically teach the 'sacrifice versus gain' aspect on the undergraduate courses or the MBA course.

Lastly, the priority funding request example of Expert 1 was explained to Expert 3. Appreciated.

In his comments, it seemed that Expert 3 was favourable to CARE: I can see the benefit of this technique, from what has been explained, as a more productive way of dealing with some problems than flowcharts. Although there was some discussion about CARE's possible use in auditing, Expert 3 felt that CARE would be best employed in tackling fundamental decision-making areas rather than the day-to-day solutions characterised by financial accounting. This meant strategic decision-making, the potential domain of management accountancy. As an example Expert 3 brought up the concept of 'materiality' in accounting. This is where a decision has to be made about whether an item affecting a business is material to that organisation. If the matter is unimportant, the item in question can then be ignored. Otherwise that item must be encompassed by the business according to the extent of its significance. This fundamental assessment can easily be very subjective, yet must be determined before there can be any consideration of, say, treatment under SSAP regulations.

6.19 Comments on the Session with Expert 3

From Expert 3's comments, CARE was properly targeted to address strategic management accounting domain problems rather than the other, more regulated, areas such as his financial accounting domain. Drawing the similarity with structured diagrams was also sound. Although it was sensible theoretically to do so, the bookkeeping bracketing-to-negate similarity could not be fully exploited in practice. Nonetheless the overall validity of the devising the CARE methodology to the point that it was worth testing with domain experts, a matter brought into question from the Expert 1 and 2 sessions, was re-confirmed by the session with Expert 3.

The real obstacle, namely the lack of domain problem feedback from the conceptual graphs themselves to the users, still remained. In the previous section the term 'appreciated', rather than 'understood', was stated in
paraphrasing Expert 3's statements. This was because Expert 3 was really agreeing with the philosophy behind devising the CARE methodology rather than the benefits of the domain expert actually employing conceptual graphs. His above statement about the accountant also needing a higher level of computer education to follow CARE somewhat reflected this view. Given his sphere of experience was established as being inappropriate to CARE, there would be little point in a further session with Expert 3. Nonetheless he backed the CARE methodology itself, a highly important result.

6.20 Enhancing CARE from Version 1.2 to 1.3

Further to the measures added to improve the browsing capability of CARE version 1.2, the software now included a new menu entitled 'Labels'. This menu embodied the ideas of the earlier 'Concepts' menu, since removed. 'Labels' is a dynamic menu that showed the user what type labels, relation labels, or individual referents existed within the knowledge-base at the time the menu is selected. In addition, if applicable, selecting any one of these constituents caused that string to be automatically pasted into any 'open-for-edit' concept or relation on a drawing sheet. As a result, apart from being aware of the current labels in the knowledge-base, users no longer needed to re-type any existing labels when drawing a new graph. Hence the occurrence of mis-keyed labels was hopefully obviated. Although the browsing matter had become peripheral from the user evaluations so far, there was no reason to suggest it would remain so. Together with bug fixes and latest included JEHCGP, the CARE software consequently became version '1.3'.

6.21 The Way Forward from the Session with Expert 3

As the overall situation stood from the user evaluations up to this point, there could be little doubt that there existed serious limitations about the value of conceptual graphs to strategic management accountants. This was despite the sound basis for hypothesizing they would be of great worth. On further thought, it became apparent that the results of evaluating CARE any further amongst domain experts within academia was by now essentially exhausted. To adequately determine if conceptual graphs turned out as too flawed in the above respect, CARE would need to be shown to accountants in
actual practice. Obtaining such persons even to give their time and effort for no proper financial reward would in itself be a challenge.

Meanwhile it was decided to abandon the 'refutation' approach, started at the session with Expert 2, and return to the original one. This was because that approach had an insufficient effect in aiding the domain expert's understanding, as any practising accountant's time was expected to be too precious. The refutation approach would only take up valuable session time that would be better employed drawing out the more essential issues that were being raised.

6.22 Taking CARE to Senior Accountants at Melton

It was becoming clear that the only effective way to get CARE into a practising accountant's environment was for the facilitator to be employed in such an environment directly. Therefore, from November 1992 to March 1993 I accepted employment as a Temporary Project Accountant at a Local Government Authority in Leicestershire, UK. That authority was Melton Borough Council, where an excellent working relationship was developed with all the authority's most senior accounting staff. Hence a rare opportunity was successfully obtained to take CARE into a 'real-world' environment and conduct a user evaluation session amongst senior practising accountants.

During the above period of employment at Melton Borough Council, this thesis was discussed extensively in general terms with the council's senior accounting staff. These accountants, six in number, expressed an interest in the research, and were pleased to participate in the session now discussed.

6.23 Background to the Melton Participants

Melton Borough Council is one of nine district parts in Leicestershire of the two-tier district and county system of the 'Shire' Local Authorities in

9 It should be mentioned that there was also the practical matter of personal finances: My living allowance from the Science and Engineering Research Council (SERC), supporting my research that lead to this thesis report, had expired.
England. At the county level is Leicestershire County Council. The district councils are essentially responsible for environmental health, refuse collection, street cleaning, leisure and sports facilities, planning, and housing. Melton is also responsible for the Cattle Market held in the town under the Melton Mowbray Cattle Market Act 1869. The county authority is essentially responsible for education, county-wide environmental regulation, refuse disposal, libraries and museums, social services, county-wide planning and transportation, police and fire services. Revenue for the services at both district and county level arrives from central government, council services such as rents from council houses, a levy on local businesses, and a levy on the local population.

In the financial year 1991-2, Melton Borough Council spent £4,612 (thousands) net on services and their share of community charge receipts was £3,046 (thousands). As regards their level of computerisation, Melton are replacing the bulk of their existing computer systems, which are provided by Leicestershire County Council, with an in-house system based on a Sequent parallel UNIX minicomputer running ORACLE software. Included in the software is a new financial package, called ‘ORACLE Financials’, tailored to local authorities.

Apart from the old system’s severe restrictions that prevented even some of the most basic accounting information to be provided adequately, the move to the new platform was precipitated by Melton’s strategy to become a single-tier authority. This situation had arisen because it was the policy of the central government to reduce the levels of local government. Whatever its basis, this decision had created inter-tier conflict between the district and county authorities. They were now in potential competition to provide all the local authority services as either many district councils or one county-wide council. As these events have fundamental ramifications on local government finance, the key local authority accountants, known as ‘Borough Treasurers’, have been propelled to the forefront of local government strategy.

10 There are also numerous parish councils within Leicestershire. These can be ignored for the purposes of this report.
11 In fact I was primarily involved in transferring the council’s accounting system to the new computerised platform.
Leicestershire County Council sought to absorb the district councils into their organisation and create one new unitary county authority. Some district councils in Leicestershire, such as Charnwood Borough Council, had expressed an interest in this route. The bulk, like Melton, rejected this approach. Instead they sought the complete abolition of the county authority.

The new highly expandable computer system at Melton was anticipated to fulfil its part in a new comprehensive Melton Council. Given the many non-trivial qualitative considerations inherently provoked by this strategy, the senior accountants at Melton were ready to consider a strategic knowledge-based decision support tool.

Figure 6.02 shows the organisational structure of the senior accounting staff in the Borough Treasurer’s Department, Melton Borough Council. It is these staff who participated in this session. The Borough Treasurer is the most senior member who, with the Deputy Borough Treasurer, are involved in helping the councillors determine the council’s strategy. As shown in
Figure 6.02, both the Borough Treasurer, John Flower, and his Deputy, Keith Aubrey, are fully qualified accountants. Similarly the Management Information Officer, Alan Bromley, is a fully qualified accountant. Alan, who is also highly computer literate, specifies the information technology requirement for the council to perform its functions. The Principal Accountant, Philip Hand, runs the accounts section of the treasury and is a qualified senior accounting technician. He is assisted by the Senior Accountant, Robert Child, and the Senior Accountancy Assistant, Maz Hussain. Robert is an accountant 'qualified-by-experience', that is without formal qualifications but knowledgeable nonetheless through many years of experience in accountancy. Maz is in the final stages of full accountancy qualification.

6.24 Revised Format for Melton Session

The format of the experiment at Melton Borough Council was to be different in some respects from the previous domain expert user evaluation sessions. To begin with, it was decided to conduct one evaluation with all the participants together rather than in a series of individual sessions. Although this choice might be seen to compromise each participant's independence, the single joint session was decided upon because:

a) The extra time that would be required from the participants to conduct the individual sessions could not reasonably be requested from Melton Borough Council.

b) Even if individual sessions could be conducted it would be practically impossible to guarantee that those subjects exposed to CARE would not talk about their experiences with those other participants awaiting their turn at a CARE session.

c) By combining the subject's minds through the single session it was anticipated that the session's events might progress to a more advanced stage.

Despite the Melton subjects' general awareness, to show explicitly the nature of conceptual graphs the subjects were each given a copy of Polovina (1991–1992a, 1992c) one week beforehand. The Melton participants were asked to try
and follow as much of these articles as possible, with a view to at least getting their gist. At the beginning of the session the subjects were each given a questionnaire, which they were asked to complete and return by the next day. That questionnaire appears in Appendix 6/12. This time limit was set intentionally so that the subjects would not forget their reactions at the session. Like all the previous sessions, the event was recorded on audio cassette tape to capture their comments. The questionnaire was designed in a form where the subjects graded their agreement numerically with a series of twenty-four statements, some of which contained sub-statements making thirty-four statements in total. Thus, the questionnaire provided an avenue for statistical analysis as well. John Heaton, JEHCP’s program author, was also in attendance both as an observer and to assist, within his arena, on any technical difficulties that may arise during the session. In line with the previous sessions, I would act during the session as facilitator in addition to my separate role as observer reporting the session’s events.

As part of the kind agreement on behalf of the participants, their names were kept anonymous in regard to the details of the actual session itself that now follows.

6.25 Responses to CARE at Melton Session

The session itself was carried out for approximately 2 hours from 12:00 pm on Thursday, February 18, 1993. The six experts were given the letters ‘A’ to ‘F’ respectively to preserve the confidentiality mentioned above. The following graph was presented to the participants:

![Graph](image)

This example describes that if a customer did not pay the Value Added Tax (VAT) element on his or her bill, then that customer would be a bad
The above simple instance was chosen because the accounts section at Melton had come across a case of an old-age pensioner who felt that he had given enough to his country in the last war. This eccentric customer thereby felt he should not be asked to pay any further 'taxes'. By selecting this somewhat amusing scenario, it was planned to provide a gentle introduction into the session.

Employing the above graph, it was explained in general terms how the above could be part of the encoded knowledge about the council's credit control policy. Each customer would have a 'conceptual graphs' profile that would match with that conceptual graphs modelled policy and, though the appropriate inferencing, reveal that customer's credit-worthiness. However the participants were also asked not to be concerned with at this point how the first shown graph was to be read, or how 'bad customer' was inferred. This was because it was felt that, as augmented by experiences from the previous academic domain expert sessions, the subjects were likely to become too confused too soon. Rather the Peirce inferencing would be dealt with separately through the "'A' to 'Z' concepts" within the "Draw 'If-Then' forms" menu item in CARE. The above graph was to make the subjects more comfortable with a familiar depiction than just an abstract "'A' to 'Z'".

From the 'If-Then' forms the subjects were shown the graph:

```
A B
```

This was followed by explaining how 'B' was derived as true when 'A' is true. Thus, given:

```
A
```

causes the same concept in the 'if-then' graph to be 'removed from consideration' giving:

```
B
```
and after the double negation the result is:

\[ B \]

Sensing that the audience was beginning to lose the significance of all the above apparent effort to execute a simple ‘if-then’ situation, the facilitator stated "One of the questions you may be asking is why go to all this trouble just to do this [above]?" Poignantly the subjects indicated their agreement.

To alleviate this concern the additional features of Peirce’s negative contexts were explained. It was demonstrated how Peirce’s negative contexts performed more than one role. Firstly they negated facts in the same manner as bracketing figures in bookkeeping. Secondly they also can be used in forming both ‘if-then’ and more complex rules. Thirdly they can double negate as appropriate. Fourthly, and most importantly, the negative contexts highlight the relationships amongst the other three features. This highlighting thereby yields further insights not otherwise evident from traditional models. It was stressed that one key aspect of this was the illumination of the closer relationship between facts and rules than the domain expert may be aware of. For instance the ‘if-then’ graph above could also be read as the fact “it is false that ‘A’ and ‘not B’ can co-exist”. Another aspect was the domain expert need not decide arbitrarily between which to use of AND, OR, EITHER...OR, IF AND ONLY IF, or NOT.

To underpin the above arguments the subjects were guided onto its simplest instance, namely a ‘modus tollens’\(^{12}\) illustration. Hence given the graph:

\[ B \]

---

\(^{12}\) As mentioned before modus tollens occurs when some fact is false, and that fact is also a consequent of one or more implications. Consequently that fact’s antecedents in all of those implications become false. The following example was quoted from Polovina (1991a) to the participants: “Credit is refused if a customer is bankrupt, or had a poor credit rating from analysis of their accounts, or so on. Hence [modus tollens shows that] a customer who has credit is in none of these categories".
causes its counterpart in the 'if-then' rule to be removed from consideration giving:

![Diagram](image)

From the above demonstration it was hoped that the participants would appreciate the advantage of the visuality in Peirce logic. Here it was that the Peirce negative contexts showed clearly the 'modus tollens' aspect of implication not evident from the common 'if-then' rule. The participants did not concur. They were finding the negative contexts too troublesome. In view of this state of affairs, it was decided that the best thing to do was to abandon these seemingly too abstract examples and try instead to get the same points across with the first shown 'VAT' example.

The participants were thereby next shown the graph:

![Graph](image)

Together with the first shown 'rule' graph the above graph was asserted, using the CARE '!" tool bar command, into the knowledge-base thus:

<reconstruction of the CARE output window 'cgp - status report'>

Command started at 12:57:17 pm.

New type or relation labels:

- Bad Customer.
- source.
- 'Keeps-the-VAT' Payment.
- Customer.

lattice relation added:

- 'Keeps-the-VAT' Payment \(<\) universal_type.
lattice relation added:
   Customer << universal_type.

lattice relation added:
   Bad Customer << universal_type.

lattice relation added:
   source << universal_rel.

Thinking...

Thinking...

I did not know that:
{
   [Bad Customer:*0]
};
['Keeps-the-VAT' Payment:*1]
   (source)->[Customer:*0]
).

Tidying up...

Command completed at 12:57:26 pm.

Command started at 12:58:05 pm.

New referents:

   Mr. Warwound.

Thinking...

Thinking...

I did not know that:
   ['Keeps-the-VAT' Payment:*2]
   (source)->[Customer:Mr. Warwound].

Tidying up...

conformity relation added:
   Customer :: Mr. Warwound.

conformity relation added:
   'Keeps-the-VAT' Payment :: *2.

Command completed at 12:58:21 pm.
Chapter 6

The knowledge-base was then given the question, using the CARE '?' tool bar command:

**Bad Customer: Mr. Warwound**

giving:

Command started at 12:59:13 pm.

Thinking...

Thinking...

The statement:  
[Bad Customer: Mr. Warwound].

is:

TRUE because:
0. proof of:
   [Bad Customer: Mr. Warwound];
0. is:

1. deduction:
   1. known rule:
      {
         ['Keeps-the-VAT' Payment:*1]-
          (source)->[Customer:*0];
          
         [Bad Customer:*0]
      }

1. so:
   {
      [Bad Customer: Mr. Warwound]
   };
   ['Keeps-the-VAT' Payment:*1]-
    (source)->[Customer: Mr. Warwound]
   )

1. proving antecedent:
   ['Keeps-the-VAT' Payment:*1]-
    (source)->[Customer: Mr. Warwound];

1. proof of:
   ['Keeps-the-VAT' Payment:*1]-
    (source)->[Customer: Mr. Warwound];

1. is:
Chapter 6

2. fact:
   ['Keeps-the-VAT' Payment:*2] -
   (source) -> [Customer: Mr. Warwound];

0. result:
   [Bad Customer: Mr. Warwound]

more? Yes
Time when questioned: 12:59:22 pm.
Time when answered: 12:59:26 pm.
No (more) solutions

Command completed at 12:59:27 pm.

Next the knowledge-base was given the question, again using the CARE 'i?' tool bar command:

![Bad Customer: Mr. Warwound]

resulting in¹³:

<extract>

The statement:

   (   [Bad Customer: Mr. Warwound]

   )

is:

FALSE because:

<same explanation as above showing the opposite:

   i.e. that '[Bad Customer: Mr. Warwound]' is true>

During the course of the above it was explained how the variable '*' referents became defined to a specific item - here 'Mr. Warwound' - , and how that referent was passed accordingly to other concepts through the relevant coreferent link. Also explained was how the above 'linear form' compared to the graphical form. In the latter regard the scope of the '( . . . )' linear style of

¹³ In fact a software bug within JEHCGP caused the statement 'UNKNOWN because ...' to be displayed rather than 'FALSE because: '. After checking with John Heaton, it was explained to the participants that this was a software problem. A manual description was then given about what the correct output should have been, as shown in this report.
negative contexts using CARE’s ‘Balance’ command under the ‘Edit’ menu was shown (See Figures ‘01 and ‘02 in Appendix B/13). The subjects remained unconvinced.

To alleviate each participant’s concern, the session was directed immediately onto more complex scenarios to see if the subjects would accept they were just experiencing difficulties that might be overcome by further training. From these more intricate cases they might agree the benefit of the additional insight into qualitative problems would outweigh the perceived high cost of learning conceptual graphs.

With the above in mind, the subjects were now shown:

The above showed the relationship between good and bad customers. For instance, through the already explained inference operations in conceptual graphs:

would lead to:

and vice versa, whereas:

would yield:
and vice versa.

Given the above, it is evident that:

![Diagram showing relationship between good and bad customers with respect to full payments before due dates.]

describing a customer who makes a full payment before the due date is a good customer could also be drawn, through the earlier general relationship between good and bad customers, as:

![Diagram showing relationship between good and bad customers with respect to full payments before due dates.]

Apart from the bad customer Mr. Warwound being:

![Diagram showing good customer Mr. Warwound.]

there would also occur the result:
The above was introduced spontaneously into the session as an extra example, with the intention of illuminating the interrelationships and thereby additional insight conceptual graphs gave. Instead the participants had essentially rejected the value of conceptual graphs by this point. Their actual comments appear under the following section. Still, however, there were three more areas to cover in this session and, because of the extra effort in trying to convince the subjects as above, the allocated time with them was running short. These three areas were a) another more complex example; b) subtypes and supertypes; and c) the subjects using the CARE software themselves. It was decided to go through these items at least briefly to complete this session properly as intended.

With regard to ‘a)’ above it was necessary to revert back to the abstract ‘if-then’ forms. Therefore the subjects were shown the following graph:

It was explained that the above showed “If ‘C’ and ‘D’ then ‘E’.

From this graph the subjects were directed to the graph:
Here it was shown how many ways this graph could be read. Apart from “If F and not G then H”, the graph could also be read “If F and not H then G”, or “If not G and not H then not F”. Again the intention was to bring about each subject’s awareness of the additional insight from this.

The above was followed by a ‘practical’ case, included in the questionnaire. Given the time constraint and the potential danger of the subjects losing interest altogether, the participants were left to comment about it in the questionnaire later. The ‘practical’ graph was:

![Diagram of the graph]

From here subtypes and supertypes were explained. As an example “‘Borough Treasurer’ is a subtype of ‘Staff Member’” was discussed. Thereby every Borough Treasurer is a Staff Member, but not every Staff Member is a Borough Treasurer. Also illustrated was ‘Bad Customer << Customer’ (Bad Customer is a subtype of Customer, or Customer is a supertype of Bad Customer). Hence a graph that contained a type label would also be effective for that label’s subtypes. Again this was covered in the questionnaire.

After breaking the session for fifteen minutes to eat lunch, so giving the subjects a respite, the group consented to ‘Expert E’ using the CARE software. With the aid of the group, and somewhat extensive facilitator prompting, he drew the following graph:
The above graph is associated with the problems the council face when attempting to obtain a subsidy for an activity from central government. Here the subsidy form has to be completed by the due date and be certified by the auditor, so that the council could receive payment. The above 'semi-trivial' real problem was selected by the subjects to see if conceptual graphs would further enlighten them in how to tackle the above. They could not see how what they drew would help them in any way. After the session the above 'rough' graph was refined by the facilitator into:

On examining the result, it was felt that even this refined form would not have significantly changed each participant’s mind.

All the above outlined events at this session revealed the subjects were substantially disposed against conceptual graphs. We now examine the fuller effect of this result through the participants’ a) verbal comments during the session, and b) their grading of the statements in the subsequent questionnaire. The grades from ‘b)’ were assessed statistically. A space was also
provided for each subject to add written comments at the end of his completed questionnaire, which was subsequently returned. Such comments that did occur are replicated in this report, and remarked upon as appropriate.

6.26 Melton Participant Disposition Towards Conceptual Graphs

At the outset it was stressed to the participants that the session's objectives would be best served if they could be as honest as possible in their evaluation of CARE. This would not present a major problem were they favourable to conceptual graphs. I was somewhat concerned, however, that they might not wish to personally upset me should they think the converse. To prevent this it was stated the purpose of the session was to test the hypothesis that "conceptual graphs are a suitable knowledge-based decision support tool for use by management accountants in strategic planning". Given the session was about this test, a result denying the hypothesis was as equally valid as one that approved it. Thereby the quality of my work would not be prejudiced in any way if they felt they had to be critical. With this potential obstacle removed, the session's salient dialogue with its implications occurred as follows.

To begin with, Expert D commented that he could appreciate what CARE was trying to achieve but found that the conceptual graphs' were very difficult to follow. He also remarked that this was probably the opinion of the whole group. Expert B added that this difficulty was at too high a level given the trivial problems discussed at the session. Expert F reinforced this view by stating plainly "You don't need a conceptual graph to show Mr. Warwound is a bad customer". Expert B asked if a more meaningful scenario could be provided. He offered one problem of interest to him that seemed ideal for CARE. He would like a computer program that could assist him in coping with the legal, social and psychological 'grey areas' the council faced in regard of employee maternity leave benefits. However the resulting far more complex conceptual graphs would clearly prevent such an illustration, because the subjects were not happy with the surface form of conceptual graphs for even simple examples.

Expert A saw that CARE's conceptual graphs were too intricate for Melton Borough Council's requirements. He added they were not really suitable for business decision makers anyway, and the only place conceptual graphs may find a use is such as in a large 'research institute' type of team modelling
some large scale problem such as the UK economy\textsuperscript{14}. Again this precluded that which would support the hypothesis above.

Expert B asked who would draw the graphs. In answer, the facilitator stated it would be people like himself. Expert A wondered if more than one person would model the problem in conceptual graphs. The facilitator answered that a typical modelling situation might involve the initial graphs being drawn by one accountant who could then consolidate his or her results with graphs drawn by other accountants. CARE, through the JEHCGP engine, could be used to check for inconsistencies between the models. This would draw out differing opinions which could then be resolved to obtain a more focused model. Such activity, as it expanded into capturing other problem domains of interest to the relevant organisation, would evolve into its corporate knowledge base.

Expert F then asked about a library of already drawn graphs from which the user could build models without always having to start from the very beginning. The facilitator explained that this would form part of the situation described to Expert B above. Also there were the 'If-Then' forms, but the subjects were not keen on them. Expert F wondered if CARE could be made easier to follow. The facilitator described how, through the use of subtypes, one smaller graph could equate to a larger one. The facilitator asked Expert F if this satisfied his query. He stated he was uncertain. His query was clearly symptomatic of the participants' dislike for conceptual graphs, underlining each participant's view that conceptual graphs offered too much complexity for too little information. The subjects really wanted the conceptual graphs-based interface of CARE to be hidden away from the user.

Expert E queried the value of flowcharts. To him they really only suited highly regulated situations such as in auditing. They were inappropriate to ill-defined strategic domains. Expert B asked what did CARE have over and above conventional flowcharts. The facilitator illustrated how those flowcharts were inherently procedural. The route of reasoning runs along a line that starts at 'A' and ends at 'Z'. If a given query just fell into this pattern, then they are arguably adequate. Should the query also, however, require how

\textsuperscript{14} He made only a vague reference to this point at the session. It was essentially elucidated 'off-tape' by the subject afterwards.
‘S’ affected ‘E’ then the procedural limitation becomes apparent. The conceptual graphs are declarative, so it possible to build up lines of reasoning in any direction. As far as Expert E could see though, conceptual graphs merely exacerbated flowcharting’s limitations. Expert F confirmed this opinion. As far as the other comparison with brackets in the bookkeeping model, the subjects agreed with the similarity but this did not extend into highlighting the use of negative contexts. The comparison thereby became meaningless.

Expert F once again pressed the facilitator on what CARE was telling them. He asked if it was giving answers or not. He stated a spreadsheet automatically showed that ‘2 + 2 = 4’ but CARE would have to be told this. The facilitator explained that CARE was targeted at strategic issues, where the problems were not solvable solely on the basis of ‘2 + 2 = 4’. Expert E commented it was apparent that CARE was tackling the very subjective area of top strategic management, but Expert F was not stating that all problems could be solved by numerical analysis alone. It was that conceptual graphs were not getting their message sufficiently across to the user.

When the facilitator asked at first if anyone wished to have a go on the CARE software, the subjects responded with quiet misgiving. It was this atmosphere that principally undermined CARE. Given that none of the subjects were eager to try CARE at this point, the facilitator attempted to elicit their reservations by simply asking what they thought.

Expert A and Expert F decided CARE was simply too difficult to follow. Expert F added that with CARE it was too easy to miss something on the interpretation. In his view the conceptual graphs needed to be able to pull the user towards the answers. Expert C could follow that ‘2 + 2 = 4’ but could not get to grips with CARE. When the facilitator asked whether the problem lay basically in one of understanding logical complexity, Expert D could agree but Expert E and Expert F felt that users are unlikely to be interested in conceptual graphs irrespective of any further training the users were given.

The group concurred that conceptual graphs would scale up badly. Expert B made the amusing yet salient remark, strongly confirmed by all the other participants, that “If all our problems were simply someone not paying their VAT then you would not need all of us”. Expert B added that anyone would
only build a spreadsheet if the time and effort was worth the perceived benefit, and he could not see that the benefit of building a conceptual graphs model outweighed the complexity of the modelling technique itself. It was interesting that Expert B had specifically referred to conceptual graphs than CARE, showing that the CARE interface was effectively highlighting the nature of conceptual graphs.

Lastly, the facilitator checked with John Heaton to see what he thought about the session. He agreed that there was indeed cause for concern.

From all the above, it was clearly evident that the participants had rejected the value of conceptual graphs. Expert F aptly summarised the general view by stating he did not like what he called 'their look and feel'. He criticised them as of far too-academic value and they would not stand up in the real world.

6.27 Analysis of Questionnaire From Melton Session

Appendix B/14 details the participant’s answers to the CARE questionnaire. That appendix also includes those answers' mean, median, standard deviation and range values. These values are illustrated by the charts that make up Figure 6.03.

![Figure 6.03(a): Appropriateness of background (Corrected for ‘4’ and ‘23’).](image)
As can be seen from Figure 6.03, the CARE questionnaire's statements have been grouped in the above charts under three broad headings. Those headings depict the key areas that the questionnaire was designed to elicit from the Melton participants, as representative of the strategic management accountant. In addition, as denoted under each (a) to (c) sub-figure of Figure 6.03, some of the statement values have been 'corrected'. These
corrections were made by switching the sign of the numeric grades for these statements, because they were worded in an opposite tone to their appropriate broad heading. Thus positive values always support the broad heading, with negative ones always supporting the contrary\textsuperscript{15}. For instance, disagreement with statement '4' - "A computer will replace my job within the next ten years" - in fact demonstrates participant awareness of the general ongoing limitations of computers. A '-2' rating on this statement, for example, would thereby be corrected to '+2'. Some of the statements were phrased in the opposite tone to avoid conditioning the participant into following some erroneously habitual answering pattern.

Each sub-figure of Figure 6.03 shows the means and standard deviations in a first chart. The median, maximum, minimum and range is shown in a second chart. The charts in '(a)' indicate that the participants were of a suitable background, in showing an awareness of computer accounting issues. In '(b)' they furthermore indicate an understanding of conceptual graphs. The charts in '(c)' reveal, however, that they were disposed against the use of the conceptual graphs-based CARE as a knowledge-based decision support tool.

In respect of the levels of variance between each participant in answer to given statements, we can see that each subject did show a comforting amount of independence from each other. This is demonstrated by the wide spectrum of answers in respect of the subjects:

a) views towards the computer over manual methods, from the answers to statements '3a' and '3c'.

b) understanding of conceptual graphs, as evident in the answers to statements '7', '8', '12' and '13'.

c) opinions towards CARE making the dimensions of problems more apparent, from statements '14b', and '14c'.

d) comparison of CARE with techniques familiar to the accountant, from '15e', '16', '17e' and '18'.

\textsuperscript{15} The exception being the standard deviations, which are always positive.
e) appreciation in respect of CARE’s relationship with the accountant, from ‘20’, ‘21’, ‘22’, ‘23’ and ‘24’.

Overall we can see, from the above initial analyses, that the participants were highly unsupportive of conceptual graphs’ worth to the strategic management accountant. To elucidate upon the above findings, the questionnaire would be subjected to further statistical analysis. The scope for any such elucidation based on the above mean and standard deviation measures was, however, severely limited. This was because both of these measures are parametric in nature. These measures thereby make a too-strong assumption about the CARE questionnaire data, as explained next.

Parametric tests assume that the data is measured in an ‘interval’ scale. For example a response of ‘3’ indicates 3 times the strength of feeling of a response of ‘1’, and that the data conforms to a normal distribution. Such assumptions are very unusual for a questionnaire, as its data generally represents only an ‘ordinal’ scale: A response of ‘3’ is stronger than a ‘1’, but by how much cannot be stated.

Given that the CARE questionnaire was ordinal, it was decided that any further analyses of the questionnaire should be based upon the more conservative nonparametric statistical tests. Such tests do not make the interval assumptions outlined above. Additionally, as illustrated by Figure 6.03, the CARE questionnaire was designed to elicit the interrelationships between a) the background of the participants, b) their understanding, and c) their representative judgement of employing conceptual graphs in the strategic problem domain. In view of these associations it was decided that the most appropriate avenue for further investigation would be through correlation. There exist correlation tests in both parametric and nonparametric statistics. Therefore, given we are only interested in the latter type of statistic, the CARE questionnaire was to be subjected to nonparametric correlation tests. All the above, including the actual nonparametric tests of correlation, are further defined and detailed in standard texts on nonparametric statistics such as Siegel (1956) and Spent (1989). The essence of correlation is mentioned below, together with the nonparametric correlation tests chosen for the CARE questionnaire.
Correlation measures the degree of association between two variables. The statistic ranges from '+1' to '-1'. A positive value shows that one variable can be used to estimate the value of the other, and vice versa. The closer the value is to '+1' the stronger that relationship is. A negative value works in exactly the same way except that the relationship is an inverse one. This means as one variable increases the other correspondingly decreases and vice versa. Again the closer the value is to '-1' the stronger that inverse relationship is. There is no correlation when the value is '0'. The nature of correlation should become clear in the actual values from the CARE questionnaire, as discussed later.

A probability of error, 'p', is assigned to the correlations and is determined as part of applying the correlation model. Conventionally $p \leq 0.05$ is required to be certain that a valid association exists. If there is a hypothesis that also expects not only a correlation but also its direction (either +ve or -ve), such as that one's understanding of computers will be negatively associated with understanding CARE, then it is possible to conduct a one-tailed test. Otherwise a more conservative two-tailed test has to be performed.

For the CARE questionnaire analysis, a number of hypotheses were previously formulated to encapsulate both the expected correlations and their direction. These hypotheses appear below. One-tail correlation tests were thereby employed on the CARE questionnaire answers to seek confirmation of predicted interrelationships between questions. The actual correlation was performed using the advanced statistical package SPSS Release 4.0 on the Hewlett-Packard HP9000 UNIX computer at Computing Services, Loughborough University. SPSS offers two nonparametric methods of computing correlations: 'Spearman' and 'Kendall'. Kendall tends to be more conservative than Spearman in determining significant results, yet is arguably better suited to smaller sample sizes such as in the Melton session. For the sake of completeness, though, Appendix B/15 details both the Spearman and Kendall correlations for the CARE questionnaire, and the results include the correlations from both one-tailed and two-tailed tests.

---

16 Also for the sake of completeness, all the significant results of the tests were shown regardless as to whether they were a) part of any hypothesis, or b) positive or negative in direction.
Chapter 6

The above mentioned hypotheses were based on and around the general hypothesis that "conceptual graphs turn out not to be a suitable knowledge-based decision support tool for use by management accountants in strategic planning", as evidenced by the CARE user evaluation sessions discussed in this chapter. For each sub-hypothesis, the null hypothesis is that there is no correlation between the two parameters as described by its alternative hypothesis. The alternative hypotheses appear below, in italics. Under each alternative are any appropriate correlations from Appendix B/15. For the null hypothesis to be rejected in favour of the alternative hypothesis, the correlation needs to be significant at \( p \leq 0.05 \). The implications of such correlations are also discussed.

a) An increasing user-awareness of computers corresponds to an increasing user-appreciation of conceptual graphs. This hypothesis would support the view that the accountant would not dismiss CARE simply through general computer illiteracy. The null can be rejected in favour of this alternative from the qualifying correlations, where the suffix "(-)" signifies a negative correlation, between the statements: '1' & '7'; '1' & '8'; '1' & '15b'(-); '1' & '15c'(-); '1' & '15d'(-); '2' & '13'; '3a' & '16'; '3b' & '16'; '3c' & '18'; '3d' & '5'; '3d' & '6'; '3d' & '11'; '3d' & '13'; '3d' & '14a'; '3d' & '15b'(-); '3d' & '15d'(-); '3d' & '15e'(-); '3d' & '23'(-); '3e' & the same as '3d'; '4' & '9'; '4' & '15a'; '4' & '15c'; '4' & '20'.

b) An increasing user-understanding of conceptual graphs corresponds to a increasing user-appreciation of the similarity with respected existing accounting techniques. Even though the subjects might be dismissive of CARE, they could agree the basis under which conceptual graphs might have brought advantages to the accountant. Acceptance of this alternative would vindicate bringing CARE to the user evaluation stage. The null can be rejected in favour of this alternative from the qualifying correlations between the statements: '5' & '14a'; '5' & '14b'; '5' & '18'(-); '5' & '23'(-); '6' & the same as '5'; '7' & '15a'(-); '7' & '15c'(-); '8' & '15a'(-); '8' & '15b'(-); '8' & '15c'(-); '8' & '15d'(-); '9' & '15a'(-); '9' & '15c'(-); '10' & '15e'; '10' & '17e'; '10' & '18'; '10' & '23'(-); '11' & '23'(-); '13' & '14a'; '13' & '15e'; '13' & '17e'; '13' & '23'(-); '14a' & '18'; '14a' & '23'(-); '14b' & the same as '14a'; '14c' & the same as '14a'; '15e' & '17e'; '18' & '23'(-).

c) An increasing user-understanding of conceptual graphs corresponds to a decreasing user-assessment of their worth. The more accountants could
get to grips with conceptual graphs, the less they saw the benefit in the technique. In accordance with the discoveries made at the user evaluations of this chapter, accepting this alternative would demonstrate that accountants are unlikely to deploy conceptual graphs even if these domain experts were given further training. The null can be rejected in favour of this alternative from the qualifying correlations between the statements: '7' & '21'; '8' & '21'; '15a' & '21'(-); '15c' & '21'(-); '15e' & '20'(-); '17e' & '20'(-); '20' & '22'.

d) An increasing user-appreciation of conceptual graphs corresponds to a decreasing user-assessment that CARE will replace the strategic management accountant. In a previous user evaluation session, Expert 2 raised the concern that conceptual graphs were going to cause CARE to be too prescriptive. Again this would raise the concern that CARE was prematurely brought to the user evaluation stage. Accepting the alternative would suggest that view was unfounded. In the event, doubt arises as the participants may be rejecting CARE irrespective of this particular concern. Given the benefit of this doubt, however, the null could be rejected in favour of this alternative from the qualifying correlations between the statements: '3d' & '19'(-); '3e' & '19'(-); '10' & '19'(-); '11' & '19'(-); '13' & '19'(-); '14a' & '19'(-); '14b' & '19'(-); '14c' & '19'(-); '18' & '19'; '19' & '23'.

There were also certain non-hypothesized dimensions that came to light. To begin with, the correlations of '12' & '20'(-); '12' & '22'(-) and '13' & '20'(-) appeared to indicate an incongruity with the correlations supporting the alternative hypothesis in 'c)' above. These correlations involved actual 'Bad Customer' conceptual graphs examples as distinct from the abstract "'A' to 'H'" conceptual graphs. Thus such correlations could arguably be highlighting conceptual graphs' inadequacy at feeding back, to the domain expert's satisfactory understanding, some meaningful insights about actual problems. Through this shortcoming, rather than any lack of sufficient user-understanding about conceptual graphs theory itself, the participants were indicating that CARE's usability would consequently falter when faced with the intricacies of real domain problems. This argument is supported from Expert F's earlier stated perception that, with conceptual graphs, it is too easy to miss something in their user-interpretation. Accepting the argument,
these correlations actually support the alternative hypothesis in line with the other correlations.

In respect of 'b)' above, the opposite to the expected correlation appeared to occur in respect of the flowchart comparison. This is shown by: '7' & '17b'; '7' & '17c'; '7' & '17d'; '8' & '17b'; '8' & '17c'; '8' & '17d'; '9' & '17b'; '9' & the same as '8'; '15a' & '17b'; '15a' & '17c'; '15a' & '17d'; '15c' & the same as '15a'. Even though not hypothesized, this result does not indicate a contradiction of the alternative hypothesis. This is because these correlations were underlining the participants' little regard for flowcharts too. Note also the support given to this view by the correlations between '4' & '17b'; '4' & '17c'; '4' & '17d'; '16' & '24'; '17b' & '21'; '17c' & '21'; '17d' & '21'.

If so hypothesized, the correlations between '10' & '17e'; '13' & '17e'; '15e' & '17e'; '17e' & '20' might also have suggested that the alternative hypothesis in 'c)' above was being erroneously accepted. Again there is no contradiction, though, as flowcharts were viewed so badly that conceptual graphs could only be seen as an improvement. Similarly '10' & '15e' and '13' & '15e' recognised that the numerical/data-based nature of spreadsheets were so poor at handling strategic issues that, again, conceptual graphs could only be better. This latter aspect offers some support, by accountants in practice, to the arguments about spreadsheets in Chapter 1.

The numerical ranks the Melton participants gave in answer to the CARE questionnaire, through statistical analysis, therefore support the view that conceptual graphs are not a suitable knowledge-based decision support tool for use by management accountants in strategic planning.

6.28 About Written Comments Added by The Melton Participants

Below are the actual comments that some of the subjects wrote in addition to numerical grades to the statements on the questionnaire. These remarks appear in italics. An analysis supplements each comment accordingly.

Expert D:

1. I do not know enough about CARE to answer questions 14, 15, 17.
2. With regard to question 20 I feel it would be useful to have a problem which gave an answer pitched between the trivial and a very complicated problem. This may not be possible. So you could see a tangible result.

3. I felt I needed a higher level of education to be able to appreciate the tool in order to make a more meaningful assessment.

Although Expert D had also found CARE difficult like the other participants, he was less dismissive in that he felt this obstacle might be overcome through a 'higher' education. From the inherent reservations arising from the previous user evaluation sessions, it was considered that such an education would not be practically useful to accountants. The salient item in the above is that Expert D recognised the 'complexity-at-triviality' of conceptual graphs to such an extent that he added it as a written comment.

Expert E:

1. CARE's applicability would only be relevant to top strategic level planning.

2. CARE even in it's simplest form seems complex. Therefore at a realistic level the variables would be endless and very subjective rendering CARE too costly and untimely to users.

3. CARE to the basic computer literate person like myself represents the idea of flowcharting. If you do not feel very comfortable with the idea of flowcharting and its graphical representations then CARE's user friendliness is destroyed.

Here Expert E agreed about the top strategic level at which CARE was pitched, but then highlighted the key reservations about conceptual graphs from the session. The dislike of flowcharts, with its consequence for conceptual graphs, is also evidenced.

Expert F:

An interesting concept but perhaps too complex for the average accountant to use.
Chapter 6

Needs to be tested in a practical environment to fully assess its potential. With any software the 'look and feel' is very important to the user. For the product to be marketed commercially would need improvements to the interface with the user and to be able to upload/download information to/from other systems. Suggest danger - could be seen as just flowcharting under another name.

Thank you for the demonstration which was both interesting and thought provoking and best wishes with your thesis.

Again Expert F underlined the events at the session. In the above, Expert F felt that CARE should be further tested in a realistic industrial situation. Quite apart from the impracticalities preventing such an exercise, both the current session and the previous ones had made the 'complexity-at-triviality' problem evident. Hence it is extremely improbable that any subsequent evaluation of the sort that Expert F is describing would reveal a favourable outcome. Expert F also criticised the interface. From this session that meant the domain expert avoiding the direct use of conceptual graphs. The 'upload/download information to/from other systems' point is an interesting one. The idea of more comprehensive graphs evolving through CARE has been discussed earlier. Linking the graphs to databases, spreadsheets, or other knowledge-based systems was another avenue for CARE. This was mentioned in Chapter 5. However, given that the conceptual graphs fundamental to CARE fell at the first hurdle, such an avenue of exploration could no longer be justified. Expert F clearly saw flowcharts as a comparison that would hinder, not augment, CARE's viability. In essence, Expert F's idea of CARE would have excluded conceptual graphs.

6.29 Comments on the Melton Session

It was clear from all the above that the Melton participants, and hence management accountants in the real world, were not going to direct their skills through conceptual graphs into solving strategic problems. Sadly, the previous suspicions arising from the academic domain expert sessions were confirmed by the Melton session. Therefore it has to be concluded that the accepted conceptual graphs technique turns out not to be a suitable knowledge-based decision support tool for management accountants within strategic planning.
6.30 Summary of Evaluation Sessions with Domain Expert Users

To discover 'the usability of conceptual graphs by strategic management accountants' objective, it was always going to be necessary to test conceptual graphs amongst representative domain experts. CARE provided the considered best means of determining this objective. The first user session with the conceptual graphs community at their annual workshop well-placed CARE, and indicated that conceptual graphs would indeed be usable in the above capacity.

Then CARE was taken into an academic domain expert environment. Throughout all the sessions CARE was enhanced iteratively in accordance with the results of each evaluation session. At the first point of contact between CARE and a domain expert, concerns began to be raised as to conceptual graphs' adequacy for the strategic management accountant. The intricacies of the strategic problem being modelled by the graphs, even at a trivial level, began to render the resultant graphs difficult to interpret by the user. As the evaluations progressed, this situation became increasingly obvious. By the time CARE was presented to actually practising accountants, where it was confirmed that conceptual graphs faced too-fundamental hurdles, even the statistical analyses carried out from that session employed hypotheses that predicted the failure of conceptual graphs in the above capacity. Not surprisingly, those analyses met this new expectation.

In the course of the above, there were two areas which might have revealed that CARE should not have been brought to the user evaluation stage at all:

a) There was a concern that conceptual graphs were inherently moulding CARE into a too-prescriptive role, directly contravening the basic intent of the CARE methodology.

b) Doubt was raised in regard as to the sense in drawing comparisons between conceptual graphs and i) negations in the accountant's bookkeeping model, and ii) structured diagrams.

However the user evaluations demonstrated these worries were unfounded, and it was proper to user test CARE as described. It was only at the user
evaluation stages that the validity of the above comparisons would become truly evident, such as the practising accountant's poor regard towards flowcharts.

Lastly, to validate further the user evaluation sessions of this chapter, the business school and Melton participants were shown this chapter and supporting appendices. All the participants confirmed their agreement with these contents.

6.31 Concluding Remarks

Despite their strong prima facie attractiveness and positive response from the conceptual graphs community session, as the user evaluations progressed it became increasingly evident that the inherent complexity of conceptual graphs fundamentally undermined them as a viable tool, other than for very trivial problems well below the level needed to be viable for strategic management accountancy. From the various studies already discussed in this thesis, such as Chen (1985) and Novak and Gowin (1984), Sowa (1984-1991b) had argued the sound psychological basis on which conceptual graphs are based. Nonetheless there is no escaping the evidence from the domain expert sessions that conceptual graphs have not aligned easily with human reasoning. The fear of burdening the domain expert with too high a computational knowledge requirement, mentioned in the early part of Chapter 2 as a limitation of rule-based expert systems, turned out to be fundamentally relevant to conceptual graphs as well. The type hierarchy inadequately offered the expected creation of small yet succinctly informative conceptual graphs modules, and thus underlined the computational burden on the user.

The above outcome can be arrived at from another perspective: It so happened that none of the domain experts at any of the sessions commented about why the output was not in the graphical form like the rest of CARE. From the pilot study discussed in Chapter 5, it was shown that future sessions should be based solely on the more visually-impacting graphical form of conceptual graphs rather than the alternate linear form. For the reasons given in Chapter 5, a residual linear form conceptual graphs element from JEHCGP's interface remained in the shape of CARE's output. From these circumstances it may seem surprising that this display matter was not raised, yet on further
reflection it can be seen that the subjects were so sceptical about the graphical form that they were unconcerned about any improvement over the linear form. It would have been interesting to have included a statement like 'I prefer conceptual graphs in the graphical to the linear form' in the Melton questionnaire. From each subject's reaction at this session the likely overall answer would have been close to the '0' mark i.e. 'indifferent'. Such a statement was not included due to the evidence from the Chapter 5 pilot study. The fact that there remained a residual linear element re-tested the display comparison in this chapter's user evaluation sessions, yet now revealing the above new facet.

Although the CARE software performed well in bringing the salient features of conceptual graphs theory across, all the domain expert participants that understood conceptual graphs were particularly unimpressed by how highly complex even the more 'user-friendly' graphical form of conceptual graphs became at such a low trivial level. The graphical form had no additional visual impact over the subjects, as evidenced by the lack of the subjects' preference over the linear form. All in all, it can no longer be suggested that conceptual graphs are a suitable knowledge-based decision support tool for strategic management accountants.
7 Conclusions

7.01 The Hypothesis

The hypothesis of the research was "conceptual graphs are a suitable knowledge-based decision support tool for use by management accountants in strategic planning". The events depicted by this thesis report eventually lead to the conclusion, and thus the original contribution of this research, that the above hypothesis turned out to be false. Those events are summarised as follows.

7.02 Retracing the Events

It was identified in Chapter 1 that computer knowledge-based approaches could help accountants apply their skills in the management of the strategic planning problems. The inexactness of this domain causes it to exist beyond other calculable means, namely numeric or data-based techniques. We saw that such problem domains could not be modelled effectively by computer alone, hence the only advanced knowledge-based methodologies of interest were those that could be adequately reviewed by the above management accountant in the light of this domain expert's own continually changing tacit and implicit knowledge. This situation was supported by examining a commercially available 'reinsurance' expert system, and the 'system dynamics' technique. From the foregoing discussion, this chapter indicated that the thereby knowledge-based decision support tool needed both to be technically advanced yet directly usable by the domain expert.

In Chapter 2, we saw that structured diagram techniques, such as flowcharting, were well known by accountants and were a clearly understandable yet important aid in problem review. Apart from being founded on a logically complete reasoning system, the knowledge-based methodology of conceptual graphs was formulated to be an enhancement of these other methods. The chapter had highlighted the limitations of those methods. From all these bases, the potential appropriateness of conceptual graphs was first revealed as a directly usable tool by the strategic management accountant. A general description of the conceptual graphs theory was given,
followed by a knowledge-based example specifically in an appropriate problem domain. That example was based on the imprecise management accounting area of ‘relevant values for decision making’. Chapter 2 thus supported the view that conceptual graphs might be usable by the strategic problem solving management accountant, who was by now referred to more succinctly as the strategic management accountant.

To determine further the extent of conceptual graphs' technical capability in the strategic problem domain, Chapter 3 and Chapter 4 each discussed a comparative study between conceptual graphs and a similarly appropriate methodology in current use. These other two methodologies, ‘cognitive mapping’ and ‘economic accounting’, underlined the technical advantages of conceptual graphs. Again these chapters supported the basis for the above hypothesis.

Given all the above support, Chapter 5 set out the most appropriate means of presenting conceptual graphs to the strategic management accountant. From the experiences with knowledge elicitation, we saw that a highly interactive human-computer environment would be necessary. The thesis' findings recommended that this integration would be best augmented by conceptual graphs. Furthermore it was identified that conceptual graphs enjoyed another similarity with a technique familiar to the accountant. This similarity was with the ‘negating’ brackets in the accountant's traditional bookkeeping model. Leading on from all the above, the Conceptual Analysis and Review Environment, or CARE for short, was devised. CARE was implemented in computer software.

In Chapter 6, CARE was employed to test the accepted graphical form of conceptual graphs through a series of user evaluation sessions. The evaluations started out with subjects from the conceptual graphs community itself, then key business school staff, and culminated in a session with senior practising accountants. In addition, CARE was enhanced iteratively in accordance with the results of each evaluation session. However, despite their strong prima facie attractiveness and positive response from the conceptual graphs community session, as the user evaluations progressed it became increasingly evident that the inherent complexity of conceptual graphs fundamentally undermined them as a viable tool, other than for very trivial problems well below the level needed to be viable for strategic management
accountancy. Therefore, as stated above, the original contribution of this research is sadly that its hypothesis turns out to be false.

7.03 Suggested Implications

Given my interest in seeking to extend the accountant's soundly developed techniques into the top-level area of strategic business knowledge, this was a disappointing result. It is thereby important to discern the implications of this result, so that others do not follow the same path. In recognition of this regard, the following considerations are suggested.

To start with, let us look at the role of conceptual graphs theory and ask "Are conceptual graphs an inappropriate tool in strategic management accountancy?". After all, the title of this thesis is "The Suitability of Conceptual Graphs in Strategic Management Accountancy". This thesis has shown that they are not for strategic management accountants, but the above question is somewhat more general. Conceivably, conceptual graphs could be used in this problem domain except now they would not be seen by the domain expert. On this success, the more general statement would be confirmed whilst its specific instance denied. In itself, this state of affairs would not be inconsistent.

We have seen, however, that the advanced knowledge-based tool must be directly usable by the domain expert, especially in the strategic problem domain. It is therefore insufficient to demonstrate that the strategic domain can be represented in conceptual graphs. This technical capability must be augmented by exhibiting a surface form the domain expert can understand. This may, in the shape of some significantly enhanced conceptual graphs theory, or some other suitable representation.

It may be argued that perhaps the only clearly domain expert usable representation is natural language. We know, though, that natural language can be ambiguous and obfuscatory. In addition, natural language ignores the visuality of pictures. Despite the usability reservations revealed in this thesis about structured diagrams, and specifically flowcharts, the overall findings from the studies referred to in this thesis reveal that good visuality does aid human understanding. Discovering that visuality is the real challenge that
lies ahead. The visual aid must not merely be usable, for this is likely to lead towards that tool being too elementary. We are aware that it must be technically capable as well.

In the strategic problem domain, it may be that technical capability and usability is too fundamental a conflict. The technical dimension necessitates complexity whilst usability requires simplicity. This thesis demonstrates the difficulty, as employing conceptual graphs was an unsuccessful attempt to bridge this gap. Given that conceptual graphs theory is the most technically yet visually advanced knowledge-based formalism, perhaps strategic knowledge may never be computer manipulated as a result. There appears to be no other currently available representation that can resolve the conflict. The design and investigation of such representations, should they arise, would be valuable future work.
Bibliography


Coulthurst, Nigel; Piper, John (1986a) "The State of Cost and Management Accounting", Management Accounting, April.

Coulthurst, Nigel; Piper, John (1986b) "The Terminology and Conceptual Basis of Information for Decision Making", Management Accounting, May.

Coulthurst, Nigel; Piper, John (1986c) "Framework for Analysis of Decision-Relevant Costs and Benefits", Management Accounting, June.


Garner B. J. and Tsui E. (1985a) "Knowledge Based Audit", Draft paper, Division of Computing and Mathematics, Deakin University, Australia.


Hayes-Roth, Frederick; Waterman, Donald A.; Lenat, Douglas B. (eds.) (1983), Building Expert Systems, Addison-Wesley.

Heaton, John E. (1989a) An Investigation into Knowledge Engineering using Conceptual Graphs, MSc Project, Department of Computer Studies, Loughborough University of Technology.
Heaton, John E. (1989b) *Conceptual Graph Processor - User Guide*, MSc Project, Department of Computer Studies, Loughborough University of Technology.


Bibliography


Kowalski, R. (1979a) Logic for Program Solving, Amsterdam, North-Holland.


Licklider, Tracy Robnett (1989) "Ten Years of Rows and Columns", Byte, December.


Bibliography


Parsaye, Kamran; Chignall, Mark; Khoshafian, Setrag; Wong, Harry (1989) Intelligent Databases, Wiley.


Polovina, Simon (1989) Software for the Artificial Intelligence Course: Syntactic Pattern Recognition Systems, MSc Project, Department of Computer Studies, Loughborough University of Technology.


The Appendices

A/01: Pilot Study Paper

Enriching Cognitive Mapping: A Technical Comparison between COPE and Conceptual Graphs based on an 'Office Location' example.

Simon Polovina
Department of Computer Studies, Loughborough University of Technology

(Unpublished paper used in Pilot Study conducted on March 11–12, 1992)

Introduction

This paper compares and contrasts the way knowledge is captured and manipulated between COPE and conceptual graphs. COPE, an acronym for Cognitive Policy Evaluation, is existing computer software whereas a conceptual graphs processor is a potential such candidate. COPE depicts the cognitive maps of Eden (1991), whose work stems from that of Kelly (1955). Conceptual graphs are another form of cognitive mapping devised by Sowa (1984) except they appear to be far more technically sophisticated than COPE and, because they are also visual in nature, have the potential to retain COPE's usability.

The aim of this paper is to demonstrate the above enhancement using the same realistic office location problem that is cognitively mapped into COPE form by Ackerman, Cropper, and Eden (1991). The existence of such a technical superiority needs to be established before it is worth considering any issues of comparative usability. Ackerman et al.'s example is a suitable candidate because Ackerman et al. would have chosen it conceivably to highlight the advantages of the COPE model. Therefore, any further reasonable improvement discovered from the use of conceptual graphs is of relatively significant importance.

The example itself is as follows (Ackerman et al. 1991, page 41):

"We need to decide on our accommodation arrangements for the York and Humberside region. We could centralise our service at Leeds or open local offices in various parts of the region. The level of service we might be able to provide could well be improved by local representation but we guess that administration costs would be higher and, in this case, it seems likely that running costs will be the most important factor in our decision. The office purchase costs in Hull and Sheffield might however be lower than in Leeds. Additionally we need to ensure uniformity in the treatment of clients in the region and this might be impaired by too much decentralization. However we are not sure how great this risk is in this case; experience of local offices in Plymouth, Taunton and Bath in the South West¹ may have something to teach us. Moreover current management initiatives point us in the direction of greater delegation of authority."

The above problem is first examined in its COPE form and then re-mapped into conceptual graphs. It is in the course of this re-mapping exercise that the technical comparative issues are evaluated.

¹ Author's note: The passage actually states 'South East' but this is a typographical error as confirmed by the references to 'South West' later in the article.
The COPE Mapping

Ackerman et al., cognitively map the above problem and produce the following diagram as a result (Ackerman et al. 1991, page 47):

The above diagram shows the two basic elements in Eden's interpretation of cognitive mapping, based on Kelly. Namely these elements are 'concepts' and 'links'. Each concept is represented as one 'pole', which describes one side of the problem, and a 'contrasting pole' which is meant to focus the concept by a meaningful contrast to the first pole. Poles may lead to other poles by means of directed links. All this is embodied by the COPE software which is now discussed.

The above concepts give the following table in COPE:

1. open local offices → centralise services at Leeds
2. local representation → (not) local representation
3. increased running costs → (not) increased running costs
4. higher administration costs → (not) higher administration costs
5. improve level of service → (not) improve level of service
6. too much decentralisation → (not) too much decentralisation
7. risk of impaired treatment of clients → ensure uniformity of treatment
8. lack of understanding about risk → (not) lack of understanding about risk
9. use experience of s w local offices → (not) use experience of s w local offices
10. lower purchase costs of local offices → higher cost in Leeds
11. greater delegation of authority → (not) greater delegation of authority
12. follow current management initiatives → (not) follow current management initiatives

Note COPE adds a sequential number to each concept entered, and automatically adds the prefix '[not]' to create a 'default' contrasting pole for any concept where a contrasting pole was not entered. The links are entered by taking the two appropriate concept numbers and placing a '+' symbol between them. The concept before the symbol leads to the concept placed after it. For example, '10+1' shows that 'lower purchase costs
of local offices' leads to 'open local offices'. This also stipulates the contrasting pole 'higher cost in Leeds' leads to the contrasting pole 'centralise services at Leeds'. Eden's model stresses that it is important the non-contrasting pole should always represent what the user can best identify with. However, this is likely to create problems when it comes to making links as this constraint means poles may lead to poles of the other kind. To cater for this, the '-' symbol replaces the '+' symbol as appropriate. This is illustrated in the following links table:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&gt;+6</td>
<td>+3</td>
<td>+2</td>
<td>.8</td>
</tr>
<tr>
<td>2</td>
<td>&gt;+5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>&gt;+4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>&gt;+7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>&gt;.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>&gt;.-8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>&gt;+1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>&gt;+1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>&gt;+11</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The above table reads from left to right. The number before the '>' symbol states this concept leads directly to all those concepts after this symbol. Each '+' shows a pole to pole, and contrasting pole to contrasting pole, link. The '-' is a pole to contrasting pole, and contrasting pole to pole, link. Hence, the above '9-8' link means 'use experience of sw local offices' leads to '[not] lack of understanding about risk', and '[not] use experience of sw local offices' to 'lack of understanding about risk'. Another link, '1.1', is a 'connotative' link which is employed when the user knows there is an insufficiently definable yet valid connection between concepts. This link is applied to the relationship between concepts '1' and '8' as overcoming 'lack of understanding about risk' may lead to either operating centralised services or opening local offices.

The overall result is illustrated in the COPE diagram below for concepts linked to concept '1':

![COPE Diagram](image)

Concepts which are preceded by '***' mean there are other concepts attached to them. These other concepts can be brought up on screen in this diagrammatic form as required, and there are other commands to show particular direct or indirect diagrammatic relationships between concepts. In addition, as can be seen above, the graphical form suppresses the default '[not]' contrasting poles.

The above example is now explored using conceptual graphs.
The Conceptual Graph Representation

The conceptual graphs representation of the above problem is based on the same cognitive map as identified in the COPE representation above. This approach should ensure a common comparative basis, retain the user-understandable element embedded in Eden's technique, and would highlight vividly any distinguishing features between the two representations.

Starting with the COPE concepts themselves, they appear to fall into two categories. The first are those which have defined contrasting poles and the second which remain undefined. In the latter case, COPE creates the contrasting pole by adding the term 'not' to the input pole. Concentrating on the defined concepts to begin with, these may be modelled in conceptual graphs as follows:

(["centralise services at Leeds"]; ["open local offices"]).

(["lower purchase costs of offices in Hull Sheffield and Harrogate"]; ["higher cost in Leeds"]).

(["risk of impaired treatment of clients"]; ["ensure uniformity of treatment"]).

The pair of poles in each 'COPE concept' became a conceptual graph by placing each pole into a separate 'conceptual graph concept', denoted by square brackets '[]', and together surrounding them by a '()'. These round brackets signify, rather like the way accountants place such brackets around figures to negate them, that whatever is contained within them is false (The semi-colon, ';', merely separates graphs to aid readability and the full stop, '.', shows the end of the graph). Therefore each graph provides contrast by stating that it is false that both poles can exist simultaneously. By the conceptual graph operation of deiteration, elaborated below, when one is true the other is false.

Taking the following graph as representative:

(["centralise services at Leeds"]; ["open local offices"]).

Say that it was decided to centralise services at Leeds. As a conceptual graph this would be shown as:

["centralise services at Leeds"].

Note the absence of the round brackets. This means the above graph is true. In addition to those round brackets negating true graphs to give false graphs, the context of any graph outside of these brackets is said to dominate over the context of any graph inside such brackets. In view of the 'dual' role these rounded brackets play, they may be better thought of as negating contexts. In effect this dominance means if the inside graph happens to 'match' an outside graph then that inside graph can be 'rubbed out', leaving the rounded bracket negating whatever else remains inside it. This is deiteration.

To illustrate, as:

["centralise services at Leeds"].

is not in rounded brackets (negating contexts), it dominates its matching concept inside:

(["centralise services at Leeds"]; ["open local offices"]).
'Therefore this inside concept can be deiterated to yield:

\[ (\{\text{open local offices}\}) \].

This shows that \((\{\text{open local offices}\}\)) is false because \((\{\text{centralise services at Leeds}\}\)) is true. Should the decision be \((\{\text{open local offices}\})\) instead then the result would be \((\{\text{centralise services at Leeds}\}\)) accordingly. Unlike COPE, which passively records the poles, a computer-based conceptual graph processor would actively make such assertions automatically as the appropriate new graphs are included with its base of knowledge.

It should be noted that the above 'true-asserts-false' form does not assert one pole as true should the other be false. For example \((\{\text{centralise services at Leeds}\}\)) and \((\{\text{open local offices}\}\)) cannot be obtained from \((\{\text{open local offices}\}\)) and \((\{\text{centralise services at Leeds}\}\)) respectively. To do this would require the additional 'false-asserts-true' graph:

\[
(\{\text{centralise services at Leeds}\};
\{\text{open local offices}\}\}).
\]

Here there are more negating contexts, some of which are nested within others. In conceptual graphs, whole negating contexts can also be deiterated provided the dominated negating context matches a dominating negating context. This can be illustrated by the following graph:

\[
(\{\text{open local offices}\}).
\]

This graph states 'open local offices' is false. It dominates the matching part in the above false-asserts-true graph because it is surrounded by a lesser number of negating contexts. Hence \((\{\text{open local offices}\}\)) can be deiterated yielding:

\[
(\{\text{centralise services at Leeds}\}).
\]

This has left two negating contexts around 'centralise services at Leeds'. On the principle that a negative of a negative equals a positive, these double negate to give:

\[
\{\text{centralise services at Leeds}\}.
\]

Hence 'centralise services at Leeds' is asserted as true.

In COPE the 'false-asserts-true' aspect is insufficiently clear. COPE may prefer the user to assume if one pole is false then the other is true, yet it is quite possible that the decision maker may for instance do nothing or decide to open mobile offices instead. In this case the above graph, like COPE, would be incorrect. The bipolar nature of COPE cannot cope with this scenario. Even worse, it could provide a too narrow framework which stifles originality of thought. The model does not lend itself to decision makers realising other alternatives, such as mobile offices. In view of this deficiency in the COPE model, the 'false-asserts-true' aspects cannot be transposed to the conceptual graph representation in a manner which guarantees validity.

Moving on, it is possible to leave the conceptual graphs in this elementary 'true-asserts-false' two concept form and manipulate them as such. Ackerman et al. stress that the sentences should remain as they are because the decision maker can identify with what he or she stated directly. This is part of the usability issue and hence falls outside the immediate scope of this paper. In conceptual graphs the above concepts could be refined to capture more about the problem, whilst potentially retaining usability, by the following 'display form' conceptual graphs:
So far the graphs illustrated have been of the 'linear' form which permits input and output using a text-based computer terminal. There is also however the display form, such as shown by the above graphs. This form may be clearer for the user to follow and easier to draw. The 'dotted lines' linking some of the concepts will be discussed below. The surrounding 'rounded boxes' in the above graphs are negating contexts. Conceptual graphs may include relations as well as concepts as shown by the following general form:

```
some concept  the relation  some other concept
```

This may be read as: "The relation of some concept is some other concept". As an alternative to the above display form, the above may be written in the following linear form:
[some concept] -> (the relation) -> [some other concept].

Care must be taken to ensure that, in the linear form, the round brackets around relations are not confused with negating contexts as they are quite separate entities.

A concept can include a 'referent' which conforms to that concept's 'type label' (In fact concepts without referents are concepts with type labels alone). In the above graphs the referent 'Leeds' conforms to the type label 'central office' and 'Hull Sheikhfield and Harrogate' conforms to 'local offices'. The 'dotted line' link referred to earlier shows that these concepts would hold the same referent. Furthermore the type labels are hierarchical. This feature is explicated in the course of the discussion below.

Referring back to the subject which the above graphs model, apart from the bottom graph which merely shortens one of the concept's phrases, these transformations require explanation to understand their basis. Starting with the top graph, 'Leeds' is an instance of a central office. Leeds will have its own peculiarities but it will probably share the same characteristics as any central office. This would permit inferences to be made about Leeds from both what is known about central offices in general and Leeds in particular. This enables a knowledge-base to be built up based on the appropriate degree of generally applicable knowledge. This also prevents unnecessary duplication when the same knowledge applies to more than one particular concept. The degree can be appreciated by developing the Leeds example in a little more detail. It may be that certain things are applicable to Leeds in its own right, Leeds as a Yorkshire central office, as a northern central office, or an English central office as well as a central office. The same principles apply to the local offices. Taking the central office case as representative, the type hierarchy would then include:

'central office' < office.
'English central office' < 'central office'.
'Northern central office' < 'English central office'.
'Yorkshire central office' < 'Northern central office'.

Each type label to the left of the '<' is the more specialised 'subtype' whereas each to the right is the more general 'supertype'. The most specialised conformity for 'Leeds' is 'Yorkshire central office'. This means Leeds conforms to all of the above central offices, but not to say 'Southern central office' ('Southern central office' < 'English central office'). Thereby any inference in respect of Southern central offices would not apply to Leeds but any for Yorkshire, Northern, central office and office would.

The second and third pair of graphs concern the purchase costs of the offices. Examining the second pair, the left graph shows that if a purchase cost is higher then it cannot be lower and vice versa. The right graph shows that if one is false the other is true. The coreferent link in both cases establishes they refer to the same cost. These graphs can therefore be modelled independently for general use wherever needed.

Turning to the third pair of graphs, these infer that a central office is an office which has a higher purchase cost whilst local offices are offices with a lower purchase cost. This would occur should 'central office' or 'local offices' dominate these graphs respectively, so allowing the appropriate deiteration and double negation. Should the matching dominating concept contain a referent then these are passed to the relevant dominated graph before it is deiterated. The referent could then first be passed to its other coreferent linked concepts. Moreover the inference would also occur if any subtype of 'central office' or 'local offices' dominated. Hence the result should the graph:
exist would be:

\[(\text{office: Leeds}) \rightarrow (\text{characteristic}) \rightarrow \{'\text{lower purchase cost}'\}\].

The third pair of graphs reflect another aspect regarding the earlier argument about the degree of generality. Although the original statement, cognitive map, and COPE representation may only appear to refer to Leeds office and surrounding local offices, it may be more generally applicable than this. Conceptual graphs present the opportunity for the user to be aware of this just by their inherent structure: They are based on hierarchical type labels and specific instances within those labels, and the user would have to think about the appropriate degree of relevance. As the graphs stand they would apply to any local or general office. They may be written to infer about Yorkshire offices only, in which case 'central office' and 'local offices' in these graphs would instead read 'Yorkshire central office' and 'Yorkshire local offices' respectively. However given the available information it is not safe to include the third pair of graphs in the knowledge-base. Therefore the following restricted forms are added in their place:

\[
\{\{'\text{central office'}\}: \text{Leeds}\};
\{\{\text{office: Leeds}\} \rightarrow \{\text{characteristic} \rightarrow \{'\text{higher purchase cost}'\}\}\}.
\]

and

\[
\{\{'\text{local offices'}\}: \{'\text{Hull Sheffield and Harrogate}'\};
\{\{\text{offices: 'Hull Sheffield and Harrogate'}\} \rightarrow \{\text{characteristic} \rightarrow \{'\text{lower purchase cost}'\}\}\}.
\]

This prevents deiteration unless the referent is specifically 'Leeds' or 'Hull Sheffield and Harrogate'.

Continuing further, the concepts with undefined contrasting poles are modelled as follows:
The concepts 'greater delegation of authority' and 'follow current management initiatives' are not included above and will be dealt with after the discussion on COPE links below. Where the contrasting pole is not defined on input, COPE creates it by prefixing the term 'not' to the input emergent pole. This poses two questions. The first question asks if this is most accurately replicated in conceptual graphs by taking the input pole and encircling it with a negated context, as denoted above. An alternative might be to write, without this context, '[not <emergent pole>]'. Instead. Then the substitute graphs would be based on the form:

```
(['<emergent pole>', 'not <emergent pole>]').
```

Unlike their defined counterparts, it is evident the 'false-asserts-true' scenario is valid for undefined contrasting poles. Hence the following graph form would need to be added to the knowledge base for each alternative:

```
([[<emergent pole>]; ([not <emergent pole>]'))).
```

This is a superfluous addition. The graph '([<emergent pole>']) clearly equates to '[not <emergent pole>]'. The second question asks if there is any need to include such poles in conceptual graphs anyway. Taking the original graph:

```
([local representation]; ([local representation]).
```

as representative, this would merely assert '[local representation]' should '[local representation]' be true, and assert similarly '([local representation])' from '([local representation])'. This shows such concepts in fact turn out to be meaningless. Therefore they can be excluded from the conceptual graphs representation.
However it should be noted that COPE even treats all default contrasting poles as sufficiently accurate to attach poles logically in other COPE concepts to it. For instance, picking up on the 'mobile office' dimension discussed earlier, 'local representation. [not] local representation' leads to 'improve level of service. [not] improve level of service' respectively. COPE assumes the user accepts this by automatically asserting such poles. There is the distinct possibility that this presumption by COPE could be quite erroneous with the user being mislead by the model. It may be possible to have '[not] local representation' and 'improve level of service' through 'use mobile offices'. It would then be false that '[not] local representation' implies '[not] improve level of service'. The dubiety of COPE's premise is heightened somewhat when COPE itself suppresses such links in its graphical output form, as indicated from the previous section.

As explained earlier, implication in conceptual graphs are of the general form:

\[(\text{antecedent}); (\text{consequent})\] .

Therefore, after allowing for double negations, the COPE links may be modelled in conceptual graphs as follows:

![Diagram of conceptual graphs](image)

229
"increased running costs" • "higher admin costs"

"increased running costs" • "higher admin costs"

"local offices": 'Hull Sheffield and Harrogate' • "too much decentralisation"

"central office": Leeds • "too much decentralisation"

"too much decentralisation" • "impaired client treatment risk"

"too much decentralisation" • "ensure uniformity of treatment"

"lack of understanding about risk" • "use local office experience": #001

"lack of understanding about risk" • "use local office experience": #001
Without worrying about the graphs affected by double negation for the moment, each COPE link is explicitly stated in conceptual graphs by two 'implication' graphs, one for each pole to pole connection. As can be seen from the graphs, the 'leads from' pole becomes a concept which is enclosed in a negated context. This context also encloses another negated context which encloses the concept of the 'leads to' pole. The 'negative' link found in COPE becomes redundant because the order in which the poles are written are irrelevant in conceptual graphs. The concept '[:use local office experience:]' has been refined to '[:use local office experience':#001]' as it describes a particular office experience identified by the serial number '#001'.

As for the double negated graphs, the effect in the case of the graphs describing the false 'local representation', 'increased running costs', and 'too much decentralisation' implications of 'central office: Leeds' is they now appear to be like COPE concepts instead of COPE links. Hence these concepts show they have links that emerge to be additional contrasting poles. Conceptual graphs have yielded this fact explicitly and drawn it to the user's attention, whilst it remains unnecessarily implicit and thereby easily undetected in COPE. The graphs were arrived at, using 'local representation' as the illustration, by double negating from:

`(["local representation"]; (["central office": Leeds])).`

Of course, these graphs are based on the acceptance of COPE's previously mentioned dubious presumption that implications to and from default contrasting poles are always valid. If not, as indicated by 'use mobile offices' for example, then the user will become aware of it. This reveals another important area where conceptual graphs provide a much more focused model of the problem than COPE's attempt.

Apart from the graph:

`(("too much decentralisation"); ("ensure uniformity of treatment")).`

the effect of double negation results in the following form for the remaining cases:

`((<antecedent>); <consequent>).`

This apparent opposite to the usual form is derived because both the original 'antecedent' and 'consequent' graphs are false, as they are of the form '(<graph>)'. Hence substituting these graphs into their appropriate places in the general 'implication' graph causes the 'antecedent' simply to be a false graph and the 'consequent', through double negation, to be a true graph. The interesting item that emerges from this is that the resultant 'consequent' can imply its resultant 'antecedent'. For example, if 'higher admin costs' is true then so is 'increased running costs'. The '[not] too much decentralisation' leads to 'ensure uniformity of treatment' is a variation on the others, producing a 'false-asserts-true' graph form, because the consequent here is a defined contrasting pole. The significance of all this is that the conceptual graph representation explicates yet another fundamental insight drawn from the default contrasting poles that can be easily overlooked in the implicit-only COPE model.

The concept 'use local office experience' has some background information relating to it about the source of that information from some actual offices in the South West. This may best be described in the following graph:
This graph can be added to the knowledge base and then called upon as necessary. A reminder, out of interest, that if the earlier graphs ('*x' signifies coreferent links in the linear form):

\[
(['local offices': '*x];

(['offices: '*x']—>

(characteristic)—>[’lower purchase cost’])).
\]

had not been restricted to:

\[
(['local offices': 'Hull Sheffield and Harrogate'];

(['offices: 'Hull Sheffield and Harrogate']—>

(characteristic)—>[’lower purchase cost’])).
\]

then the following would be asserted from the background South West offices graphs:

[office: 'Plymouth Taunton Bath & others']—>

(characteristic)—>[’lower purchase cost’].

The above instance underlines how the powerful hierarchical nature of conceptual graphs can be employed, as brought out towards the beginning of this section.

At this point all the concepts and links have been discussed apart from 'greater delegation of authority', 'follow current management initiatives' and the relationship between 'lack of understanding about risk' and the choice of office. The first two concepts are modelled in the following graph:

\[
('current management initiative': #002 1 4 'authority style': delegate)
\]

This reflects that these two concepts are a statement of fact, whereas the previous concepts depend basically on which office type is chosen. Furthermore, adding the 'standard' link graph:

\[
(['authority style': delegate];

(['local offices': 'Hull Sheffield and Harrogate'])).
\]

would imply, from the above preference graph that 'local offices: Hull Sheffield Harrogate' would be true when it is really undecided. This truth also occurs in the graph:

['authority style': delegate]—>

(preference)—>[’local offices': 'Hull Sheffield and Harrogate'].

This can be avoided by making the statement apply to local offices in general. Although the following graph states there exist some local offices, it cannot purport that those local offices are in Hull, Sheffield and Harrogate:

['authority style': delegate]—>

(preference)—>[’local offices'].

. 232
It may be argued this is merely a convenient device on the part of conceptual graphs to get round this problem. Whilst this would not be denied, the inherent nature of conceptual graphs would demand eventually a more exacting analysis of the problem. Ensuing invalid inferences for instance would draw the user into finding out more about the situation or, if that is not possible, to apply expert judgement. The appropriate graphs could then be devised. In the above instance it may be wise to determine if delegation really must mean local offices. This can perhaps be provided somehow in a central office environment, or by mobile offices. Despite COPE's link, reference to the office text does not sufficiently clarify this. Therefore this link, even in the above 'convenient' form, cannot be included in the conceptual graphs knowledge base.

The above convenient argument also applies to the concept 'office' in the following graph that describes the 'lack of understanding about risk' relationship:

![Diagram showing conceptual graph for 'lack of understanding about risk' affecting choice.]

In COPE this was modelled as a connotative link which the above graph is intended to reflect. Even though there is an element of convenience in the graph, it is less ambiguous than the first one and therefore is included in the knowledge base. This, of course, is subject to appropriate revisions as discussed already. The existence of the 'lack of understanding about risk' graph would have an effect on the earlier graph:

\[
\text{('lack of understanding about risk'); ('use local office experience': #001)}.
\]

in that it would cause 'use local office experience': #001' to be asserted. In view of this this assertion might as well appear as a true graph to begin with. However, this is not done for two reasons. The first is to show that as new knowledge is added, this has a dynamic effect on the present knowledge. The second is that it shows a line of reasoning that the user may wish to know about.

One other general point worthy of discussion arises when a COPE concept has more than one link to it. This is whether all the 'leads from' poles have to be true, or just some of them, for that 'leads to' pole to be true. COPE does not bother to differentiate. In the COPE model the example has three concepts leading to 'centralise services at Leeds, open local offices'. As it happens, only one of these, 'lower purchase costs ...... higher purchase costs .......', ends up as an implication in the conceptual graphs model. However, if there had been more a decision would have to be made as to how many of them would need to be true. Taking three abstract concepts "A', 'B', 'C" as an example, the conceptual graphs would be:

\[
\text{('A'); ('B'); ('C').}
\]

should both 'A' and 'B' need to be true for 'C' to be asserted. The following graphs would assert 'C' if one of 'A' or 'B' was true (The other could be true, false or unknown):

\[
\text{('A'); ('C').}
\]

\[
\text{('B'); ('C').}
\]

Even though it does not arise in the 'offices' problem, the above 'and / or' distinction is yet another area where COPE may well mislead the user.

After allowing for all the above amendments, the knowledge base is as follows:
Appendix A

101

(['central office': Leeds]; ['local offices': 'Hull Sheffield and Harrogate']).

(['lower purchase cost': *x]; ['higher purchase cost': *x]).

(['lower purchase cost': *x]; ['higher purchase cost': *x]).

(['central office': Leeds]; ([office: Leeds]->
  (characteristic)->['higher purchase cost'])).

(['local offices': 'Hull Sheffield and Harrogate']; ([offices: 'Hull Sheffield and Harrogate']->
  (characteristic)->['lower purchase cost'])).

(['impaired client treatment']; ['ensure uniformity of treatment']).

(['local representation']; ([improved service level])).

('local representation'); ['improved service level']).

(['local offices': 'Hull Sheffield and Harrogate']; ['local representation']).

(['central office': Leeds]; ['local representation']).

(['local offices': 'Hull Sheffield and Harrogate']; ['increased running costs']).

(['central office': Leeds]; ['increased running costs']).

(['increased running costs']; ['higher admin costs']).

(['increased running costs']; ['higher admin costs']).

(['local offices': 'Hull Sheffield and Harrogate']; ['too much decentralisation']).

(['central office': Leeds]; ['too much decentralisation']).

(['too much decentralisation']; ['impaired client treatment risk']).

(['too much decentralisation']; ['ensure uniformity of treatment']).

(['lack of understanding about risk']; ['use local office experience': #001]).

(['lack of understanding about risk']; ['use local office experience': #001]).

['use local office experience': #001]->
  (source)->
  ['local offices': 'Plymouth Taunton Bath and others'].

['current management initiative': #002]->
  (preference)->['authority style': delegate].

['lack of understanding about risk']-
  ('affected choice')]->[office].

Now the knowledge base can be fully animated. This animation will show that, unlike conceptual graphs, the COPE model does not lend itself to proving false 'antecedents' from...
false 'consequents'. This is a logical operation known as *modus tollens*. Hence, for instance, if '((higher admin costs))' was an over-riding consideration then '((local offices'): 'Hull Sheffield and Harrogate') would be true. In other words, if higher administration costs must be avoided then there cannot be these local offices.

Concluding Remarks

In cognitive mapping it can be seen that more can be gleaned from concepts and their links than COPE. From the offices example the conceptual graphs representation has managed to reveal:

a) Like COPE, the concepts and relations in conceptual graphs can use the language that the decision maker identifies with.

b) Unlike COPE, the typed structure of conceptual graphs allows degrees of general knowledge to be applied to specific problem situations.

c) Through its bipolar limit, which conceptual graphs overcome, COPE could stifle creative thought by the decision maker.

d) COPE's default contrasting poles in concepts turn out to be meaningless.

e) The 'negative' link in COPE is irrelevant in conceptual graphs.

f) By always implicitly linking concepts with default contrasting poles COPE obfuscates the distinction between legitimate and potentially damaging relationships.

g) Conceptual graphs, on the other hand, removes this arbitrary situation by focusing the user's mind on what in fact are valid and invalid contrasting poles, including default ones.

h) Even though conceptual graphs may permit convenient devices to 'quick-fix' obstacles in the mapping exercise, they ultimately cause the user to improve the quality of the knowledge itself and refine the graphs accordingly.

i) COPE fails to distinguish between any 'and/or' relationships between knowledge elements.

Even though it may successfully elicit knowledge through its contrasting poles and links, the model of cognitive mapping embodied in the COPE software fails to extract properly the genuine impact of these relationships. Moreover it can be wrong, as the references to 'use mobile offices' have revealed for example. A conceptual graphs processor, unlike COPE, could automatically recognise and deal with the contrasting aspects of Eden's technique. This would occur as a direct part of the negated contexts upon which conceptual graphs inference is based. As well as inference, the processor would also be able to check for any inconsistencies as they are entered into the knowledge-base. All this should free the user to declare merely what he or she believes and then review that mental model, or its computer paradigm, in the light of the processor's output.

This discussion started to bring out the other facets of conceptual graphs theory such as the type label hierarchy, referents and conceptual relations. There is no doubt that the above 'final' conceptual graph knowledge base could be refined much further. Nevertheless this paper has achieved what it was intended to do: It has revealed the technical limit of Eden's cognitive maps have been reached whereas the technical aspects of cognitive mapping in conceptual graphs are barely embraced, and hence demonstrates ample evidence that conceptual graphs is by far the more expressive paradigm of knowledge.
References


A/02: Pilot Study Instructions I

From: Simon Polovina

Wednesday, March 11, 1992

Instructions

Please read and try to understand the problem in the attached paper before tomorrow (Thursday @ 2pm). The things to pay particular attention to are:

1) Where the conceptual graphs model improves upon COPE.

2) The structure of the conceptual graph knowledge-base at the end of the paper. Try to understand the relationships between the items within it, and note where you feel it may be deficient.

On Thursday we will be taking this knowledge-base and:

1) I will demonstrate how to query the knowledge-base. For instance I may assert the decision to have a central office in Leeds or to improve quality of service and see what the consequences are.

2) You will then query the existing knowledge-base and evaluate what new things you find out or what is wrong with the current model.

3) After this you will attempt to modify the knowledge base in the light of what you feel may be missing or wrong. You may decide to use your judgement to add, refine or remove graphs from the existing knowledge-base deriving from how you interpret the original problem.

4) Lastly, you will try and refine the knowledge-base to consider some new dimension to the problem. In particular this will relate to introducing the mobile office option into the knowledge-base.

At the end I hope you will write down what you thought, what you got and what you think you can get out of using conceptual graphs. I will issue more detailed instructions (eg how to use the conceptual graphs processor) on Thursday.
From: Simon Polovina

Thursday, March 12, 1992

Conceptual Graphs Pilot Study: Instructions and Tutorial

Loading and preparing the conceptual graphs processor (cgp)

To invoke the conceptual graphs processor type:

cgp

at the unix prompt.

This will give the cgp prompt:

>>

At this prompt type

%officescript.

Ensure you include the full stop at the end! (In fact all commands made to cgp must end in a full stop). The '%' command starts to execute the file 'officescript' which contains a series of commands to initialise cgp and load the file 'officesfile' which has the existing conceptual graphs knowledge-base on the office location problem. Executing 'officescript' produces the following output (plus my explanatory annotations):

Reading script officescript...

officescript:>> ~.

The 'filename:>>' shows cgp is taking input from that file rather than the terminal. The '~.' clears cgp of any previous knowledge-base, although this is not significant here because cgp always starts up empty.

all retracted

Shows knowledge-base is cleared.

officescript:>> p.

evaluation = partial

Ignore this for now.
officescript:>> v.
report = brief

This command toggles cg to only supply a brief explanation of queries subsequently made to it. The output 'report = brief' confirms this. The alternative verbose explanation is examined later.

officescript:>> <officesfile.

Loading officesfile...
The '<' command loads 'officesfile'.

OK

The file 'officesfile' is now loaded.

officescript:>> *.

This command displays the contents of the knowledge-base as follows:

{ ['local offices':'Hull Sheffield and Harrogate'];
  ['central office':Leeds]
}.

{ ( [office:Leeds]-
    (characteristic)->['higher purchase cost':*0]
  );
  ['central office':Leeds]
}.

{ ( [offices:'Hull Sheffield and Harrogate']-
    (characteristic)->['lower purchase cost':*1]
  );
  ['local offices':'Hull Sheffield and Harrogate']
}.

{ ( ['higher purchase cost':*2]
  );
  ['lower purchase cost':*2]
}.

• 239
<table>
<thead>
<tr>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Higher purchase cost</td>
<td></td>
</tr>
<tr>
<td>Lower purchase cost</td>
<td></td>
</tr>
<tr>
<td>Ensure uniformity of treatment</td>
<td></td>
</tr>
<tr>
<td>Impaired client treatment</td>
<td></td>
</tr>
<tr>
<td>Improved service level</td>
<td></td>
</tr>
<tr>
<td>Local representation</td>
<td></td>
</tr>
<tr>
<td>Hull, Sheffield, and Harrogate</td>
<td></td>
</tr>
<tr>
<td>Central office: Leeds</td>
<td></td>
</tr>
<tr>
<td>Increased running costs</td>
<td>Hull, Sheffield, and Harrogate</td>
</tr>
<tr>
<td>Central office: Leeds</td>
<td></td>
</tr>
<tr>
<td>Higher admin costs</td>
<td></td>
</tr>
<tr>
<td>Increased running costs</td>
<td></td>
</tr>
</tbody>
</table>
Appendix A/03

('higher admin costs': *16);
  ('increased running costs': *17)
).

('too much decentralisation': *18);
['local offices': 'Hull Sheffield and Harrogate']
).

('too much decentralisation': *19);
['central office': 'Leeds']
).

('impaired client treatment risk': *20);
['too much decentralisation': *21]
).

('ensure uniformity of treatment': *22);
('too much decentralisation': *23)
).

('use local office experience': *1]
)['lack of understanding about risk': *24]
).

('use local office experience': *1];
  ('lack of understanding about risk': *25)
).

['use local office experience': *1]-
  (source)->
['local offices': 'Plymouth Taunton Bath and others'].

['current management initiative': *2]-
  (preference)->['authority style': 'delegate'].

241
These are the graphs themselves. They should be mostly self-evident from reading the office location paper given to you yesterday. Concepts without referents are given '*sequential number' default referent. There is no need to be concerned about this other than to note when two or more of these numbers are the same then those referents are coreferent. This would occur where an input graph had concepts with the same '*x' referent, as discussed in yesterday's paper.

(remainder of output)

The remaining output can be ignored for the purposes of this pilot study.

OK

This signifies the end of the script. The prompt returns to the normal:

Adding new knowledge

New graphs may be entered as follows:

>> ![central office]: Leeds]

The '!' command causes the graph that follows it to be added into the knowledge-base (ensuring the full stop has been added at the end - please do not use either full stops or commas in the middle of a graph!). Here, it has been decided to have the central office in Leeds and so this is added as a true graph. It is also possible to assert graphs which include negating contexts, such as false graphs and implications. Spaces in all input graphs are not significant unless they appear within quotes. In the above graph the space in "central office" is significant whereas in ": Leeds" it is not (i.e. ":Leeds" is the same).

Thinking...

I did not know that:
['central office': Leeds].

Tidying up...

Conformity relation added:
'central office' :: Leeds.

This output from cgp shows it has checked the input for consistency and discovered it did not know this fact already.
Querying the knowledge-base

The truth of any graph may be queried as follows:

>> ?['improved service level'].

Here a '?' is used in place of a '!'. The output from cgp is:

Thinking...

The statement:
  ['improved service level':*31].
is FALSE because:
  fact: ['central office':Leeds].

Similarly query graphs in negating contexts may be entered:

>> ?(['improved service level']).
Thinking...

The statement:
  ( ['improved service level':*33]
  ),
is TRUE because:
  fact: ['central office':Leeds].

In both these cases cgp has worked out the truth of an improved service level consequent to deciding what if the decision is to have a central office at Leeds.

If a more comprehensive explanation is required, this may be obtained by the toggle command:

>> v.

which is confirmed by the output:

report = verbose

Hence re-querying the improved service level as follows:

>> ?['improved service level'].
Thinking...

The statement:
  ['improved service level':*35].
is FALSE because:
known rule:
   ( ['improved service level':*8];
      ['local representation':*9]
   )
);

so:
   ( ['improved service level':*31];
      ['local representation':*9]
   )
);

proof of antecedent:
   ( ['local representation':*9]
   )
);

is:
known rule:
   ( ['local representation':*11];
      ['central office':Leeds]
   )
);

so:
   ( ['local representation':*9];
      ['central office':Leeds]
   )
);

proof of antecedent:
   ['central office':Leeds];

is:

fact:
   ['central office':Leeds];
result:
   ( ['local representation':*9]
   )
);

result:
   ( ['improved service level':*31] )
).

Similarly:

>> ?(['improved service level']).
Thinking...
The statement:

```javascript
('improved service level': 37)
```

is TRUE because:

*(same explanation as false case above)*

**Other commands**

To save the contents of the knowledge-base as it currently stands, the '>' command is used, with a file name, as follows:

```plaintext
>> >polovina.
```

The output is:

```
Saving polovina...
OK
```

*(NB: I have given the file name as my surname. I strongly recommend you use your own surname. This is to avoid overwriting each other's work by using 'common' file names should we all be working in the same directory! Please also ensure you do not get mixed up between this and the '<' command which loads files, otherwise you will overwrite rather than load a file and vice versa!)*

The following should by now be self-explanatory:

```plaintext
>> ~.
all retracted
```

```plaintext
>> <polovina.
Loading polovina...
OK
```

```plaintext
>> *:
```

*(displays knowledge-base)*

It is also possible to retract particular graphs with the '～' command by specifying the graph after it. For instance:

```plaintext
>> ～['central office': Leeds].
gives:
retracted:
['central office': Leeds].
```
not known:
['central office':Leeds].

The following removes an implication:

>> ~(['increased running costs'];(['higher admin costs'])).

retracted:


not known:


The '*.' command will show the modified knowledge-base.

Lastly, the following command quits cgp (The knowledge-base is not saved. If required save it by '>' command first):

>> ^.

Summary of commands

All commands must end with a full stop!

%filename. Takes commands from a script rather than the terminal (All scripts must finish with the term "end_of_file.").

<filename. Load a knowledge-base.

>filename. Save the current state of the knowledge-base.

*.. List the current state of the knowledge-base.

!graph. Add graph to the knowledge-base (Do not use full stops or commas in middle of graphs. This also applies to the other commands which include graphs).

~graph. Retract graph from the knowledge-base. Omitting a graph retracts the whole knowledge-base.
Appendix A103

?graph. Query the truth of \textit{graph} in the knowledge-base.

v. Toggle between brief and verbose explanations of queried graphs.

^ Quit cgp (Knowledge-base not saved unless previously saved by ' > ' command).

Ask me if you wish to use hierarchies (i.e. \textit{Subtype} < \textit{Supertype}).
/*
  -----------------------------BROWSING-----------------------------
*/

/*
find_string_- finds a given text string (called from menu handling).

find_string_in_output - also used to find again a given text string (called from menu handling).
*/

find_string_(Dialog,1,String):- % finds the string, if it exists, in drawing sheets
  remember(last_string,[String,0]),
  find_string_in_graphs. % see below

find_string_(Dialog,3,String):- % finds the string, if it exists, in jehcgp knowledge-base
  remember(last_string,[String,0]),
  concat("\",String,CommandedString),
  stringof(CSChars,CommandedString),
  special_command(CSChars). % see engine additions

find_string_(Dialog,4,String):- % finds the string, if it exists, in jehcgp output window
  remember(last_string,[String,0]),
  find_string_in_output.

find_string_(Dialog,_,String):- % string not found ('1' and '4' above only, '3' has own in jehcgp)
  remember(last_string,[String,0]),
  message([String,'could not be found. Sorry!']).

find_string_in_graphs:- % finds the string, if it exists, in drawing sheets
  windows(1),
  remove('Z Graphic Window',Windows,Windows1), % exclude default LPA graphic window
  find_string_in_graphs1(Windows1).

find_string_in_graphs1([]):-
  recall(last_string,[String,Start]),
  !.
  Start \= 0. % fails if search exhausted and String not found

find_string_in_graphs1([FirstWindow|OtherWindows]):-
  desel_all(FirstWindow),
  get_all_pics(FirstWindow,Names),
  find_string_in_graphs2(FirstWindow,Names),
  !,
  find_string_in_graphs1(OtherWindows).

find_string_in_graphs1([]):- !. % still succeeds when 'no' selected in 'Continue search?' below

find_string_in_graphs2([]).
find_string_in_graphs2(FirstWindow,[FirstName|OtherNames]):- % finds, if exists, in window
get_pic(FirstWindow,FirstName,Description),
  (Description = concept_rectangle(Text,_______),
   Description = relation_oval(Text,_______)),
stringof(TextString,Text),
recall([last_string,(SearchText,_)],
stringof([SearchTextString,SearchText],
  l,
  ([sublist(SearchTextString,TextString), % see general routines
    wfront(FirstWindow),
    sel_pics(FirstWindow,[FirstName]),
    remember([last_string,(SearchText,-1)], % prevents 'not found' message (see above)
    l,
    yesno([SearchText,'found in front drawing sheet. Continue search?']),
    desel_pics(FirstWindow,[FirstName]);
    true),
  l,
find_string_in_graphs2(FirstWindow,OtherNames).)
find_string_in_graphs2(FirstWindow,[FirstName|OtherNames]):- % not conrel
  l,
find_string_in_graphs2(FirstWindow,OtherNames).
find_string_in_output:- % finds the string, if it exists, in jehcgp output window
  recall(jehcgp_output,Window),
  recall([last_string,(String,Start)],
  wsearch(Window,String,Start,From,To),
cursor(Window,From,To),
wscroll(Window),
wfront(Window),
remember([last_string,(String,To)]).
find_string_in_output:-
  recall([last_string,(String,Start)],
  Start \= 0, % fails if search exhausted and String not found
  remember([last_string,(String,0)], % resets to start again
  l,
find_string_in_output.
/*
retrieve_labels - retrieves all the current labels in jehcgp's knowledge base.
*/
retrieve_labels(FinalLabelList1):-
  retrieve_labels1(1,[],FinalLabelList),
  (recall(a2z_labels,enabled),
sort(FinalLabelList,FinalLabelList1)); % sorts list if 'A to Z' in 'Labels' menu is marked
FinalLabelList1 = FinalLabelList
.
249
retrieve_labels1(N,LabelList,FinalLabelList):-
  clauseX(lex(LexLabelList,Integer),true,N,N),  % lex signifies a jehcgp clause

  ((switch(easy),
    (Integer >= 0,
     append(LabelList,LexLabelList,NewLabelList);
     true));  % excludes jehcgp's global labels in easy mode
     append(LabelList,LexLabelList,NewLabelList)),

    N1 is N + 1,
    !,
    retrieve_labels1(N1,NewLabelList,FinalLabelList).

retrieve_labels1(_,FinalLabelList,FinalLabelList).
/- CONCEPTS -/

new_concept: the '[t:i]' tool - draws new concepts

new_concept_(activate,Window):-
  desel_all(Window),
  gcursor(Window,cross_hair).

new_concept_(deactivate,Window):-
  gcursor(Window,arrow).

new_concept_(Window,Y,X,Mode):- % edits an existing concept (old concept ends up being replaced)
  find_pic(Window, (Y,X), Name, Description, (Ya,Xa)),
  Description = concept_rectangle(Text,Font,Face,Size,Top,Left),
  (recall(encumbrances,retain); % leave attached contexts/links (may result in 'untidy' contexts)
   remove_its_encumbrances(Window,Name)), % otherwise remove (see general routines)
  del_pic(Window,Name),
  set_gfont(Window,Font,Face,Size),
  Top1 is Top + Ya,
  Left1 is Left + Xa,
  forget(edit_mode), % ensures edit functions are reset to default (see menu handling),
  set_prop(conrel_name,Window,Name), % retain name if needed for attached contexts/links
  menus(deactivate,Window), % use gactdeact 'menus' program
  edit_box(Window,box(Top1,Left1,Size,400),0,Text).

new_concept_(Window,Y,X,Mode):- % creates a new concept provided it is properly formed
  marqui(Window, (Y,X), Rectangle),
  concept_overlap(Window,Rectangle).

concept_overlap(Window,box(Top,Left,_,Width)):-
  wfont(Window,_,_,Size),
  check_overlap(Window,box(Top,Left,Size,Width)),% see general routines
  forget(edit_mode), % ensures edit functions are reset to default (see menu handling)
  unique_graph_gensym(Window,concept,NewName), % obtain NewName (see general routines)
  set_prop(conrel_name,Window,NewName), % used because of edit existing concepts case above
  menus(deactivate,Window), % use gactdeact 'menus' program
  edit_box(Window,box(Top,Left,Size,Width),0).

concept_overlap(_,_). % do nothing as concept was improperly formed

new_concept_(close_edit,Window):-
  get_text(Window,Details),
  design_new_concept_(Details,Description),
  get_prop(conrel_name,Window,NewName), % gives edited concept its name
  del_prop(conrel_name,Window),
  get_all_pics(Window,NonNewNames),
  add_pic(Window,NewName,Description),
  reposition_links(Window,[NewName],NonNewNames), % see general tools
  menus(activate,Window), % use gactdeact 'menus' program
  remember(edit_mode,drawing). % ensure edit functions are set to drawing (see menu handling)
design_new_concept_(text(Font,Face,Size,box(Top,Left,...),Text),
    concept_rectangle(Text,Font,Face,Size,Top,Left)).

concept_rectangle(Text,Font,Face,Size,Top,Left,Description):-
    text_width(Text,Font,Face,Size,Width),
    Top1 is Top + 2,
    Left1 is Left + 2,
    inset_box(box(Top1,Left1,Size,Width),(-3,-3),box(Top2,Left2,Size1,Width1)),
    Description = [blank(fillbox(Top2,Left2,Size1,Width1)),
                    textline(Font,Size,Face,Top,Left1,Text)].
appendix A104

/*

------------------------DIRECTED LINKS------------------------

*/

new_directed_link(activate, Window): -
desel_all(Window),
gcursor(Window, cross_hair).

new_directed_link(deactivate, Window): -
gcursor(Window, arrow).

new_directed_link(Window, Y, X, Mode): -
find_pic(Window, Y, X, Name, Description, (Ya, Xa)),
Description = concept_rectangle(Ya, Xa).
1,
((sel_pics(Window, [Name]),
includes_contexts(Window, [Name]), % see general routines
get_sel_pics(Window, Names),
Names = [Name], % ensures link cannot be drawn from a concept inside a negated context
rubber_band(Window, Y, X, NewPoint),
continue_rubber_band(Window, Name, trans(Ya, Xa, Description), NewPoint),
desel_pics(Window, [Name])). % otherwise deselect graph and don't continue

continue_rubber_band(Window, ConceptName, ConceptTransDescription, NewPoint): -
find_pic(Window, NewPoint, Name, Description, (Y, X)),
Description = relation_oval(Y, X),
sel_pics(Window, [Name]),
append([ConceptName], [Name], FromConRelNames),
1,
((not relations_all_linked(Window, [Name]), % relation does not already have a directed link
append([ConceptTransDescription], [trans(Y, X, Description)], FromConRelDescriptions),
rubber_band(Window, NewPoint, NewPoint1),
continue_rubber_band1(Window, FromConRelNames, FromConRelDescriptions, NewPoint1),
desel_pics(Window, FromConRelNames)). % otherwise deselect graphs and don't continue

continue_rubber_band1(Window, FromConRelNames, FromConRelDescriptions, NewPoint1): -
find_pic(Window, NewPoint1, Name, Description, (Y, X)),
Description = concept_rectangle(Ya, Xa),
sel_pics(Window, [Name]),
append(FromConRelNames, [Name], ConRelNames),
1,
((not on(Name, FromConRelNames), % user has not linked back to original concept
includes_contexts(Window, [Name]), % see general routines
get_sel_pics(Window, Names),
on(Name, Names), % ensures link cannot be drawn to a concept inside a negated context
append(FromConRelDescriptions, [trans(Y, X, Description)], ConRelDescriptions),
design_directed_link(ConRelNames, ConRelDescriptions, LinkDescription),
unique_graph_gensym(Window, link, NewName), % see general routines
add_pic(Window, NewName, LinkDescription),
bring_to_front(Window, ConRelNames),
desel_pics(Window, ConRelNames);
desel_pics(Window, ConRelNames)). % otherwise just deselect graphs

. 253
design_directed_link([FromConName, RelName, ToConName],
  [FromConDescription, RelDescription, ToConDescription], Description):
  find_centres([[FromConDescription, RelDescription, ToConDescription], []],
    [Point1, Point2, Point3]),  % see below (draws lines between conrel centres)
  pic_frame(RelDescription, RelFrame),
  define_arrow(RelFrame, Point1, Point2, Arrow1),  % see below (draws first arrow)
  pic_frame(ToConDescription, ToConFrame),
  define_arrow(ToConFrame, Point2, Point3, Arrow2),  % see below (draws second arrow)
  Description =

directed_link([FromConName, RelName, ToConName], [Point1, Point2, Point3], Arrow1, Arrow2).

find_centres([], Centres, Centres).

find_centres([FirstDescription|OtherDescriptions], SomeCentres, AllCentres):
  pic_centre(FirstDescription, Centre),
  append(SomeCentres, [Centre], SomeCentres1),
  find_centres(OtherDescriptions, SomeCentres1, AllCentres).

directed_link([ConRelNames, Centres, Arrow1, Arrow2],
  [lines(Centres), fillpoly(Arrow1), fillpoly(Arrow2)]).

/*
define_arrow - defines where the arrowheads should be on the directed links (thanks to Mark
Sofroniou for providing the basis of the calculations that determine (Yb,Xb) and (Yc,Xc) given
(Ya,Xa)).*/

define_arrow(Frame, Point1, Point2, [ArrowPoint1, ArrowPoint2, ArrowPoint3]):
  define_arrow1(Frame, Point1, Point2, 1, ExactArrowPoint1, ArrowPoint1),
  define_arrow2(Point1, Point2, ExactArrowPoint1, ArrowPoint2, ArrowPoint3).

% define_arrow1 - arrow is visible such that it is approximately just outside its lead to conrel:

define_arrow1(Frame, (Y1, X1), (Y2, X2), N, ExactArrowPoint1, ArrowPoint1):-
  Na is 1 - N,
  Y3 is (Y1 * Na) + (Y2 * N),
  X3 is (X1 * Na) + (X2 * N),
  int(Y3, Ya),
  int(X3, Xa),

  ((pi_in_box((Ya, Xa), Frame),
    N1 is N - 0.01,
    1,  % still inside so try again
    define_arrow1(Frame, (Y1, X1), (Y2, X2), N1, ExactArrowPoint1, ArrowPoint1));
  (ExactArrowPoint1 = (Y3, X3),
    ArrowPoint1 = (Ya, Xa)).  % now outside so accept
define arrow2((Yl, Xl), (Y2, X2), (Ya, Xa), (Yi, Xi), (Yii, Xii)) :-
Horizontal is X2 - X1,
Vertical is Y2 - Y1,

((abs(Horizontal, Abs),
 Abs >= 1,
Tangent is (Vertical/Horizontal),
tan(Angle, Tangent));
(sign(Vertical, Sign),
Angle is Sign * pi/2)), % get Angle

((Horizontal < 0,
Angle1 is Angle - pi);
Angle1 is Angle),

cos(Angle1, Cosine), % get Cosine
sin(Angle1, Sine), % get Sine
Hd is 12, % Hd is depth of arrow
Hw is Hd/4, % Hw is the width of the arrow
Yb is Ya - (Hd * Sine) - (Hw * Cosine),
Xb is Xa - (Hd * Cosine) + (Hw * Sine), % (Yb, Xb) is second corner of arrow
Yc is Ya - (Hd * Sine) + (Hw * Cosine),
Xc is Xa - (Hd * Cosine) - (Hw * Sine), % (Yc, Xc) is third corner of arrow
int(Yb, Yi),
int(Xb, Xi),
int(Yc,Yii),
int(Xc, Xii).
/*
 ***************************************** DRAWING WINDOWS *****************************************
 */

/*
load_drawing_sheets_only(Filename) - loads only the graph drawings that are in Filename.
*/
load_drawing_sheets_only(Filename):-
    read(Filename,graph_drawings(Window,Names,Descriptions)),
    assert(graph_drawings(Window,Names,Descriptions)),
    load_drawing_sheets_only(Filename).

load_drawing_sheets_only(Filename):-
    read(Filename,end_of_file),!.

load_drawing_sheets_only(Filename):
    !,
    load_drawing_sheets_only(Filename).

/*
open_drawing_sheets - opens file-saved windows and their graphs.
*/
open_drawing_sheets:-
    retract(graph_drawings(Window,Names,Descriptions)),
    process_graph_drawings(Window,Names,Descriptions),
    !,
    open_drawing_sheets.

open_drawing_sheets.

process_graph_drawings(Window,Names,Descriptions):-
    avoid_window_conflict(Window,Window1),
    create_drawing_sheet1(Window1), % see start
    process_graph_drawings1(Window1,Names,Descriptions).

avoid_window_conflict(Window,Window2):-
    is_win(Window,...),
    concat(Window,"*",Window1),
    !,
    avoid_window_conflict(Window1,Window2).

avoid_window_conflict(Window,Window).

process_graph_drawings1(_,[],[]).

process_graph_drawings1( Window,[FirstName|OtherNames], [trans(Y,X,Desc)|OtherDescs]):-
    record_pic(Window,FirstName,Desc,(Y,X)),
    (!,(Desc = directed_link(_,_,_,_);
        Desc = negating_context(_,_,_,_),
        send_to_back(Window,[FirstName])); % links and contexts are in their proper place
    true),
    !,
    process_graph_drawings1( Window,OtherNames,OtherDescs).
/**
save_drawing_sheets - writes windows and their graphs into a file.
**/

save_drawing_sheets(Filename):-
windows(grap,Windows),
remove('Σ Graphic Window',Windows,Windows1), % exclude default LPA graphic window
drawing_windows_details(Filename,Windows1).

drawing_windows_details(Filename,[]).

drawing_windows_details(Filename,[FirstWindow|OtherWindows]):-
get_all_pics(FirstWindow,Names),
get_all_descriptions(FirstWindow,Names,Descriptions),
writeq(Filename,graph_drawings(FirstWindow,Names,Descriptions)),
writeq(Filename,''),
1,
drawing_windows_details(Filename,OtherWindows).

get_all_descriptions(Window,Names,Descriptions):-
get_all_descriptions1(Window,Names,[],Descriptions).

get_all_descriptions1(_,[],Descriptions,Descriptions).

get_all_descriptions1(Window,[FirstName|OtherNames],Descriptions,Descriptions2):-
get_pic(Window,FirstName,Description,(Y,X)),
append(Descriptions,[trans(Y,X,Description)],Descriptions1),
1,
get_all_descriptions1(Window,OtherNames,Descriptions1,Descriptions2).
/*
-------------------------------------------ENGINE ADDITIONS-------------------------------------------

[Makes refinements to John Heaton's cgp, or 'jehcgp' (other than those made in jehcgp itself)].
*/

starttext:-  
'<jehcgp_clause(s)>(succeed),
remember(jehcgp_output,'cgp - status report'), % window name is defined here
create_text_window,
(switch(easy);
assert(switch(easy))). % ensures jehcgp always starts off in easy output mode

create_text_window:-
recall(jehcgp_output,Window),
create_text_window1(Window),
tell(Window),
unknown(_, fail).

create_text_window1(Window):-  
(is_win(Window_);
wccreate(Window,1,190,3,149,506)).

/*
command - works in lieu of jehcgp's 'read_command', which specifically handles linear form graphs input on command line processors (LPA requires use of dialog boxes anyway). Also gives time command started and ended. NB: 'read_command' is not actually replaced because it is also used for reading in scripts.

read_data - similar to 'command' above, but actually replaces same command in jehcgp. Use arises when jehcgp requests user to respond by giving some defining graph for super / sub types, canonical relations
*/

command(Com,Data):-  
jehcgp_window, % see below
current_time(Time), % see below
nl,
writeseqnl(['Command started at',Time]),

((recall(new_labels,NewLabelList),  
write_new_labels(NewLabelList), % see below
forget(new_labels));
true),

((recall(new_referents,NewRefList),  
write_new_referents(NewRefList), % see below
forget(new_referents));
true),
extra_command_preperation(Data,Data1), % see after 'read_data' below  
'<jehcgp_clause(s)>(succeed),
(recall(jehcgp_interactives,enabled); % established in menu handling
'<jehcgp_clause(s)>(succeed)),
!,
(special_execute(Com,Data1); % see below
(!,  
execute(Com,Data1))), % enter jehcgp (always succeeds)
!,
% (continued overleaf)
% (continued from previous page)

(recall(jehcg_interactives, enabled); % established in menu handling
'<jehcg_clause(s)>(succeed)),

current_time(Time1), % see below
nl,
writeseqn(['Command completed at', Time1]).

read_data(Data):-
unique_window_gen_sym('Definition ', Window), % see general routines
create_drawing_sheet2(Window), % see start
wsize(Window, __, 187, __), % reduce Window depth so can see jehcg's request etc.
suspending_dialog(Window), % see immediately below
get_all_pics(Window, Names), % see start
graph_input_0(Window, Names, InternalGraph), % see input
jehcg_window, % see below
extra_command_preperation(InternalGraph, Data), % see immediately below
write_seqnl(['Window, 'entered as:']),
print_data(Data), % in jehcg (always succeeds)
(extra = [],
wkill(Window)); % don't keep window if nothing was entered
add_remaining_tools(Window), % otherwise do this (see start)
)

extra_command_preperation(Data, Data4):- % also in 'read_command' (and jehcg's 'read_data' '
'<jehcg_clause(s)>(succeed).

suspending_dialog(Window):- % effectively suspends execution whilst user draws defining
graph
current_time(Time), % see below
dialog('Defining structure request', 240, 10, 48, 490, % see immediately below
    [button([14, 440, 20, 40], 'Ok'),
     text([8, 10, 32, 420, wseq([Please draw the appropriate structure on', Window, 'drawing sheet if you wish. Click "Ok" to
     continue.
    ])], Button),
     current_time(Time1), % see below
     writeseqn(['Time when questioned:', Time1]),
     writeseqn(['Time when answered:', Time1]).
*

special_execute - used for executions not supported by jehcg.
*/

special_execute(retract,[conf(X, Y)]):- % retract the episode, if exists, with this conformity
'<jehcg_clause(s)>(succeed_or_fail),
retract_output([conf(X, Y)]). % see immediately below

special_execute(retract,[lattice(X, Y)]):- % retract the episode, if exists, with this lattice
'<jehcg_clause(s)>(succeed_or_fail),
retract_output([lattice(X, Y)]). % see immediately below

special_execute(retract,Data):- % retract all episodes, if exists, that have the graph Data
'<jehcg_clause(s)>(succeed_or_fail),
retract_output(Data)). % see immediately below
special_execute(retract, Data):-
    nl,
    writeln("Sorry, but the episode: "),
    print_data(Data), % in jehcgp (always succeeds)
    nl,
    writeln("could not be retracted as it does NOT exist within my knowledge-base.").

retract_output(Data):-
    nl,
    writeln("Retracted the episode: "),
    print_data(Data), % in jehcgp (always succeeds)
    nl,
    writeln("Please consider overriding my old knowledge:").

/*
special_command - also works in lieu of jeh's 'read_command' similar to above, except this caters for those commands that don't fit into 'command' above.
*/
special_command(*):- % jehcgp knowledge base listing command
    jehcgp_window, % see below
    % NB: the following jehcgp clauses exclude those that only give internal info:
    '<jehcgp_clause(s)>(succeed),
    current_time(Time), % see below
    nl,
    wriiteqnl(["Command completed at", Time]).

special_command(\):% jehcgp retract and reset whole knowledge-base command
    jehcgp_window, % see below
    current_time(Time), % see below
    nl,
    writeqnl(["Command started at", Time]),
    execute(\,['\',\]], % to jehcgp (always succeeds)
    current_time(Time1), % see below
    nl,
    writeqnl(["Command completed at", Time1]).

special_command(<):- % jehcgp load knowledge-base command (also loads drawing windows)
    dvol(Volume),
    old("TEXT",Filename,Volume1),
    jehcgp_window, % see below
    open(Filename,Volume1),
    dvol(Volume1),
    load_data(Filename), % in jehcgp (always succeeds)
    close(Filename),
    dvol(Volume),
    current_time(Time), % see below
    nl,
    writeqnl(["Command completed at", Time]),
    open_drawing_sheets, % see drawing windows
    cleanup(grap,41,3,20,10).
special_command(>): % jehcgp save knowledge-base command (also saves drawing windows)
   dvol(Volume),
   new(Filename,Volume1,'Save current state as:'),
   jehcgp_window, % see below
   create(Filename,Volume1,'TEXT'),
   open(Filename,Volume1),
   dvol(Volume1),
   1,
   save_drawing_sheets(Filename), % see drawing windows
   1,
   save_data(Filename), % in jehcgp (always succeeds)
   close(Filename),
   dvol(Volume),
   current_time(Time), % see below
   nl,
   write(seqnl('Command completed at',Time)).

special_command(build_model): % rebuilds knowledge-base
   jehcgp_window, % see below
   current_time(Time), % see below
   nl,
   write(seqnl('Command started at',Time)),
   1,
   execute('%', [%', build_model]), % to jehcgp (always succeeds)
   current_time(Time1), % see below
   nl,
   write(seqnl('Command completed at',Time1)).

special_command('%'): % jehcgp execute a linear form script command
   dvol(Volume),
   old('TEXT',Filename,Volume1),
   jehcgp_window, % see below
   open(Filename,Volume1),
   dvol(Volume1),
   1,
   execute('%', [%', Filename]), % to jehcgp (always succeeds)
   close(Filename),
   dvol(Volume),
   current_time(Time), % see below
   nl,
   write(seqnl('Command completed at',Time)).

special_command([Command]): % precludes these jehcgp commands:
   on(Command,[<,>,, ^,%,"",',"","",',""],
   message([The',Command,'command cannot be entered this way in CARE.']).

special_command([Command]): % executes those single character commands that jehcgp expects to be ASCII:
   not on(Command,["",'-'], % further excludes these jehcgp non-ASCII commands
   name(Command,[Number]),
   jehcgp_window, % see below
   1,
   execute(Number,[]), % to jehcgp (always succeeds)
   current_time(Time), % see below
   nl,
   write(seqnl('Command completed at',Time)).
special_command([Command]): - % executes remaining single character jehcgp commands:
jehcgp_window, % see below
!,
execute(Command,[Command,[]], % to jehcgp (always succeeds)
current_time(Time), % see below
nl,
write_seqnl(['Command completed at',Time]).

special_command([Command|RestComList]): - % executes remaining jehcgp commands (which may support an argument):
jehcgp_window, % see below
!,
stringof(RestComList,Arguments),
execute(Command,[Command,Arguments], % to jehcgp (always succeeds)
current_time(Time), % see below
nl,
write_seqnl(['Command completed at',Time]).

write_new_labels - outputs new type and relation labels from input graphs.
*/
write_new_labels(NewLabelList):-
   writeln('ZMNew type or relation labels: -M'),
   write_new_labels1(NewLabelList).

write_new_labels1([]).  
write_new_labels1([FirstNewLabel|Rest]):-
   write (FirstNewLabel),
   writeln('.'),
   !,
   write_new_labels1(Rest).

write_new_referents - outputs new referents from input graphs.
*/
write_new_referents(NewLabelList):-
   writeln('ZMNew referents: -M'),
   write_new_referents1(NewLabelList).

write_new_referents1([]).  
write_new_referents1([FirstNewRef|Rest]):-
   write (FirstNewRef),
   writeln('.'),
   !,
   write_new_referents1(Rest).
 Appendix A/04

jehcgp_window - prepares text window (defined in engine additions) for jehcgp output etc.

jehcgp_window:-
  recall(jehcgp_output,Window),
  create_text_window1(Window), % see engine additions
  wfront(Window),
  cursor(Window,-1,-1). % ensure cursor is below any existing text
cursor(watch).

*/

yesno_more - replaces jehcgp's 'read(A)' when the user is asked for more solutions to a query. Also shows time at which user was asked and when he or she answered.

yesno_more(A):-
current_time(Time), % see below
  ((yesno(['Do you wish to know more?']),
    writeln('Yes'),
    A = y);
  (writeln('No'),
    A = n)),
  current_time(Time1), % see below
  writeln(['Time when questioned:',Time]),
  writeln(['Time when answered:',Time1]),
  !.

*/

yesno_read - replaces jehcgp's 'read(y)' when the user is asked a 'y/n.' question. Also shows time at which user was asked and when he or she answered.

yesno_read:-
current_time(Time), % see below
  ((yesno(['What is your answer?']),
    writeln('Yes'),
    A = y);
  (writeln('No'),
    A = n)),
  current_time(Time1), % see below
  writeln(['Time when questioned:',Time]),
  writeln(['Time when answered:',Time1]),
  (A = n, fail);
  true),
  !.
Appendix A/104

/*
  echoed_read(R) - replaces jehcgp's 'read' clauses where user has to enter a response. Unlike
  read, 'echoed_read' echoes the output. Also shows time at which user was asked and when he or
  she answered.
 */

  echoed_read(R):-
    current_time(Time), % see below
    (prompt_gread(["Please enter your answer:"],R);
      R = u), % 'Cancel' button equates to 'u' for 'unknown'
    writeln(R),
    current_time(Time1), % see below
    writeseqnl(["Time when questioned:",Time]),
    writeseqnl(["Time when answered:",Time1]),
  1.

  quick_entry(Prompt,G) - replaces jehcgp's same clause that uses 'get0', which does not work for
  keyboard input in LPA. Also shows time at which user was asked and when he or she answered.
 */

  quick_entry(Prompt,G):-
    current_time(Time), % see below
    quick_entry_dialog(Prompt,G), % see immediately below
    name(Response,G),
    writeseqnl([Prompt,Response]),
    current_time(Time1), % see below
    writeseqnl(["Time when questioned:",Time]),
    writeseqnl(["Time when answered:",Time1]),
  1.

  quick_entry_dialog(Prompt,G):-
    dialog("Quick-entry" dialog box',50,60,120,370,
      [button(90,10,20,60,'Ok'),
        button(90,290,20,60,'Cancel'),
        text(10,10,32,350,Prompt),
        edit(45,10,32,350,"bytes(G)"),
        Button).
/**
current_time - returns current time in latinate form.
*/
current_time(Time):-
time(Hour,Min,Sec),

((Hour = 0, Hour1 = 12);
 (Hour < 12, Hour1 = Hour),
   AmPm = 'am');
((Hour = 12, Hour1 = Hour);
 Hour1 is Hour - 12),
   AmPm = 'pm'))).

((Min < 10, concat(0,Min,Min1));
 Min1 = Min),

((Sec < 10, concat(0,Sec,Sec1));
 Sec1 = Sec),

I, concat([Hour1,':',Min1,':',Sec1,AmPm,'.'],Time).
/*
   ---------------------------------GENERAL ROUTINES---------------------------------
*/

/*
check_overlap - ensures Rectangle does not overlap with existing graphs.
*/

check_overlap(Window,Rectangle):-
    get_all_pics(Window,Names),
    exclude_links(Window,Names,Names1),  % see below
    intersect_pics(Window,Names1,Rectangle,Names2),  % see below
    Names2 = [],
    !.

/*
exclude_links - removes all the directed links from Names. Used in conjunction with intersect_pics
(see below) to prevent too wide an overlap zone due to the potential large frame of a link,
although this relaxation does allow conrels to overlap links.
*/

exclude_links(Window,Names,Names1):-
    exclude_links1(Window,Names,[]).

exclude_links1(Window,Names,Names):-
    append(Names,Names1,Names2),  % is other than above so keep FirstName
    !,
    exclude_links1(Window,OtherNames,Names1,Names2).

/*
includes_contexts - keeps graphs coherent by deselecting, from SelNames, those graph
elements which have unselected negating contexts.
*/

includes_contexts([],[]).  % nothing selected so don't proceed any further

includes_contexts(Window,SelNames):-
    get_desel_pics(Window,DeselNames),
    DeselNames = [],  % ensures there are unselected graphs to begin with
    [FirstDeselName|OtherDeselNames] = DeselNames,
    includes_contexts1(Window,SelNames,SelNames,FirstDeselName,OtherDeselNames).

includes_contexts([],[]).  % nothing was unselected so don't proceed any further

includes_contexts1([],[],[]).  % nothing selected so don't proceed any further

includes_contexts1(Window,SelNames,SelNames,OtherDeselNames):-
    !,
    includes_contexts1(Window,SelNames,SelNames,OtherDeselNames).
includes_contexts1(Window, SelNames, [FirstSelName|OtherSelNames],
   FirstDeselName, OtherDeselNames):-
   get_pic(Window, FirstDeselName, Description),
   Description = negating_context(DominatedNames, _, _, _),
   on(FirstSelName, DominatedNames),
   desel_pics(Window, [FirstSelName]),
   includes_contexts1(Window, SelNames, OtherSelNames, FirstDeselName, OtherDeselNames).

includes_contexts1(Window, SelNames, [FirstSelName|OtherSelNames],
   FirstDeselName, OtherDeselNames):-
   !,
   includes_contexts1(Window, SelNames, OtherSelNames, FirstDeselName, OtherDeselNames).

% intersect_pics - returns IntersectNames, which is a list taken from a list of incoming Names whose
% frames intersect with a given Rectangle.

/*

intersect_pics(Window, Names, Rectangle, IntersectNames): -
   intersect_pics1(Window, Names, Rectangle,[],IntersectNames).

intersect_pics1(_,[],IntersectNames,IntersectNames).

intersect_pics1(Window, [FirstName|OtherNames], Rectangle, IntersectNames, IntersectNames2):-
   get_pic(Window, FirstName, Description, (Y, X)),
   pic_frame(trans(Y, X, Description), box(Top, Left, Depth, Width)),
   intersect_box(box(Top, Left, Depth, Width), Rectangle,_),
   append(IntersectNames, [FirstName], IntersectNames1),
   !, % also does not allow FirstName to be subsequently rejected on backtracking
   intersect_pics1(Window, OtherNames, Rectangle, IntersectNames1,IntersectNames2).

intersect_pics1(Window, [FirstName|OtherNames], Rectangle, IntersectNames, IntersectNames2):-
   !,
   intersect_pics1(Window, OtherNames, Rectangle, IntersectNames, IntersectNames2).

/*

links_inclusive - returns, from the list Names, the name of every 'odd' directed link (i.e. those
which do not also have in Names its relation and two concepts).

*/

links_inclusive(Window, Names, OddLinkNames): -
   links_inclusive1(Window, Names, Names,[],OddLinkNames).

links_inclusive1(Window, [], [], OddLinkNames, OddLinkNames).

links_inclusive1(Window, [FirstName|OtherNames], Names, OddLinkNames, OddLinkNames2):-
   get_pic(Window, FirstName, Description),
   Description = directed_link([FromConName, RelName, ToConName],_ _),
   (on(FromConName, Names),
   on(RelName, Names),
   on(ToConName, Names),
   OddLinkNames1 = OddLinkNames; % don't add link as its correls are also in Names
   append(OddLinkNames, [FirstName], OddLinkNames1)); % add it as ≥1 correl not in Names
   !,
   links_inclusive1(Window, OtherNames, Names, OddLinkNames1,OddLinkNames2).

links_inclusive1(Window, [FirstName|OtherNames], Names, OddLinkNames, OddLinkNames1):-
   !,
   links_inclusive1(Window, OtherNames, Names, OddLinkNames, OddLinkNames1).
Appendix A

I. redraw contexts - redraws contexts around those graphs that have had elements previously deleted.

redraw_contexts(Window):-
    get_all_pics(Window, Names),
    redraw_contexts1(Window, Names).

redraw_contexts1(Window, [FirstNamelRestNames]):-
    get_pic(Window, FirstName, Description, (Y, X)),
    Description = negating_context(_, Top, Left, Depth, Width),
    del_pic(Window, FirstName),
    Top1 is Top + Y,
    Left1 is Left + X,
    pics_in_box(Window, box(Top1, Left1, Depth, Width), Names),
    (Names = [], % some graphs are still surrounded
     union_boxes(Window, Names, Box)); % see general routines
    Depth1 is Size + 3, % otherwise no graphs now surrounded
    Box = box(Top1, Left1, Depth1, Depth1)), % so prepare to draw 'standard size' empty context
    inset_box(Box, (-8, -8), Biggerbox),
    design_negating_context(Names, Biggerbox, Description1), % see negating contexts
    unique_graph_gensym(Window, negating, NewName), % see general routines
    add_pic(Window, NewName, Description1),
    bring_to_front(Window, Names),
    redraw_contexts1(Window, RestNames).

redraw_contexts1(Window, [FirstNamelRestNames]):-
    redraw_contexts1(Window, RestNames).

relations_all_linked(Window, Names):-
    get_all_pics(Window, AllNames),
    retrieve_linked_relations(Window, AllNames, [], LinkedRelNames),
    check_given_relations(Window, Names, LinkedRelNames).

retrieve_linked_relations (Window, [FirstName|OtherNames], LinkedRelNames, LinkedRelNames2):-
    get_pic(Window, FirstName, Description),
    Description = directed_link([_, RelName_, _]),
    append(LinkedRelNames, [RelName], LinkedRelNames1),
    !,
    retrieve_linked_relations(Window, OtherNames, LinkedRelNames1, LinkedRelNames2).

retrieve_linked_relations (Window, [FirstName|OtherNames], LinkedRelNames, LinkedRelNames2):-
    !,
    retrieve_linked_relations(Window, OtherNames, LinkedRelNames, LinkedRelNames2).
check_given_relations([FirstNameliOtherNames], LinkedRelNames).

check_given_relations(Window,[FirstNameliOtherNames],LinkedRelNames):-
  get_pic(Window,FirstName,Description),
  Description = relation_ovil(_________),
  !, % isn't relation_ovil so carry on with the next one
  check_given_relations(Window,OtherNames,LinkedRelNames).

check_given_relations(_, [FirstNameliOtherNames], LinkedRelNames): -
  on(FirstName, LinkedRelNames). % relation-oval is linked
/* remove_its_encumbrances - delete all the links connected to, and the negating contexts surrounding, a concept or relation. */

remove_its_encumbrances(Window, ConReIName): -
  get_all_pics(Window,Names),
  remove_an_encumbrance(Window,ConReIName,Names).
remove_an_encumbrance(_(_[])).

remove_an_encumbrance(Window,ConReIName, [FirstNameliOtherNames]): -
  get_pic(Window,FirstName,Description),
  remove_if_encumbrance(Window,ConReIName,FirstName,Description),
  !,
  remove_an_encumbrance(Window,ConReIName,OtherNames).

remove_if_encumbrance(Window,ConReIName,FirstName,Description): -
  Description = negating_context(Names,_______);
  Description = directed_link(Names,_______),
  on(ConReIName,Names),
  del_pic(Window,FirstName).

remove_if_encumbrance(_,_______).
/* remove_double_negations - removes double negations, if desired by user. */

remove_double_negations(Window): -
  get_all_pics(Window,Names),
  remove_double_negations1(Window,Names).
remove_double_negations1(_(_[])).

remove_double_negations1(Window,[FirstNameliSecondNameliOtherNames]): -
  get_pic(Window,FirstName,Description,(Y,X)),
  get_pic(Window,SecondName,Description1,(Y1,X1)),
  Description = negating_context(______,Top,Left,Depth,Width),
  Description1 = negating_context(______,Top1,Left1,Depth1,Width1), % two negating contexts together
  Top2 is Top + Y,
  Left2 is Left + X,
  pics_in_box(Window,box(Top2,Left2,Depth,Width),Names),
  Top3 is Top1 + Y1,
  Left3 is Left1 + X1,
  pics_in_box(Window,box(Top3,Left3,Depth1,Width1),Names1),
  append(Names,[FirstName,Names1]), % one inside the other, which has nothing else inside it
  !, % therefore can be double negated
  (yesno(‘’Remove double negations as well?’’)), % asks for user's choice del_pic(Window,FirstName),

  269

Appendix A/04
Appendix A

```
del_pic(Window, SecondName), % yes, therefore double negate by deleting them both
I,
remove_double_negations2(Window, OtherNames)); % see below
(I,
true)). % no, therefore do nothing

remove_double_negations1(Window, [\OtherNames]):
I,
remove_double_negations1(Window, OtherNames).

remove_double_negations2([\]).

remove_double_negations2(Window, [FirstName, SecondName|OtherNames]): % as for 'remove_double_negations1' above but without user prompt now:
get_pic(Window, FirstName, Description),
get_pic(Window, SecondName, Description1),
Description = negating_context(_____),
Description1 = negating_context(_____),
pic_frame(Description, Frame),
pics_in_box(Window, Frame, Names),
remove(SecondName, Names, Names1),
pic_frame(Description1, Frame1),
pics_in_box(Window, Frame1, Names2),
Names2 = Names1,
del_pic(Window, FirstName),
del_pic(Window, SecondName),
I,
remove_double_negations2(Window, OtherNames).

remove_double_negations2(Window, [\OtherNames]):
I,
remove_double_negations2(Window, OtherNames).

/*
sublist - determines a Sublist within List. e.g. The list [c,d,e] is a sublist of [a,b,c,e,f] whereas the list [c,e] is not.
*/
sublist(Sublist, List): -
append(_, List1, List),
append(Sublist, _, List1).

/*
union_boxes - creates Box, which is the overall frame for a given set of picture Names.
*/
union_boxes(Window, [FirstName|OtherNames], Box): - % Names = [FirstName|OtherNames].
get_pic(Window, FirstName, Description, (Y, X)),
pic_frame(trans(Y, X, Description), box(Top, Left, Depth, Width)),
union_boxes1(Window, OtherNames, box(Top, Left, Depth, Width), Box).

union_boxes1([\], Box, Box).

union_boxes1(Window, [NextName|OtherNames], Box, Box2):-
get_pic(Window, NextName, Description, (Y, X)),
pic_frame(trans(Y, X, Description), box(Top, Left, Depth, Width)),
union_box(box(Top, Left, Depth, Width), Box, Box1),
I,
union_boxes1(Window, OtherNames, Box1, Box2).
```
/*
unique_graph_gensym - ensures each graph element has an unique NewName (may become significant when graphs are saved, CARE is quitted and subsequently started up again).

unique_window_gensym - as above except for windows rather than graphs.
*/

unique_graph_gensym(Window,GenericName,NewName):-
    get_all_pics(Window,Names),
    unique_gensym(Names,GenericName,NewName).

unique_gen_sym(List,Root,Symbol):-
    gensym(Root,PossibleSymbol), % generate a possible symbol
    check_gen_sym(List,Root,PossibleSymbol,Symbol).

check_gen_sym(List,Root,PossibleSymbol,Symbol):-
    on(PossibleSymbol,List),
    unique_gen_sym(List,Root,Symbol). % NewName already exists, so try the next

check_gen_sym(_,_,Symbol,Symbol):-! % NewName unique, therefore accept it

unique_window_gensym(GenericWindow,Window1):-
    gensym(GenericWindow,Window),
    ((is_win(Window,_),
    !,
    unique_window_gensym(GenericWindow,Window1)); % window name already exists, get next no.
Window1 = Window).

271
Appendix A104

/*

----------GENERAL TOOLS----------

*/

define graph_select: the 'arrow' tool

graph_select(activate,Window):-
gcursor(Window,arrow).

graph_select(Window,Y,X,Mode):- % drags selected graphs
find_pic(Window,(Y,X),Name),
get_pic(Window,Name,_,_,Selflag),
Selflag = 1,
\[
del_pic(Window,outline, % counters LPA bug should user drag graphs beyond viewing pane
get_sel_pics(Window,SelNames),
links_inclusive(Window,SelNames,OddLinkNames), % see general routines
desel_pics(Window,OddLinkNames), % prevent graph split by deselecting odd links
includes_contexts(Window,SelNames), % see general routines
get_sel_pics(Window,SelNames1),
union_boxes(Window,SelNames1,Box),
add_pic(Window,outline,nilpen(Box)), % add temporary outline showing size of Box
append([outline],SelNames1,DragNames), % enables outline to show size of Box is being dragged
drag_pics(Window,DragNames,(Y,X),NewPoint),
del_pic(Window,outline, % delete outline as finished with it
get_desel_pics(Window,DeselNames),
overlaps_graphs(Window,SelNames1,DeselNames,Box,NewPoint). % see below

graph_select(Window,Y,X,Mode):- % selects graphs by surrounding them with a marqui
marqui(Window,(Y,X),Rectangle),
pics_in_box(Window,Rectangle,Nomes),
sel_pics(Window,Nomes).

graph_select(Window,_,_,Mode):- % click elsewhere deselects all graphs
desel_all(Window).

/*

overlaps_graphs - graphs in SelNames are only shifted if they do not overlap any of those in DeselNames, apart from directed links. These links are excluded (by 'exclude_links' - see general routines). This exclusion stops too wide an overlap zone due to the potential large frame of a link, although allowing conrels to be shifted over links. In any event links attached to SelNames are redrawn by 'reposition_links' below.

*/

overlaps_graphs(Window,SelNames,DeselNames,Box,NewPoint):-
exclude_links(Window,DeselNames,DeselNames1), % see general routines
overlaps_graphs1(Window,SelNames,DeselNames,DeselNames1,Box,NewPoint).

overlaps_graphs1(Window,_,DeselNames1,Box,(Y,X)):-
DeselNames1 \= [], % only continue check if there are any unselected graphs to begin with
pic_frame(trans(Y,X,Box),Frame), % get dimensions of new frame
intersect_pics(Window,DeselNames1,Frame,IntersectNames),
IntersectNames \= [], % do nothing as graphs have been dragged to overlap

overlaps_graphs1(Window,SelNames,DeselNames,_,NewPoint):-

1,shift_pics(Window,SelNames,NewPoint), % go ahead and shift graphs as none overlap
reposition_links(Window,SelNames,DeselNames). % see below

272
/*
reposition_links - ensures links remain properly attached to shifted graphs by changing their appropriate links not shifted at the same time
*/

reposition_links([Window,ShiftedNames,[FirstUnshiftedName]OtherUnshiftedNames]) :-
   reposition_links1(Window,ShiftedNames,ShiftedNames,FirstUnshiftedName,OtherUnshiftedNames).

reposition_links1([Window,ShiftedNames,OtherShiftedNames,FirstUnshiftedName,OtherUnshiftedNames]) :-
   get_pic(Window,FirstUnshiftedName,Description),
   pic_centre(trans(Y,X,Description),NewCentre),
   pic_frame(trans(Y,X,Description),Frame),
   define-arrow(Frame,NewCentre,Point2,NewArrow),
   chg_pic(Window,FirstUnshiftedName,
            directed-link([FromConName,RelName,ToConName],[NewCentre,Point2,Point3],NewArrow,Arrow2)).

1.
reposition_links1(Window,ShiftedNames,OtherUnshiftedNames,FirstUnshiftedName,OtherUnshiftedNames).

1

reposition_links1(Window, ShiftedNames, [FirstShiftedName|OtherShiftedNames],
    FirstUnshiftedName, OtherUnshiftedNames):-
    true,
    reposition_links1(Window, ShiftedNames, OtherShiftedNames, FirstUnshiftedName, OtherUnshiftedNames).

/*
graph_erase: the 'rubber' tool
*/

graph_erase(activate, Window):-
    deselect_all(Window),
    gcursor(Window, garbage).

graph_erase(deactivate, Window):-
    gcursor(Window, arrow).

graph_erase(Window, Y, X, Mode):-
    find_pic(Window, (Y, X), Name),
    remove_its_encumbrances(Window, Name),
    del_pic(Window, Name), % see general routines
/*

--------------------------INPUT--------------------------

*/

graph_assert(activate, Window) :-
    forget(unevaluated_graph), % ensures reset to default
gcursor(Window, exclaim).

graph_query(activate, Window) :-
gcursor(Window, thick_cross).

graph_retract(activate, Window) :-
gcursor(Window, garbage).

graph_assert(deactivate, Window) :-
gcursor(Window, arrow).

graph_query(deactivate, Window) :-
gcursor(Window, arrow).

graph_retract(deactivate, Window) :-
gcursor(Window, arrow).

graph_assert(double, Window) :-
    myesno(['Do you wish the next entered structure to remain unevaluated until the old
knowledge is overridden?']),
gcursor(Window, thick_cross), % shows is in different mode
remember(unevaluated_graph, yes). % used in next clause

graph_assert(Window, Y, X, Mode) :-
    ((recall(unevaluated_graph, yes),
      forget(unevaluated_graph),
      1,
      graph_input(Window, Y, X, 36), % 36 is ASCII for '$' (unevaluate mode)
gcursor(Window, exclaim)); % shows reverted to original mode
    (1,
      graph_input(Window, Y, X, 33))). % 33 is ASCII for 't' (evaluate mode)

graph_query(double, Window) :-
    myesno(['Do you wish to list all the structures in the knowledge-base?']),
    1,
special_command('*'). % see engine additions

graph_query(Window, Y, X, Mode) :-
    graph_input(Window, Y, X, 63). % 63 is ASCII for '?'

graph_retract(double, Window) :-
    myesno(['Do you wish to RETRACT ALL the structures in the knowledge-base?']),
    1,
special_command('-'). % see engine additions

graph_retract(Window, Y, X, Mode) :-
    graph_input(Window, Y, X, retract).
Appendix A

graph_input(Window,Y,X,Command):- % prepares and enters properly selected graphs
desel_all(Window), % deselect any previously selected graphs to arrive at clear starting position
marqu_i(Window,(Y,X),Rectangle),
pics_in_box(Window,Rectangle,Names),
Names \= [], % only proceed if any graphs are selected
sel_pics(Window,Names),
check_graph_selection(Window,Rectangle,Names),
graph_input_0(Window,Names,InternalGraph),
!,
command(Command,InternalGraph). % see engine additions.

graph_input(Window,_,_,_):- % reverts to starting position as graphs improperly selected
desel_all(Window).

check_graph_selection(Window,Rectangle,Names):-
relations_all_linked(Window,Names), % see general routines
get_desel_pics(Window,DeselNames),
intersect_pics(Window,DeselNames,Rectangle,IntersectNames), % see general routines
IntersectNames \= [], % ensures only whole graphs were selected
!. % prevent backtracking should >1 colon be found in checking routines below

check_graph_selection(_,_,_):-
message("Sorry, but nothing entered because at least one incomplete structure was selected.
!), fail.

graph_input_0(Window,Names,InternalGraph):-
reverse(Names,ReversedNames),
remember(label_names,[]), % used in 'concepL_input2' below
graph_input1(Window,ReversedNames,InternalGraph),
forget(label_names). % no longer needed (for 'concepL:input2' below)

graph_input1(Window,Names,InternalGraph2):-
graph_input2(Window,Names,[],InternalGraph,[],NonNegNames),
!,
graph_input3(Window,NonNegNames,InternalGraph,InternalGraph1,[],NonLinkNames),
!,
graph_input4(Window,NonLinkNames,InternalGraph1,InternalGraph2),
!

graph_input2(_,InternalGraph,InternalGraph,NonNegNames,NonNegNames).

graph_input2(Window,[FirstName|OtherNames],InternalGraph,NegatedInternalGraph2, NonNegNames,NonNegNames1):-
get_pic(Window,FirstName,Description),
Description = negating_context(DominatedNames,_,_,_), % see general routines
reverse(DominatedNames,ReversedDomNames),
!,
graph_input1(Window,ReversedDomNames,InternalGraph1),
<jpgp_clause(s)>!InternalGraph1,InternalGraph2), % tidies up the internal graphs
append(InternalGraph,[neg(InternalGraph2)],NegatedInternalGraph1),
remove_all(DominatedNames,OtherNames,OtherNames1),
!,
graph_input2(Window,OtherNames1,NegatedInternalGraph1,NegatedInternalGraph2, NonNegNames,NonNegNames1).

graph_input2(Window,[FirstName|OtherNames],InternalGraph,InternalGraph1, NonNegNames,NonNegNames2):-
append(NonNegNames,[FirstName],NonNegNames1),
!,
graph_input2(Window,OtherNames,InternalGraph,InternalGraph1, NonNegNames1,NonNegNames2).
Appendix A

grapt\_input3([], Internal\_Graph, Internal\_Graph, NonLink\_Names, NonLink\_Names).

grapt\_input3(Window, [FirstName\_OtherNames], Internal\_Graph, Internal\_Graph3, NonLink\_Names, NonLink\_Names1):-
get\_pic(Window, FirstName, Link\_Description),
Link\_Description = directed\_link([From\_name, Rel\_name, To\_name], \ldots),
get\_pic(Window, Rel\_name, Description),
Description = relation\_oval(Text, \ldots),
stringof(Text\_Char\_List, Text),
!,
colon\_absent(Text\_Char\_List), % see below
stringof([\ldots]graph([rel(\ldots)])
label\_options([\ldots], Text\_Char\_List, Text\_Char\_List1), % see below
stringof([\ldots'\ldots'),
append(Text\_Char\_List1, L1, Text\_Char\_List2),
get\_pic(Window, From\_name, Description1),
Description1 = concept\_rectangle(Text1, \ldots),
stringof(Text1\_Char\_List, Text1),
!
concept\_input(Text1\_Char\_List, Wrong\_Quoted\_Internal\_Graph), % see below
stringof(WQ\_L\_List, Wrong\_Quoted\_Internal\_Graph),
stringof([Cons\_Front\_Part\_List, [graph(\ldots)]),
append([Cons\_Front\_Part\_List, Part\_List, WQ\_L\_List],
stringof([Cons\_Back\_Part\_List, \ldots])),
append([Correct\_List, Cons\_Back\_Part\_List, Part\_List],
append(Text\_Char\_List2, Correct\_List, Text\_Char\_List3),
append(Text\_Char\_List3, [\ldots], Text\_Char\_List4),
get\_pic(Window, To\_name, Description2),
Description2 = concept\_rectangle(Text2, \ldots),
stringof(Text2\_Char\_List, Text2),
!
concept\_input(Text2\_Char\_List, Wrong\_Quoted\_Internal\_Graph1), % see below
stringof(WQ\_L\_List1, Wrong\_Quoted\_Internal\_Graph1),
append([Cons\_Front\_Part\_List, Part\_List1, WQ\_L\_List1],
append([Correct\_List1, Cons\_Back\_Part\_List, Part\_List1],
append([Text\_Char\_List4, Correct\_List1, Text\_Char\_List5],
append([Text\_Char\_List5, [\ldots], Text\_Char\_List6],
stringof([Rel\_Back\_Part\_List, \ldots])),
append([Text\_Char\_List6, Rel\_Back\_Part\_List, Text\_Char\_List7],
stringof(Text\_Char\_List7, Quoted\_Internal\_Graph),
pname(Internal\_Graph1, Quoted\_Internal\_Graph),
append([Internal\_Graph, Internal\_Graph1, Internal\_Graph2],
remove\_all([From\_name, Rel\_name, To\_name], Other\_Names, Other\_Names1),
!,
grapht\_input3(Window, Other\_Names1, Internal\_Graph2, Internal\_Graph3, NonLink\_Names, NonLink\_Names1).

graph\_input3(Window, [FirstName\_OtherNames], Internal\_Graph, Internal\_Graph1, NonLink\_Names, NonLink\_Names2):-
append(NonLink\_Names, [FirstName], NonLink\_Names1),
!,
grapht\_input3(Window, Other\_Names, Internal\_Graph, Internal\_Graph1, NonLink\_Names1, NonLink\_Names2).

graph\_input4([], Internal\_Graph, Internal\_Graph).
graph_input4(Window,[FirstName|OtherNames],InternalGraph,InternalGraph3):- get_pic(Window,FirstName,Description), Description = concept_rectangle(Text,_,_,_,_), stringof(TextCharList,Text), 1, concept_input(TextCharList,QuotedInternalGraph),  % see below pname(InternalGraph1,QuotedInternalGraph), append(InternalGraph,InternalGraph1,InternalGraph2), 1, graph_input4(Window,OtherNames,InternalGraph2,InternalGraph3).

graph_input4(Window,[FirstName|OtherNames],InternalGraph,InternalGraph1):- 1, graph_input4(Window,OtherNames,InternalGraph,InternalGraph1).

/*
self-explanatory supporting clauses for above graph_inputs
*/

colon_absent(TextCharList):- on(.;TextCharList), message(["Sorry, but nothing entered - a relation incorporated a colon (;)"]1), !,fail.

colon_absent(_).

concept_input(TextCharList,QuotedInternalGraph):- on(.;TextCharList), % determine if concept contains a colon (;)
  1, % prevent backtracking should >1 colon be found in 'check_for_other_colons' below
  check_for_other_colons(TextCharList,QuotedInternalGraph).

concept_input(TextCharList,QuotedInternalGraph):- % no colon in concept
  concept_input1(TextCharList,QuotedInternalGraph).

check_for_other_colons(TextCharList,._):- remove(.;TextCharList,TextCharList1), on(.;TextCharList1),
  message(["Sorry, but nothing entered - a concept had more than one colon (;)"]1), !,fail.

check_for_other_colons(TextCharList,QuotedInternalGraph):- concept_input2(TextCharList,QuotedInternalGraph). % no other colon so proceed with input

concept_input1(TypeLabelCharList,QuotedInternalGraph):- append(TypeLabelCharList,[.;],TextCharList), % add colon so can be handled by concept_input2
  concept_input2(TextCharList,QuotedInternalGraph).
concept_input2(TextCharList,QuotedInternalGraph) :-
  append(TypeLabelCharList1,"["RawReferentCharList],TextCharList),
  stringof(L,"["graph(["cons(",
  label_options(L,TypeLabelCharList1,TypeLabelCharList1), % see below
  stringof(L1,""),
  append(TypeLabelCharList1,L1,TypeLabelCharList2),
  remove_blank_edges(RawReferentCharList,ReferentCharList), % see below
  stringof(TypeLabelCharList1,TypeLabelMarker), % see comment on next line but one
  recall(label_names,LabelList), % prepares for type label checking against generated referents:
  (((ReferentCharList = []) ; % do this if referent is empty (via concept_input1 or otherwise)
    ReferentCharList = [""]), % also do this if referent merely contains"
  (on((TypeLabelMarker,X),LabelList), % do this if type label found in other inputting
    VariableReferent = X); % take previously generated temporary symbol
    gensym("",VariableReferent), % otherwise generate new temporary symbol
    remember(label_names,
      [(TypeLabelMarker,VariableReferent) | LabelList])), % keep it in case needed again
  stringof(ReferentCharList1,VariableReferent));
  ReferentCharList1 = ReferentCharList), % otherwise leave referent term as entered
  stringof(RedferentCharList1,Referent),
  new_referent_info(ReferentCharList1,Referent), % see below
  obtain_label(Referent,Referent1), % see below
  pnamj(Referent1,QuotedReferent1),
  stringof(ReferentCharList2,QuotedReferent1), % returned to stringof form
  stringof(L3,"")),
  append(ReferentCharList2,L3,ReferentCharList3),
  append(TypeLabelCharList2,ReferentCharList3,CharList),
  stringof(CharList,QuotedInternalGraph).

label_options(L,RawLabelCharList,LabelCharList3):- % used for type and relation labels
  remove_blank_edges(RawLabelCharList,LabelCharList), % see below
  (((LabelCharList = []); % do this if label is in fact empty
    LabelCharList = [""], % also do this if label merely contains"
    VariableLabel = "0", % give label temporary "0" symbol ("0" specially reserved - see start
    stringof(LabelCharList1,VariableLabel));
    LabelCharList1 = LabelCharList), % otherwise leave label term as entered
  stringof(LabelCharList1,Label),
  new_label_info(LabelCharList1,Label), % see below
  obtain_label(Label,Label1), % see below
  pnamj(Label1,QuotedLabel1),
  stringof(LabelCharList2,QuotedLabel1), % returned to 'stringof' form
  append(L,LabelCharList2,LabelCharList3).

new_label_info(LabelCharList,Label):- % selects new labels appearing in graphs
  (LabelCharList = [""|] ; % allow variable labels through
    lex(LabelList,_,% see jehcgp
    on(Label,LabelList))); % graphs have previously defined labels
  ((recall(new_labels,NewLabelList),
    append(NewLabelList,[Label],NewLabelList1),
    remember(new_labels,NewLabelList1));
    remember(new_labels,[Label]))), % used later in 'engine additions'
  !.  

279
new_referent_info(RefCharList, Referent):- % selects new referents appearing in graphs

(RefCharList = ['*'?): % allow variable referents through
(lex(RefList, _), % see jehcgp
on(Referent, RefList)); % graphs have previously defined referents

((recall(new_referents, NewRefList1),
append(NewRefList1, [Referent], NewRefList1),
remember(new_referents, NewRefList1)), % used later in 'engine additions'
).

obtain_label(Label, Label1):- % all other cases
name(Label, LabelName), % done for the benefit of the next jehcgp clause
'<jehcgp_clause(s)>{LabelName, Label1}. % (always succeeds)

conf_situation - sets about the input of directly entered conformity relationships from 'CGP' menu.

conf_situation(Dialog, 1, TypeLabel, IndividualMarker):-
prepare_conf(TypeLabel, IndividualMarker, Conformity),
command(33, [Conformity]). % see engine additions

conf_situation(Dialog, 3, TypeLabel, IndividualMarker):-
prepare_conf(TypeLabel, IndividualMarker, Conformity),
command(63, [Conformity]). % see engine additions

conf_situation(Dialog, 4, TypeLabel, IndividualMarker):-
prepare_conf(TypeLabel, IndividualMarker, Conformity),
command(retract, [Conformity]). % see engine additions

prepare_conf(TypeLabel, IndividualMarker, Conformity1):-
stringof(ConformityCharList, TypeLabel),
remove_blank_edges(TypeLabelCharList, TypeLabelCharList1), % see below
stringof(IndividualMarkerCharList, IndividualMarker),
remove_blank_edges(IndividualMarkerCharList, IndividualMarkerCharList1), % see below
append(IndividualMarkerCharList, IndividualMarkerCharList1, ConformityCharList),
stringof(ConformityCharList, Conformity),
name(Conformity, ConformityNameList), % prepares conf into form usable by next jehcgp clause
'<jehcgp_clause(s)>{ConformityNameList, Conformity1}. % (always succeeds)

lattice_situation - sets about the input of directly entered lattice relationships from 'CGP' menu.

lattice_situation(Dialog, 1, Super, Sub):-
prepare_lattice(Super, Sub, Lattice),
command(33, [Lattice]). % see engine additions

lattice_situation(Dialog, 3, Super, Sub):-
prepare_lattice(Super, Sub, Lattice),
command(63, [Lattice]). % see engine additions
lattice_situation(Dialog,4,Super,Sub):-
  prepare_lattice(Super,Sub,Lattice),
  command(retract,[Lattice]). % see engine additions

prepare_lattice(Super,Sub,Lattice1):-
  stringof(SuperCharList,Super),
  remove_blank_edges(SuperCharList,SuperCharList1), % see below
  stringof(SubCharList,Sub),
  remove_blank_edges(SubCharList,SubCharList1), % see below
  append(SubCharList1,['<','<',SuperCharList1],LatticeCharList),
  stringof(LatticeCharList,Lattice),
  name(Lattice,LatticeNameList), % prepares lattice into form usable by next jehcgp clause
  '<jehcgp_clause(s)>jLatticeNameList,Latticel). % (always succeeds)

/*
remove_blank_edges - remove any blank spaces at the beginning or the end of TextCharList
*/
remove_blank_edges(TextCharList,TextCharList2):-
  remove_leading_blanks(TextCharList,TextCharList1),
  remove_trailing_blanks(TextCharList1,TextCharList2).

remove_leading_blanks(['|Remainder],TextCharList):-
  remove_leading_blanks(Remainder,TextCharList).

remove_leading_blanks(TextCharList,TextCharList).

remove_trailing_blanks(TextCharList,TextCharList3):-
  reverse(TextCharList,TextCharList1),
  remove_leading_blanks(TextCharList1,TextCharList2),
  reverse(TextCharList2,TextCharList3).
/*  
 * MENU HANDLING  
 */

initialise_menus:-
clear_menu('Edit'),
extend_menu('Edit', ['-Undo /Z', '-Copy /C', 'Cut /X', 'Paste /V', 'Clear', 'Draw without/'],
    ['-Balance /B', 'Select All /A', 'Show clipboard ']),
install_menu('CARE', ['-About CARE...', '-New drawing sheet/N', 'Rename front drawing sheet.../M', 'Kill front drawing sheet/', '-Draw "if-Then" forms/7', 'Draw browsing forms/8', '-Find a text string...', 'Find it again in output/'],
    ['-Always remove encumbrances/H', 'MacProlog menus/L']),
install_menu('CGP', ['-Open a previous state.../O', 'Save the current state../P', 'Override old knowledge.../1', 'Lattice...', 'Conformity...', 'Execute a linear form script.../5', 'Command line.../=', 'Engage interactives/-']),

((recall(encumbrances,retain), % sets to position before reinitialisation
unmark_item('CARE', 'Always remove encumbrances'));
mark_item('CARE', 'Always remove encumbrances')),
((recall(macprolog_menus,enabled), % sets to position before reinitialisation
mark_item('CARE', 'MacProlog menus'));
(disable_menu('CARE', 'MacProlog menus')), % see below
((recall(jehcgp_interactives,enabled), % sets to position before reinitialisation
mark_item('CGP', 'Engage interactives'));
unmark_item('CGP', 'Engage interactives'))),
install_menu('Labels', []). % sets up heading for dynamic menu item 'Labels'

menus(activate,Window):-
  (get_prop(conrel_name,Window,); % window has conrel with open edit field
      remember(edit_mode,drawing), % otherwise set edit functions to drawing mode
      enable_item('Edit', 'Draw without'),
      disable_item('Edit', 'Balance '),
      enable_item('CARE', 'Rename front drawing sheet...'),
      enable_item('CARE', 'Kill front drawing sheet'))). % and do these

menus(deactivate,Window):-
forget(edit_mode), % resets edit functions to their defaults (see below)
      disable_item('Edit', 'Draw without'),
      enable_item('Edit', 'Balance '),
      disable_item('CARE', 'Rename front drawing sheet...'),
      disable_item('CARE', 'Kill front drawing sheet').

/*
  'Edit' menu
*/

'Undo' :-
   wfront(Window),
   undo(Window).

282
Appendix A104

'Copy' :- % does this when in drawing mode
recall(edit_mode,drawing),

((wfront(Window),
  get_sel_pics(Window,SelNames),
  links_inclusive(Window,SelNames,OddLinkNames), % see general routines
  OddLinkNames = [],
  % proceed only if there are no selected links attached to unselected conrels
  copy(Window);
  message(['Sorry, but nothing copied as at least one link was selected without its
  attachments.'])�).

'Copy' :- % does this as default
wfront(Window),
copy(Window).

'Cut' :- % does this when in drawing mode
recall(edit_mode,drawing),

((wfront(Window),
  get_sel_pics(Window,SelNames),
  links_inclusive(Window,SelNames,OddLinkNames), % see general routines
  OddLinkNames = [],
  % proceed only if there are no selected links attached to unselected conrels
  get_deselected_pics(Window,DeseINames),
  links_inclusive(Window,DeseINames,OddLinkNames), % see general routines
  OddLinkNames = [],
  % proceed only if there are no unselected links attached to selected conrels
  includes_contexts(Window,SelNames), % see general routines
  SelNames = SelNames, % proceed only if nothing deselected by 'includes_contexts'
  cut(Window);
  message(['Sorry, but nothing cut because at least one incomplete structure was selected.'])�).

'Cut' :- % does this as default
wfront(Window),
cut(Window).

'Paste' :- % does this when in drawing mode
recall(edit_mode,drawing),
% designed to a) discourage users from cluttering windows with pasted graphs, and
% b) prevent name conflicts whereby pasted conrel names & their contexts/links no longer tally:

((wfront(Window),
  get_all_pics(Window,Names),
  Names = [], % window is empty
  Window1 = Window); % therefore paste in front window
  create_drawing_sheet('untitled drawing sheet '),
  wfront(Window1))), % otherwise paste into new window
  paste(Window1).

'Paste' :- % does this as default
wfront(Window),
paste(Window).
'Clear': - % does this when in drawing mode
recall(edit_mode,drawing),

((wfront(Window),
get_sel_pics(Window,SelNames),
links_inclusive(Window,SelNames,OddLinkNames), % see general routines
OddLinkNames = [],
% proceed only if there are no selected links attached to unselected conrels
get_desel_pics(Window,DeselNames),
links_inclusive(Window,DeselNames,OddLinkNames1), % see general routines
OddLinkNames1 = [],
% proceed only if there are no unselected links attached to selected conrels
includes_contexts(Window,SelNames), % see general routines
get_sel_pics(Window,SelNames1),
SelNames1 = SelNames, % proceed only if nothing deselected by 'includes_contexts'
clear(Window);
message(['Sorry, but nothing cleared because at least one incomplete structure was
selected.']))).

'Clear':- % does this as default
wfront(Window),
clear(Window).

'Draw without':-
% user can see what a graph looks like without its properly selected elements:
recall(edit_mode,drawing),
wfront(Window),
get_desel_pics(Window,DeselNames),
DeselNames = [], % only continue if something left to empty from
get_sel_pics(Window,SelNames),
SelNames = [], % only continue if anything selected for removal
links_inclusive(Window,SelNames,OddLinkNames), % see general routines
OddLinkNames = [],
% proceed only if there are no selected links attached to unselected conrels
links_inclusive(Window,DeselNames,OddLinkNames1), % see general routines
OddLinkNames1 = [],
% proceed only if there are no unselected links attached to selected conrels
desel_pics(Window,DeselNames),
set_pics(Window,DeselNames),
copy(Window),
desel_all(Window),
get_gfont(Window,Font,Face,Size),
'Paste';
wfront(Window1),
set_gfont(Window1,Font,Face,Size),
desel_all(Window1),
remove_double_negations(Window1), % see general routines
redraw_contexts(Window1). % see general routines

'Select All ':-
wfront(Window),
((wtype(Window,grap),
set_all(Window)); % does this if graphic Window
cursor(Window,0,-1)). % otherwise does this

'Balance ':-
wfront(Window),
balance(Window).

'Show clipboard':-
wfront('Clipboard').
Appendix A

/*
'CARE' menu
*/

'CARE'('About CARE...'): -
mdialog(60,15,120,480,
[button(90,430,20,40, 'Ok'),
text(6,10,32,460,
'CARE: Conceptual Analysis and Review Environment v1.3 (© Simon Polovina, 1992/3).
Based on John Sowa's Conceptual Structures. '),
text(44,10,32,460,
'Devised in LPA MacProlog® by Simon Polovina, Computer Studies, Loughborough
University of Technology (LUT). Acknowledgements to:'),
text(82,10,32,400,
'John Heaton, Chris Hinde (both at Computer Studies, LUT) and Mark Sofroniou
(Mathematical Sciences, LUT). ')],
Button).

'CARE'('New drawing sheet') :-
create_drawing_sheet('untitled drawing sheet '). % see start

'CARE'('Rename front drawing sheet...'): -
wfront(Window),
mdialog(220,50,80,400,
[button(53,15,20,60, 'Change'),
button(53,325,20,60, 'Cancel'),
text(5,10,16,380, wseq([['Rename', Window, 'to:']]),
edit(25,10,16,380, Window, NewWindow)],
Button),
avoid_window_conflict(NewWindow, NewWindow1), % see drawing windows
wrename(Window, NewWindow1).

'CARE'('Kill front drawing sheet'): -
wfront(Window),
del_prop(conrel-name, Window), % get rid of this, if exists, as well
wkill(Window).

'CARE'('Draw "if-Then" forms'): -
draw_form('If-Then.STD').

'CARE'('Draw browsing forms'): -
draw_form('Browse.STD').

draw_form(Filename): - % loads drawing windows for 'standard' file 'If-Then forms' (helps user to understand graphs):
dvol(Volume),

(find_file(Filename, Volume1);
(concat('Please find ', Filename, Prompt),
old('TEXT', Filename1, Volume1, Prompt),
(Filename1 = Filename; % check that file chosen agrees with expected file
message(['Sorry, but', Filename1, 'is not the correct file. ']),
!, fail))),

jehcgp_window, % see engine additions
nl,
writeseqn(['About to draw the structures in', Filename, '(Knowledge-base unaffected)...']),
dvol(Volume1),
open(Filename, Volume1),
load_drawing_sheets_only(Filename), % see drawing windows
close(Filename), % (continued overleaf)
% (continued from previous page)

dvol(Volume),
current_time(Time), % see engine additions
nl,
write_seqnl(['Command completed at',Time]),
!,
open_drawing_sheets, % see drawing windows
cleanup(grap,41,3,20,10).

'CARE('Find a text string...'):-
  (recall(last_string,[String,]),
   String = "")
!,
dialog('Search string (case sensitive):',100,10,96,490,
   [button(56,75,20,120,'Drawing Sheets'),
    button(56,420,20,60,'Cancel'),
    button(56,210,20,130,'Knowledge-base'),
    button(56,350,20,60,'Output'),
    edit(20,10,16,470,NewString),
    text(56,10,16,60,'Look in:'),
    Button,find_string_(NewString)). % see browsing for 'find_string_'

'CARE('Find it again in output'):-
  find_string_in_output. % see browsing

'CARE('Always remove encumbrances'):-
  % user can leave contexts/links attached to edited conrels
  % (may lead to 'untidy' contexts but useful if want to simply copy inside text):
  marked_item('CARE,'Always remove encumbrances'),
  unmark_item('CARE,'Always remove encumbrances'),
  remember(encumbrances,retain). % tested in concepts and relations

'CARE('Always remove encumbrances'):-
  % user can have contexts/links attached to edited conrels automatically removed
  % (also serves to ensure that these attachments are not retained accidentally):
  mark_item('CARE,'Always remove encumbrances'),
  forget(encumbrances). % tested in concepts and relations

'CARE('MacProlog menus'):-
  % disables LPA MacProlog menus when 'CARE,'MacProlog menus' is unmarked:
  marked_item('CARE,'MacProlog menus'),
  unmark_item('CARE,'MacProlog menus'),
  forget(macprolog_menus). % relevant for 'initialise_menus' above
  disable_lpa_menus. % see below

'CARE('MacProlog menus'):-
  % enables LPA MacProlog menus when 'CARE,'MacProlog menus' is marked:
  mark_item('CARE,'MacProlog menus'),
  remember(macprolog_menus,enabled). % relevant for 'initialise_menus' above
  enable_lpa_menus. % see below

/*
'CARE' menu
*/

'CGP'('Open a previous state...'):-
special_command(.).
'CGP('Save the current state...'): -
   special_command(>).

'CGP('Override old knowledge...'): % rebuilds knowledge-base
   myesno('Rebuild knowledge-base in light of new contradictory and unevaluated / retracted
   knowledge?'),
   special_command(build_model). % see engine additions

'CGP('Lattice...'): -

   (recall(super_sub,[Super,Sub]);
    (Super = "",
     Sub = ""),

   1,
   dialog('Directly entered lattice relationships',100,10,116,490,
    [button(88,30,20,60,’Assert’),
     button(88,390,20,60,’Cancel’),
     button(88,150,20,60,’Query’),
     button(88,270,20,60,’Retract’),
     text(8,10,16,470,’Enter lattice relationship’),
     text(32,10,16,50,’Super:’),
     edit(32,60,16,420,’Super’,NewSuper),
     text(56,10,16,50,’Sub:’),
     edit(56,60,16,420,’Sub’,NewSub)],
    Button, lattice_situation(NewSuper,NewSub)), % see input for 'lattice_situation'
    remember(super_sub,[NewSuper,NewSub]).

'CGP('Conformity...'): -

   (recall(type_conf,[TypeLabel,IndividualMarker]);
    (TypeLabel = "",
     IndividualMarker = ""),

   1,
   dialog('Directly entered conformity relationships',100,10,116,490,
    [button(88,30,20,60,’Assert’),
     button(88,390,20,60,’Cancel’),
     button(88,150,20,60,’Query’),
     button(88,270,20,60,’Retract’),
     text(8,10,16,470,’Enter conformity relationship’),
     text(32,10,16,130,’Type Label:’),
     edit(32,140,16,340,’Type Label’,NewTypeLabel),
     text(56,10,16,130,’Individual Marker:’),
     edit(56,140,16,340,’Individual Marker’,NewIndividualMarker)],
    Button, conf_situation(NewTypeLabel,NewIndividualMarker)), % see input for 'conf_situation'
    remember(type_conf,[NewTypeLabel,NewIndividualMarker]).

'CGP('Execute a linear form script...'): % facility to execute jehcgp's linear form scripts
   special_command('%').
'CGP('Command line...')': % facility to enter jehcgp's single character commands
mdialog(100,50,68,400,
  [button(40,10,20,40,'Ok'),
   button(40,330,20,60,'Cancel'),
   text(8,10,16,120,'Enter command:'),
   edit(8,130,16,260,'o',Command)],
  Button),
stringof(CommandList,Command),

((reverse(CommandList,RevComList),
  RevComList = []\RestRevComList,
  reverse(RestRevComList,CommandList1)); % caters for user entering '.' after command
CommandList1 = CommandList),
special_command(CommandList1). % see engine additions

'CGP('Engage interactives'): % disables interaction with user about super/sub types, canonical relations:
marked_item('CGP';'Engage interactives'),
unmark_item('CGP';'Engage interactives'),
forget(jehcgp_interactives). % tested in engine additions

'CGP('Engage interactives'): % enables interaction with user about super/sub types, canonical relations:
mark_item('CGP';'Engage interactives'),
remember(jehcgp_interactives,enabled). % tested in engine additions

(['/Labels' menu (dynamic)

*/

'Labels'(A to Z):- % disables the placing of labels in A to Z alphabetical order:
marked_item('Labels';'A to Z'),
unmark_item('Labels';'A to Z'),
forget(a2z_labels). % see browsing

'Labels'(A to Z):- % enables labels to be displayed in A to Z alphabetical order:
mark_item('Labels';'A to Z'),
remember(a2z_labels,enabled). % see browsing

'Labels'((item):- % pastes label item into open-for-edit conrel of front drawing window (otherwise does nothing)
cursor('Σ Output Window',0,-1),
clear('Σ Output Window'),
write('Σ Output Window',item),
cursor('Σ Output Window',0,-1),
cut('Σ Output Window'),
wfront(Window),
% wtype(Window,grap), % commented out, so can also paste into other types of window
not recall(edit_mode,drawing), % front drawing window has edit-able conrel
'Paste'.

288
/*
 * disable_lpa_menus - restrict menus to only those items relevant to CARE.
 * enable_lpa_menus - have LPA MacProlog menu items as well.
 */

disable_lpa_menus:-
    disable_item('File', 'New...'),
    disable_item('File', 'Open...'),
    disable_item('File', 'Close...'),
    disable_item('File', 'Save all source'),
    disable_item('File', 'Save file...'),
    disable_item('File', 'Save as...'),
    disable_item('File', 'Source files...'),
    disable_item('File', 'Help...'),
    disable_menu('Search'),
    disable_item('Windows', 'Create...'),
    disable_item('Windows', 'Kill...'),
    disable_item('Windows', 'Window details...'),
    disable_item('Windows', 'Cross reference...'),
    disable_item('Windows', 'Hide'),
    disable_item('Windows', 'Hide all'),
    disable_item('Windows', 'Show all'),
    disable_item('Windows', 'Clean up'),
    disable_item('Windows', 'New graphic window...'),
    disable_menu('Eval').

enable_lpa_menus:-
    enable_item('File', 'New...'),
    enable_item('File', 'Open...'),
    enable_item('File', 'Close...'),
    enable_item('File', 'Save all source'),
    enable_item('File', 'Save file...'),
    enable_item('File', 'Save as...'),
    enable_item('File', 'Source files...'),
    enable_item('File', 'Help...'),
    enable_menu('Search'),
    enable_item('Windows', 'Create...'),
    enable_item('Windows', 'Kill...'),
    enable_item('Windows', 'Window details...'),
    enable_item('Windows', 'Cross reference...'),
    enable_item('Windows', 'Hide'),
    enable_item('Windows', 'Hide all'),
    enable_item('Windows', 'Show all'),
    enable_item('Windows', 'Clean up'),
    enable_item('Windows', 'New graphic window...'),
    enable_menu('Eval').
/*

------------------- NEGATING CONTEXTS -------------------

*/

new_negating_context(activate,Window):-
    deselect_all(Window),
    gcursor(Window,cross_hair).

new_negating_context(deactivate,Window):-
    gcursor(Window,arrow).

new_negating_context(Window,Y,X,Mode):-
    marquee(Window,(Y,X),Rectangle,box(30,30)),
    graphs_within(Window,Rectangle).

graphs_within(Window,box(Top,Left,Depth,Width)):-
    pics_in_box(Window,box(Top,Left,Depth,Width),Names),
    ((Names = [], % ensure some graphs were surrounded
        relations_all_linked(Window,Names), % see general routines
        union_boxes(Window,Names,Box)); % see general routines
    (Names = [], % otherwise if no graphs surrounded
        get_gfont(Window,_,_,Size),
        Depth1 is Size + 3,
        Box = box(Top,Left,Depth1,Depth1)), % then prepare to draw 'standard size' empty context
    inset_box(Box,(-8,-8),Biggerbox),
    get_all_pics(Window,AllNames),
    intersect_pics(Window,AllNames,Biggerbox,IntersectNames), % see general routines
    IntersectNames = Names, % ensure no partial overlaps with other graphs occurred
    design_negating_context(Names,Biggerbox,Description),
    unique_graph_gensym(Window,negating,NewName), % see general routines
    add_pic(Window,NewName,Description),
    bring_to_front(Window,Names).

graphs_within(_,__). % nothing happens because negating context improperly formed

design_negating_context(Names,box(Top,Left,Depth,Width),
    negating_context(Names,Top,Left,Depth,Width)).

negating_context(Names,Top,Left,Depth,Width,
    hashpen(blank(fillbox(Top,Left,Depth,Width,30,30)))).
]/  
  
  ---------------------RELATIONS---------------------
  
  new_relation: the 'r' tool - draws new concepts
  
  new_relation(activate,Window):-
    desel_all(Window),
    gcursor(Window,cross_hair).
  
  new_relation(deactivate,Window):-
    gcursor(Window,arrow).
  
  new_relation(Window,Y,X,Mode):- % edits an existing relation (replaces old relation)
    find_pic(Window,(Y,X),Name,Description,(Ya,Xa)),
    Description = relation_oval(Text,Font,Face,Size,Top,Left),
    l,
    (recall(encumbrances,retain); % leave attached contexts/links (may result in 'untidy' contexts)
      remove_its_encumbrances(Window,Name)), % otherwise remove (see general routines)
    del_pic(Window,Name),
    set_gfont(Window,Font,Face,Size),
    Top1 is Top + Ya,
    Top2 is Top1 - 9,
    Left1 is Left + Xa,
    Left2 is Left1 - 6,
    Depth is Size * 3,
    set_prop(conrel-name,Window,Name), % retain name if needed for attached contexts/links
    forget(edit_mode), % ensures edit functions are reset to default (see menu handling)
    add_pic(Window,temp,oval(Top2,Left2,Depth,150)), % temporary outline relation oval
    menus(deactivate,Window), % use gactcleact 'menus' program
    edit_line(Window,Top1,Left1,0,Text).
  
  new_relation(Window,Y,X,Mode):- % creates a new relation provided it is properly formed
    marqui(Window,(Y,X),Rectangle,oval),
    relation_overlap(Window,Rectangle).
  
  relation_overlap(Window,box(Top,Left,Width)):-
    wfont(Window,_,_,Size),
    Top1 is Top + 9,
    Left1 is Left + 6,
    Depth is Size * 3,
    check_overlap(Window,box(Top,Left,Depth,Width)), % see general routines
    add_pic(Window,temp,blank(filloval(Top,Left,Depth,Width))), % temp outline relation oval
    forget(edit_mode), % ensures edit functions are reset to default (see menu handling)
    unique_graph_gensym(Window,relation,NewName), % obtain NewName (see general
    routines)
    set_prop(conrel-name,Window,NewName), % used because of edit existing relations above
    menus(deactivate,Window), % use gactcleact 'menus' program
    edit_line(Window,Top1,Left1,0).
  
  relation_overlap(_). % do nothing as relation improperly formed
new_relation(close_edit,Window):-
    get_text(Window,Details),
    design_new_relation(Details,Description),
    get_prop(conrel_name,Window,NewName),  \% gives edited relation its name
    del_prop(conrel_name,Window),
    del_pic(Window,temp),  \% remove temporary outline relation oval as done with now
    get_all_pics(Window,NonNewNames),
    add_pic(Window,NewName,Description),
    reposition_links(Window,[NewName],NonNewNames),  \% see general tools
    menus(activate,Window),  \% use gactdeact 'menus' program
    remember(edit_mode,drawing).
    \% ensure edit functions are set to drawing (see menu handling)

    design_new_relation(text(Font,Face,Size,box(Top,Left,_,_),Text),
    relation_oval(Text,Font,Face,Size,Top,Left,Description)).

relation_oval(Text,Font,Face,Size,Top,Left,Description):-
    text_width(Text,Font,Face,Size,Width),
    Top1 is Top - 9,
    Left1 is Left - 6,
    Depth is Size * 3,
    Width1 is Width + 12,
    Description = [ blank(filloval(Top1,Left1,Depth,Width1)),
    textline(Font,Size,Face,Top,Left,Text)].
Appendix A

I.

*/

-----------------------------START-----------------------------

*/

"<LOAD>'(CARE):-  
grafix,  
starttext, % see engine additions  
startsheets. % see below

"<QUIT>'(CARE):-  
myesno(['Do you wish to save the current state?']),  
special_command(>).

"<MENUS>'(ForLabels):-  
is_menu('Labels'),  
retrieve_labels(LabelList), % see browsing  
clear_menu('Labels'),  
extend_menu('Labels','A to Z',[,[],LabelList]),

((recall(a2z_labels,enabled),  
mark_item('Labels','A to Z')); % restore its status  
true).

startsheets:-  
initialise_menus, % see menu handling  
% disable_item('CARE','MacProlog menus'), % stops users interfering with CARE itself  
init gensym(untitled drawing sheet '),  
genint('untitled drawing sheet ', % dispenses with 'cgp - drawing sheet 0' (i.e. will start at 1)  
init gensym('Definition '),  
genint('Definition ', % similarly with 'Definition 0' (see get_data in engine additions)  
create_drawing_sheet("untitled drawing sheet ").

create_drawing_sheet(GenericWindow):-  
unique_window gensym(GenericWindow,Window), % see general routines  
create_drawing_sheet1(Window).

create_drawing_sheet1(Window):-  
create_drawing_sheet2(Window),  
add_remaining_tools(Window). % see below

create_drawing_sheet2(Window):-  
init gensym(""),  
genint(""), % "0" is specifically reserved for undefined type/relation labels (see input)  
init gensym(concept),  
init gensym(link),  
init gensym(negated),  
init gensym(relation),  
wcreate(Window,41,3270,460,80,300,300,1,1),  
wfont(Window,'Geneva',0,12),  
gviewer(Window,on),  
gactdeact(Window,menus), % see menu handling  
add_tools(Window,  
[graph select (arrowo),  
graph erase (rubbero),  
new concept (concept rectangle('t:i','Geneva',9,12,8,4),  
new relation (relation oval (' r '; 'Geneva',9,10,10,7),  
new directed link (quasi directed link([[0,0),(16,16),(32,32)])), % see below  
new negating context (textline('Geneva',24,1,3,-2,'(.'))])].
add_remaining_tools(Window):- % adds graph assert, query and retract tools
    add_tools(Window,
    [graph_assert(textline('Chicago',24,1,1,10,'!'))],
    graph_query(textline('Chicago',24,1,1,7,'?'))),
    graph_retract(
    [textline('Chicago',24,1,1,10,'!'),
      line((0,0),(32,32)),
      line((0,32),(32,0))]),
    add_tools(Window,[pic_info(info_icon())]), % used in CARE program development
    !.

% quasi_directed_link - draws arrows for tool pane only:

quasi_directed_link([Point1,Point2,Point3],Description):-
    define_arrow2(Point1,Point2,Point3,ArrowPoint1,ArrowPoint2),
    define_arrow2(Point2,Point3,Point3,ArrowPoint3,ArrowPoint4),
    Description = [lines([Point1,Point2,Point3]),
                    fillpoly([Point2,ArrowPoint1,ArrowPoint2]),
                    fillpoly([Point3,ArrowPoint3,ArrowPoint4])].
A/05: Guideline Procedures in Adapting JEHCGP Program Itself for CARE.

General points on compiling source code in LPA MacProlog:

- A warning is usually given should LPA i) find a functor already compiled in another window, or ii) be about to compile a new definition for a 'no source' relation\(^1\). Furthermore LPA does not distinguish between different arities for any such functors. Therefore be prepared to bring them into the same window or, if such a step is inappropriate, to rename these functors accordingly. Otherwise the 'old' clause will be lost through being overwritten by the 'new' one. All this does not matter when it is deliberately intended to overwrite existing 'no source' clauses. These exceptions are defined below.

- Similarly be prepared to update for functors in JEHCGP which clash with LPA itself, irrespective of arity as indicated above. Change the name given to any such functors (e.g. by adding an underscore at the end, such as changing 'get-text' to 'get_text_').

After duly considering the above, proceed as follows:

- As JEHCGP is too big for a single LPA window, let LPA open it over a number of windows but then cut and paste 'chopped clauses' to rejoin them. Make sure the same predicate heads are in the same window.

- Check, and update accordingly, for JEHCGP clauses requiring replacement of command-line commands, such as 'read', with dialogue box commands. Latest instances are a) 'read(y)' to 'yesno_read' throughout, b) 'read(A)' under the clause 'execute(63,Data,single)' to 'yesno_more(A)', c) 'read(C)' to 'tfu_read(C)' under 'check_with_user(G,D,A)'.

- Examine the JEHCGP command-line functors 'welcome', 'cgp' and 'cgp1'. Check that any changes made to these clauses are also appropriately replicated in their corresponding drawing window-based equivalents in CARE (e.g. 'command'). Similarly check for functors which are instead redefined by CARE such as 'read_data' (detailed under the 'no source' clauses step below).

- Check 'special_command(*)', located in the 'care (interface)' program file window called 'engine additions'. Ensure that the print out of the knowledge-base enacted by this predicate is updated to reflect any JEHCGP enhancements, whilst excluding all JEHCGP program internal items (e.g. 'alias').

---

\(^1\) The term 'no source' refers to a prolog clause that exists only in previously compiled object form. These essentially occur for a) LPA MacProlog predefined system predicates, or b) those already compiled clauses brought in from an object program file such as 'JEHCGP.obj' (discussed later).
• Comment out the inappropriate functor 'valid_label' including its call in 'build_label2'. Check, and update accordingly, for any new JEHCGP 'asserta' clauses that need to be replaced by 'assertx' or 'retractx' in LPA\(^2\).

• Compile JEHCGP to find any predicates that LPA finds defined in more than one window, irrespective of arity as indicated above. Correct such predicates (e.g. by editing or moving them) so that this problem no longer occurs.

• Recompile. Save as 'JEHCGP' in both source and object form. Reinitialise LPA prolog.

• Load 'JEHCGP.obj' ('JEHCGP' in object form). Then load the 'care (interface)' program. NB: Must always edit the program window 'start' of 'care (interface)', even if only to add and then delete the same character. Otherwise LPA will not recompile that window and thereby not subsequently execute the '<LOAD>' clause.

• Recompile. Where a warning appears about redefining a 'no source' clause where the redefining clause occurs in the 'engine additions' program window of 'care (interface)', click the continue button to deliberately redefine said clause if valid to do so. This step applies to the clauses a) 'read_data', b) 'info', c) clauses using 'assertx' and 'retractx'. Change the name given to any others in 'care (interface)' by, say, adding an underscore at the end. For instance turn 'new-concept' into 'new_concept_'. Recompile if necessary.

• Finally, save as 'care.obj'.

\(^2\) To preserve the agreed confidentiality of the JEHCGP program, the details of these changes cannot be explained further.
A/06: Example CARE Screens (version 1.0)

Figure A/06.01: Menu and tool bars, together with example graphs.

Figure A/06.02: Output from asserting example graphs into knowledge-base.
Figure A/06.03: Building the negative contexts for the example graphs.
B/01: Attendees at the Seventh Annual Workshop on Conceptual Graphs

(New Mexico State University, Las Cruces, New Mexico, USA, 8–10 July 1992)

Alex Bejan
IBM Corp.
CIM Architecture 30AZ
971 Neighborhood Rd
Kingston, NY 12401
alge@kgnvny.vnet.ibm.com

Dr. John Esch
Paramax Systems Inc.
Unisys
P.O. Box 64525 M.S. U1N28
St. Paul, MN 55164-0525
esch@email.sp.unisys.com

Allan Hodgson
Dept. of Manufacturing Engineering
Loughborough University of Technology
Loughborough, Leicestershire
LE11 3TU
UNITED KINGDON
a.hodgson@lut.ac.uk

Bill Rich
1525 Addiewell Pl.
San Jose, CA 95120
billrich@vnet.ibm.com

Bernard Moulin
Computer Science Department
Universite Laval
Ste Foy, Quebec G1K 7P4
CANADA
moulin@lavalvm1.bitnet

Brian Garner
Dept. of Comp. & Math.
Deakin University
Geelong 3217
Victoria
Australia
brian@deakin.oz.au

Chris Khoo
School of Information Studies
Syracuse University
4-206 Center for Sci. and Tech.
Syracuse, NY 14244
cskhoo@svum.bitnet

Dickson Lukose
Dept. of Comp. & Math.
Deakin University
Geelong 3217
Victoria, Australia
lukose@deakin.oz.au

Dr. Gary Berg-Cross
Advanced Decision Systems/BA&H
2111 Wilson Blvd, Suite 800
Rosslyn, VA 22201
garybc@chesapeake.ads.com

Gi-Chul Yang
Computer Science
Univ. of Missouri-K.C.
Kansas City, Missouri 64112
yang@vax2.cstp.umkc.edu

Guy Mineau
Dept. d'Informatique (Computer Sc. Dept.)
Universite Laval
Quebec City, Quebec
CANADA, G1K 7P4
mineau@lavalvm1.bitnet

Harry S. Delugach
Department of Computer Science
Univ. of Alabama @ Huntsville
Huntsville, AL 35899
delugach@cs.uah.edu

Heather Pfeiffer
Computing Research Lab.
New Mexico State University
Box 30001/3CRL
Las Cruces, NM 88003-0001
hp@nmsu.edu

Jacques Bouaud
DIAM & SIM
INSERM U.194 & Service d'Informatique Medicale
91, bd de l'Hospital
F-75635 Paris Cedex 13-FRANCE
bouaud@frsim51.bitnet

Jean Fargues
IBM Paris Scientific Center
3 et 5 Place Vendome
75001 Paris, FRANCE
fargues@frsim51.bitnet

John Sowa
Education Center
IBM Systems Research
500 Columbus Avenue
Thornwood, NY 10594
sowa@watson.ibm.com

Lutz Euler
University of Hamburg
Computer Science Department
Natural Language Systems (NatS)
Bodenstedstr. 16
W-2000 Hamburg 50, Germany
lutz@nats4.informatik.uni-hamburg.de

M. Chein
LIRMM
(CNRS et Univ. Montpellier II)
860 rue de Saint-Priest
34090 France
chein@cri.fr

Marie-Laure Mugnier
LIRMM
(CNRS et Univ. Montpellier II)
860 rue de Saint-Priest
34090 France
mugnier@cri.fr

Martin Schroder
University of Hamburg
Computer Science Department
Natural Language Systems (NatS)
Bodenstedstr. 16
W-2000 Hamburg 50, Germany
martin@nats4.informatik.uni-hamburg.de

Mary Keeler
1102 NW 83rd St.
Seattle, WA 98117
mak@vm.gwu.edu
Conceptual Analysis and Review Environment (CARE) Version 1.0: Tool Pane Quick Reference

Arrow Tool *(shown activated)*. Select a drawing by dragging a marqui over it as required. Drag a selected drawing by positioning the cursor over the upper left-hand corner of the appropriate concept, relation or outer negative context and then holding down the mouse button. A 'hand' cursor will appear together with a rectangular 'outline' of the entire selected drawings to show they are being dragged. Cut, Copy or Clear selected drawings via the 'Edit' menu.

Eraser Tool. Erase a drawing with this tool by positioning the cursor over that drawing, in the upper left-hand corner if appropriate, and clicking once.

Concept Edit Tool. Create a new concept by positioning the cursor on the desired starting point on the drawing pane and, by holding down the mouse button, dragging a marqui to an estimated size of the concept. Then type in the appropriate text string. Once complete, click anywhere outside the concept. Edit previously created concepts by clicking within the upper left-hand of that concept.

Relation Edit Tool. Performs for relations the same as the concept edit tool above.

Directed Link Tool. Create a new directed link by positioning the cursor on the upper left-hand of the first concept and, by holding down the mouse button, drag a line to the upper left-hand of the relation. Then release the mouse button and drag a second line to the upper left-hand of the second concept.

Negative Context Tool. Create a negative context by dragging a marqui over a drawing as required.

Structure Assert Tool. Assert a structure into the knowledge-base by dragging a marqui over a drawing as required. *Double click* on the tool if you do not wish the structure to be evaluated until the old knowledge is overridden in the 'CGP' menu.

Structure Query Tool. Query a structure in the knowledge-base by dragging a marqui over a drawing as required. *Double click* on the tool if you wish to list the entire knowledge-base.

Structure Retract Tool. Retract a structure in the knowledge-base by dragging a marqui over a drawing as required. NB: The structure is not actually retracted until the old knowledge is overridden via the 'CGP' menu. *Double click* on the tool if you wish to retract immediately *all* that is held in the entire knowledge-base.

Graphic Information Tool. Shows the internal prolog-based form of a drawing. *Double click* on the tool for further details.

Author: Simon Polovina

Version as at: Friday, July 10, 1992
Conceptual Analysis and Review Environment (CARE) Version 1.0: Menu Quick Reference

**Edit** *(where differs from standard Macintosh edit functions):*

**Paste**: When pasting a drawing where the front drawing sheet is not empty, a new untitled drawing sheet is automatically generated to take the pasted drawing.

**Balance**: In a linear output structure, by positioning the cursor within that structure and enacting this function, the area between the structure's matching brackets will be highlighted so enabling their scope to be seen more easily.

**CARE** *(supplementary to self-explanatory menu items):*

**Open a previous state**: This also clears the prevailing knowledge-base, although any existing drawing sheets are left untouched.

**Find a text string**: Does not search the internal knowledge-base. Therefore list the knowledge-base beforehand by double clicking on the 'T' tool if required.

**MacProlog menus**: Shows, by being disabled ('greyed out') itself, that all LPA MacProlog menu items not relevant to the operation of CARE are disabled.

**CGP** *(in addition to JEHCSP User Guide):*

**Override old knowledge**: Each previously unknown structure added to, or retracted from, the knowledge-base is recorded as episode of knowledge. Unless a retraction, or a structure entered immediately after double clicking on the 'T' tool, the structure is also evaluated. If the newly evaluated structure does not make the knowledge-base inconsistent then that structure is also added to the main knowledge-base. Otherwise it remains an episode only. However the knowledge-base may be rebuilt to override its previous knowledge in favour of the later contradictory knowledge, together with retracted and double clicked 'T' items, by enacting this menu item.

**Engage interactives**: Replaces the JEHCSP command '%user. '. When this menu item is ticked the interactive conversational mechanism is on (See section 4.3 of JEHCSP User Guide).

**Lattice**: Replaces the JEHCSP command line equivalents for the lattice.

**Conformity**: Replaces the JEHCSP command line equivalents for conformities.

**Execute a linear form script**: Replaces the JEHCSP command '%<file>.'.

**Command line**: Facility to execute certain JEHCSP commands in their native form.

Author: Simon Polovina

Version as at: Friday, July 10, 1992
CARE and JEHCGP:
User Evaluation Questionnaire

JEHCGP (Version 2.0) is the conceptual structures processor written in prolog by John Heaton. CARE (Version 1.0), written in LPA MacProlog by Simon Polovina, is a user-friendly front end to JEHCGP. In particular CARE is based on conceptual structures in their diagrammatic display form whereas JEHCGP employs purely the textual linear form.

Thank you for taking the time and trouble to evaluate CARE and JEHCGP. Please take a few moments to let me know what you thought about the software by letting me have your written comments. To help you arrange your thoughts and give your reasons, the following pointers may be useful:

- What was best feature about a) CARE, b) JEHCGP?
- What was worst feature about a) CARE, b) JEHCGP?
- What do you think should be added to a) CARE, b) JEHCGP?
- What do you think should be removed from a) CARE, b) JEHCGP?
- How intuitive was the software? If appropriate, where would you like to see improvements on this aspect?
- Is CARE sufficiently self-explanatory? What form of elucidation should go with it if any (e.g. Full User Reference Manual, Brief User Introduction or Tutorial)?
- Do you think CARE's way of maintaining the integrity of the display form structures is adequate? What changes, if any, would you make?
- Do you like JEHCGP's way of reporting its results? What changes, if any, would you make?
- How useful were the browsing facilities (e.g. listing the knowledge-base, text string searches). What would you add, change or remove?

Thanks again,

Simon
To: A Key Departmental Member

From: Expert 1

12th January 1992

RESEARCH COMMITTEE GUIDELINES

Attached are what I think we agreed were to be the guidelines by which the School's Research Committee will proceed to assess submissions to it for funding.

I think we have got it almost right but not quite. We do not seem to have any policies that are different from those of the University Research Committee except in a minor way. The whole set of guidelines is to prioritise submissions to the University Committee and not to prioritise our own funding. Should we have some others? For instance, if there is no money available for attendance at a good conference in the UK out of other School funds, should we not be willing to discuss a submission for some?

**

To: Research Committee

From: Expert 1

15th January 1992

Please find attached an updated and augmented version of the guidelines that we discussed at the last Research Committee meeting.

Please let me have any comments that you may have. 'A Key Departmental Member' wishes to circulate our guidelines to all members of the academic staff by the end of January. Thus would you reply with your comments by the 27th January.

Many thanks
To: All Academic Staff

From: Business School Research Committee

31st January 1992

PRIORITISING RESEARCH SUBMISSIONS AND FUNDING RESEARCH APPLICATIONS

Until now, there have been no guidelines to help the School Research Committee prioritise bids for University research monies and for deciding how the Business School's research budget should be allocated. There is a need for such guidelines so that the decisions of the committee are consistent, and to provide guidance to School staff as to the eligibility and chances of success of any application for funds that they might be thinking of making.

The following guidelines have been agreed by the School Research Committee:

a) All things being equal, funds will be allocated in the following order (highest priority first):

   Probationers
   Lecturers
   Senior lecturers
   Readers
   Professors

   Joint applications will be looked on particularly favourably.

b) Only in exceptional circumstances will funding be given more than twice under any one research heading to any individual (including inclusion in joint applications) in any one financial year.

d) Applications will be judged not only on the quality of the application itself, but also on the use to which the applicants have used monies previously granted and/or supported by the School Research Committee.

e) Subsequent grants of monies will be significantly jeopardised if some tangible output does not result from any funding from the School Research Committee or supported by the Committee. This output can be in the form of a paper in the research series, an article in an academic journal or through giving a School seminar.

f) Where a bid is for attendance at a conference, the standing of a conference will be taken into account in assessing whether the School Research Committee will support the bid. The lack of any associated official proceedings in which the applicants' work will be published will significantly weaken an application.
The Business School Research Committee will consider costs associated with attendance at UK conferences although this is outside the remit of the University Research Committee.

g) In general, expenditure will only be funded by the School Research Committee in line with the officially agreed University levels. Exceptions are:

   i) that travel for any member of staff (including Professors) will only be funded at second/economy class rates.

   ii) Expenditure on surface travel will be reimbursed at public transport rates. The car allowance will not be given.

Accommodation at conferences will be funded by the School Research Committee to the higher of the University rate and the lowest offered conference rate.

h) For consistency, the expenditure for interview visits to unspecific destinations should be estimated at the standard rate of £40 per visit: this sum includes travel, subsistence and any other costs.
B/06: The Initial Conceptual Graphs Representing Expert 1’s Chosen Problem

Figure B/06.01: Probationer priority (part of guideline ‘a’)

Figure B/06.02: Lecturer priority (part of guideline ‘a’)

Figure B/06.03: Senior Lecturer priority (part of guideline ‘a’)

307
Figure B/06.04: Reader priority (part of guideline ‘a’)

Figure B/06.05: Professor priority (part of guideline ‘a’)

Figure B/06.06: Say Probationer makes a funding request (relates to guideline ‘a’)
Figure B/06.07: Say Professor makes a funding request (relates to guideline 'a')

Figure B/06.08: Query: Does a Probationer obtain priority funding? (relates to guideline 'a')

Figure B/06.09: Only in exceptional circumstances... (guideline 'b')
Figure B/06.10: UK conferences are not funded (guideline ‘g’)

Figure B/06.11: Travel excepted from University levels (part of guideline ‘h’)

Appendix B/06
B/07: Pro Forma 'If-Then' Graphs

**Drawing sheet title:** 'If A then B'

**Drawing sheet title:** 'If C and D then E'

**Drawing sheet title:** 'If F and not G then H'

**Drawing sheet title:** 'If not I then J'

**Drawing sheet title:** 'If K then not L (before double negation)'

**Drawing sheet title:** 'If K then not L (after double negation)'

311
Figure B/08.01: Current seniority relationships

Figure B/08.01 (or Figure '01 for short) shows the current interrelationship between the direct seniorities between the staff members. Indirect seniorities, such as that between a Probationer and a Reader for example, could also be modelled in the same fashion. This, however, would make for a large number of graphs as each permutation is drawn. The advanced nature of conceptual graphs permit a more elegant alternative which is shown by Figure '02.

Figure B/08.02: The generally applicable concept of seniority

After including the type hierarchy graph of Figure '03, Figure '02 reveals the generally applicable concept of seniority. It shows that seniority is transitive in nature. For example if we wished to establish if a Reader was the senior of
a Probationer, the above graph could determine this by specialising its concepts 'Person: *b' and 'Person: *c' to 'Probationer: *b' and 'Lecturer: *c' respectively. This would be valid because of the dominant graphs in Figure '01 showing this particular seniority. Similarly 'Person: *a' and 'Person: *b' could specialise to 'Lecturer: *a' and 'Senior Lecturer: *b'. The simple graph at level 1 could thus be deiterated and the simple graph at level 2 asserted at level 0 after the consequent double negation, but not before that level 2 graph is specialised via the coreferent links to show that Senior Lecturer is the senior of Probationer.

Figure B/08.03: hierarchy of staff member interrelationships.

The new level 0 graph in turn could call upon Figure '02 again, this time specialising 'Person: *b' and 'Person: *c' to 'Probationer: *b' and 'Senior Lecturer: *c'. From Figure '01 'Person: *a' and 'Person: *b' could specialise to 'Senior Lecturer: *a' and 'Reader: *b'. Thus after coreferent passing, deiteration and double negation the graph in Figure '04 would be asserted, showing that Reader is indeed senior of Probationer.

Figure B/08.04: Derived graph showing that Reader is senior of Probationer

The following edited output from CARE demonstrates the above:

Command started at 10:15:19 am.
New type or relation labels:

   senior.
   Person.

lattice relation added:
   Person << universal_type.

lattice relation added:
   senior << universal_rel.

Thinking...

Thinking...

I did not know that:
{
   {[Person:*0]-
     (senior)->[Person:*1]
   }
   [Person:*2]-
     (senior)<-[Person:*0]
     (senior)->[Person:*1]
   .

Tidying up...

Command completed at 10:15:28 am.

Command started at 10:15:59 am.

New type or relation labels:

   Probationer.
   Lecturer.
   Senior Lecturer.
   Reader.
   Professor.

lattice relation added:
   Reader << universal_type.

lattice relation added:
   Professor << universal_type.

lattice relation added:
   Senior Lecturer << universal_type.

lattice relation added:
   Lecturer << universal_type.

lattice relation added:
   Probationer << universal_type.
Thinking...

Thinking...

I did not know that:

[Reader:*6]-
  (senior)<-[Senior Lecturer:*5]-
  (senior)<-[Lecturer:*4]-
  (senior)<-[Probationer:*3],
  (senior)->[Professor:*7].

Tidying up...

conformity relation added:
  Reader :: *6.

conformity relation added:
  Professor :: *7.

conformity relation added:
  Senior Lecturer :: *5.

conformity relation added:
  Lecturer :: *4.

conformity relation added:
  Probationer :: *3.

Command completed at 10:16:19 am.

Command started at 10:17:04 am.

New referents:
  Staff member.

Thinking...

Thinking...

I did not know that:
  (supertype)
  <-[type:Senior Lecturer]
  <-[type:Reader]
  <-[type:Lecturer]
  <-[type:Professor]
  <-[type:Probationer]
  ->[type:Staff member];
  [type:Staff member]-
  (supertype)->[type:Person].

Tidying up...
Appendix B/08

lattice relation added:
   Staff member << Person.

lattice relation added:
   Probationer << Staff member.

lattice relation added:
   Professor << Staff member.

lattice relation added:
   Lecturer << Staff member.

lattice relation added:
   Reader << Staff member.

lattice relation added:
   Senior Lecturer << Staff member.

Command completed at 10:17:31 am.

Command started at 10:18:35 am.

Thinking...

Thinking...

The statement:
   [Probationer:*8]-(senior)->[Reader:*9].

is:

TRUE because:
0. proof of:
   [Probationer:*8]-(senior)->[Reader:*9];
0. is:
1. deduction:
1. known rule:
   ( [Person:*2]-(senior)->[Person:*1];
   [Person:*0]-(senior)->[Person:*2];
   ( [Person:*0]-(senior)->[Person:*1] )
);
1. so:
   ( 
   [Person:*8]-
   (senior)->[Person:*9] 
   )
   [Person:*2]-
   (senior)->[Person:*9];
   [Person:*8]-
   (senior)->[Person:*2] 
   );

1. proving antecedent:
   [Person:*2]-
   (senior)->[Person:*9];

1. proof of:
   [Person:*2]-
   (senior)->[Person:*9];

1. is:

2. fact:
   [Lecturer:*4]-
   (senior)->[Senior Lecturer:*5];

1. proving antecedent:
   [Person:*8]-
   (senior)->[Lecturer:*4];

1. proof of:
   [Person:*8]-
   (senior)->[Lecturer:*4];

1. is:

2. fact:
   [Probationer:*3]-
   (senior)->[Lecturer:*4];

1. results:
   [Probationer:*3]-
   (senior)->[Senior Lecturer:*5];

0. proof of:
   [Probationer:*8]-
   (senior)->[Reader:*9];

0. is:
1. deduction:
2. known rule:
   \[
   \begin{align*}
   &([\text{Person:}*2]-
   \quad (\text{senior})\rightarrow[\text{Person:}*1]; \\
   &([\text{Person:}*0]-
   \quad (\text{senior})\rightarrow[\text{Person:}*2]; \\
   &([\text{Person:}*0]-
   \quad (\text{senior})\rightarrow[\text{Person:}*1]
   \end{align*}
   \]

3. so:
   \[
   \begin{align*}
   &([\text{Person:}*8]-
   \quad (\text{senior})\rightarrow[\text{Person:}*9] \\
   &([\text{Person:}*2]-
   \quad (\text{senior})\rightarrow[\text{Person:}*9]; \\
   &([\text{Person:}*8]-
   \quad (\text{senior})\rightarrow[\text{Person:}*2]
   \end{align*}
   \]

4. proving antecedent:
   \[
   ([\text{Person:}*2]-
   \quad (\text{senior})\rightarrow[\text{Person:}*9]; \\
   ([\text{Person:}*2]-
   \quad (\text{senior})\rightarrow[\text{Person:}*9]; \\
   \text{is:}
   \]

5. fact:
   \[
   ([\text{Senior Lecturer:}*5]-
   \quad (\text{senior})\rightarrow[\text{Reader:}*6]; \\
   \]

6. proving antecedent:
   \[
   ([\text{Person:}*8]-
   \quad (\text{senior})\rightarrow[\text{Senior Lecturer:}*5]; \\
   ([\text{Person:}*8]-
   \quad (\text{senior})\rightarrow[\text{Senior Lecturer:}*5]; \\
   \text{is:}
   \]

7. fact:
   \[
   ([\text{Probationer:}*3]-
   \quad (\text{senior})\rightarrow[\text{Senior Lecturer:}*5]; \\
   \]

0. result:
   \[
   ([\text{Probationer:}*3]-
   \quad (\text{senior})\rightarrow[\text{Reader:}*6].
   \]

Command completed at 10:19:59 am.
Should a new category of, say, Assistant Lecturer be created then this can be modelled simply as illustrated by Figure '05. Figure '06 shows that no Staff member has a Probationer as his or her senior.

Figure B/08.05: New graphs depicting seniority of Assistant Lecturer

Figure B/08.06: A Probationer is no Staff member’s senior

Figure '07 models the fact that the most junior staff member gets priority funding, given no other factors. This can be seen from Figure '06, which would cause its supertype counterpart to be deiterated, specialising '[Staff member: *s]' to '[Probationer: *s]'. Assume the existence of a) Figure '08, and b) no other factor, that is '((Other Factor))'. These would cause their corresponding parts of Figure '07 to be deiterated. The remaining double negation around '[Priority funding request: *r]' causes that concept to be released out to level 0. As the '*r' referent of the priority funding request concept was coreferent with the deiterated funding request concept, it is thereby valid to show Figure '09.
Appendix B/08

Funding request: *rl
Priority funding request: *rl

Figure B/08.07: Probationer priority

Figure B/08.08: Probationer makes a funding request

Figure B/08.09: Probationer funding request determined to be priority one

Again the above is demonstrated by the following CARE output:

Command started at 3:32:41 pm.

New type or relation labels:

Other factor.
Priority funding request.
senior.
Staff member.
applicant.
Funding request.

lattice relation added:
Funding request << universal_type.

lattice relation added:
Staff member << universal_type.

lattice relation added:
Priority funding request << universal_type.
Appendix BIOS

lattice relation added:
   Other factor << universal_type.

lattice relation added:
   applicant << universal_rel.

lattice relation added:
   senior << universal_rel.

Thinking...

Thinking...

I did not know that:
{
   ( [Other factor:*0]
    );
   ( [Priority funding request:*1]
    );
   [Staff member:*2]-(senior)->[Staff member:*3]
   [Funding request:*1]-(applicant)->[Staff member:*3]
   ).

Tidying up...

Command completed at 3:32:55 pm.

Command started at 3:33:10 pm.

New type or relation labels:
   Probationer.

lattice relation added:
   Probationer << universal_type.

Thinking...

Thinking...

I did not know that:
{
   [Staff member:*4]-(senior)->[Probationer:*5]
 }. 

Tidying up...
Command completed at 3:33:18 pm.
Command started at 3:34:19 pm.

New type or relation labels:

- Lecturer.
- Senior Lecturer.
- Reader.
- Professor.

lattice relation added:

- Reader << universal_type.

lattice relation added:

- Professor << universal_type.

lattice relation added:

- Senior Lecturer << universal_type.

lattice relation added:

- Lecturer << universal_type.

Thinking...
Thinking...

I did not know that:

- [Reader:*9]-(senior)<-[Senior Lecturer:*8]-(senior)<-[Lecturer:*7]-(senior)<-[Probationer:*6],[senior]->[Professor:*10].

Tidying up...

conformity relation added:

- Reader :: *9.

conformity relation added:

- Professor :: *10.

conformity relation added:

- Senior Lecturer :: *8.

conformity relation added:

- Lecturer :: *7.

conformity relation added:

- Probationer :: *6.

Command completed at 3:34:43 pm.
Command started at 3:35:34 pm.
I did not know that:
[type:Funding request]-(subtype)->[type:Priority funding request].

Tidying up...

lattice relation added:
Priority funding request << Funding request.

Command completed at 3:35:51 pm.

Command started at 3:36:28 pm.

Thinking...

I did not know that:
[Funding request:*12]-(applicant)->[Probationer:*13];
( [Other factor:*11]
).

Tidying up...

conformity relation added:
Funding request :: *12.

conformity relation added:
Probationer :: *13.

Command completed at 3:36:46 pm.

Command started at 3:37:09 pm.

Thinking...

Thinking...

The statement:
[Priority funding request:*14]-(applicant)->[Probationer:*15].

is:

UNKNOWN because of unknown graphs:
[Funding request:*14]-(applicant)->[Staff member:*3].
[Funding request:*1]-(applicant)->[Staff member:*3].

(Priority funding request:*1)

(Staff member:*2)-(senior)->[Staff member:*3]

No (more) solutions

Command completed at 3:37:36 pm.

Command started at 3:38:20 pm.

New referents:

Person.

Thinking...

Thinking...

I did not know that:

(supertype)

<-[type:Senior Lecturer]
<-[type:Reader]
<-[type:Lecturer]
<-[type:Professor]
<-[type:Probationer]
->[type:Staff member];
[type:Staff member]-(supertype)->[type:Person].

Tidying up...

lattice relation added:
Person << universal_type.

lattice relation added:
Staff member << Person.

lattice relation added:
Probationer << Staff member.

lattice relation added:
Professor << Staff member.

lattice relation added:
Lecturer << Staff member.
lattice relation added:
  Reader << Staff member.

lattice relation added:
  Senior Lecturer << Staff member.

Command completed at 3:38:54 pm.

Command started at 3:39:19 pm.

Thinking...

Thinking...

conformity relation added:
  Priority funding request :: *12.

code specialised:
  [Funding request:*12];
to:
  [Priority funding request:*12].

The statement:
  [Priority funding request:*16]-
    (applicant)->[Probationer:*17].

is:

TRUE because:
0.    proof of:
    [Priority funding request:*16]-
      (applicant)->[Probationer:*17];
0.    is:

1.    deduction:
1.    known rule:
    ( [Funding request:*1]-
      (applicant)->[Staff member:*3];
    ( [Staff member:*2]-
      (senior)->[Staff member:*3]
    );
    ( [Priority funding request:*1]
    );
    ( [Other factor:*0] )
  );
so:

([Priority funding request:*16]
[Funding request:*16]-
(applicant)->[Staff member:*3];
([Staff member:*2]-
(senior)->[Staff member:*3]
);
([Other factor:*0]
);

proving antecedent:
[Funding request:*16]-
(applicant)->[Staff member:*3];

proof of:
[Funding request:*16]-
(applicant)->[Staff member:*3];

is:

fact:
[Funding request:*12]-
(applicant)->[Probationer:*13];

proving antecedent:
([Staff member:*2]-
(senior)->[Probationer:*13]
);

proof of:
([Staff member:*2]-
(senior)->[Probationer:*13]
);

is:

fact:
([Staff member:*4]-
(senior)->[Probationer:*5]
);

proving antecedent:
([Other factor:*0]
);
1. proof of:
   ( [Other factor:*0] )
   is:

2. fact:
   ( [Other factor:*11] )

1. results:
   [Priority funding request:*12];

0. yielding fact:
   [Priority funding request:*12]--
   (applicant)-->[Probationer:*13].

Command completed at 3:40:08 pm.

Lastly, Figure '10 models the funding position for a staff member who can be more senior than other staff. The figure shows that these staff get priority funding provided it can be shown that any of his or her juniors have not made a funding request, and given no other factor.

![Diagram](image)

Figure B/08.10: Non-Probationer priority
B/09: The Many 'if-then' Rule Equivalents of Guideline 'a'

NB: The following is not an exhaustive list. There are many more evident permutations than shown here!

If Applicant for 'Funding request': *r is Probationer and not 'Other factor'
then Applicant for 'Priority funding request': *r is Probationer.

If Applicant for 'Funding request': *r is Lecturer and not Applicant for 'Funding request': *r is Probationer and not 'Other factor'
then Applicant for 'Priority funding request': *r is Lecturer.

If Applicant for 'Funding request': *r is 'Senior Lecturer' and not Applicant for 'Funding request': *r is Probationer and not Applicant for 'Funding request': *r is Lecturer and not 'Other factor'
then Applicant for 'Priority funding request': *r is Lecturer.

If Applicant for 'Funding request': *r is Reader and not Applicant for 'Funding request': *r is Probationer and not Applicant for 'Funding request': *r is Lecturer and not Applicant for 'Funding request': *r is 'Senior Lecturer' and not 'Other factor'
then Applicant for 'Priority funding request': *r is Reader.

If Applicant for 'Funding request': *r is Professor and not Applicant for 'Funding request': *r is Probationer and not Applicant for 'Funding request': *r is Lecturer and not Applicant for 'Funding request': *r is 'Senior Lecturer' and not Applicant for 'Funding request': *r is Reader and not 'Other factor'
then Applicant for 'Priority funding request': *r is Professor.

**

If Applicant for 'Funding request': *r is Probationer
then ('Priority funding request': *r or 'Other factor').

If Applicant for 'Funding request': *r is Probationer
then if not 'Priority funding request': *r
then 'Other factor'.

If Applicant for 'Funding request': *r is Probationer
then if not 'Other factor'
then 'Priority funding request': *r.

If Applicant for 'Funding request': *r is Lecturer
then (Applicant for 'Funding request': *r is Probationer or 'Other factor' or 'Priority funding request': *r).

If Applicant for 'Funding request': *r is Lecturer
then if not Applicant for 'Funding request': *r is Probationer then ('Other factor' or 'Priority funding request': *r).
If Applicant for 'Funding request': *r is Lecturer then
  if not 'Other factor'
  then (Applicant for 'Funding request': *r is Probationer or 'Priority funding request': *r).

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Priority funding request': *r
  then (Applicant for 'Funding request': *r is Probationer or 'Other factor').

If Applicant for 'Funding request': *r is Lecturer then
  if not Applicant for 'Funding request': *r is Probationer
  then if not 'Priority funding request': *r
  then 'Other factor'.

If Applicant for 'Funding request': *r is Lecturer then
  if not Applicant for 'Funding request': *r is Probationer
  then if not 'Other factor'
  then 'Priority funding request': *r.

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Other factor'
  then (Applicant for 'Funding request': *r is Probationer or 'Priority funding request': *r).

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Other factor'
  then if not 'Priority funding request': *r
  then Applicant for 'Funding request': *r is Probationer.

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Other factor'
  then if not Applicant for 'Funding request': *r is Probationer
  then 'Priority funding request': *r.

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Priority funding request': *r
  then (Applicant for 'Funding request': *r is Probationer or 'Other factor').

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Priority funding request': *r
  then if not Applicant for 'Funding request': *r is Probationer
  then 'Other factor'.

If Applicant for 'Funding request': *r is Lecturer then
  if not 'Priority funding request': *r
then
  if not 'Other factor'
  then Applicant for 'Funding request':*r is Probationer.

**

If not 'Priority funding request':*r
then
  if Applicant for 'Funding request':*r is Probationer
  then 'Other factor'.

If not 'Priority funding request':*r
then
  if not 'Other factor'
  then not Applicant for 'Funding request':*r is Probationer.

If not 'Priority funding request':*r
then not (Applicant for 'Funding request':*r is Probationer
  and not 'Other factor').

If not 'Priority funding request':*r
then
  if not Applicant for 'Funding request':*r is Probationer
  and not 'Other factor'
  then not Applicant for 'Funding request':*r is Lecturer.

**

If not 'Other factor'
then
  if Applicant for 'Funding request':*r is Probationer
  then 'Priority funding request':*r.

If not 'Other factor'
then
  if not 'Priority funding request':*r
  then not Applicant for 'Funding request':*r is Probationer.

If not 'Other factor'
then not (Applicant for 'Funding request':*r is Probationer'
  and not Priority funding request':*r).

**

Not (Applicant for 'Funding request':*r is Probationer
  and not 'Priority funding request':*r
  and not 'Other Factor').

**

If not 'Priority funding request':*r and not 'Other factor'
then not Applicant for 'Funding request':*r is Probationer.
B/10: Graphs Partially Analysing the 'Other factor' Dimension in the Expert 1 Problem

Figure B/10.01: For the most junior staff member, representing that the resultant priority funding request is subject to any 'fund recinding event'.

Figure B/10.02: For the most junior staff member, showing that the priority cannot be determined until the funding request is an eligible one. *(Alternative to Figure B/10.01).*
The above figures suggest the direction in which the ‘other factor’ dimension in guideline ‘a’ of the Expert 1 problem may be refined, and should be self-explanatory. Note however that Figure B/10.04 and Figure B/10.05 are the same as graphs Figure B/06.09 and Figure B/06.10 respectively (See
Appendix B/06), except that the label 'Eligible funding request' has replaced 'Acceptable funding request' in each. An alternative, if considered appropriate, to re-drawing these graphs as such would be to add the following graphs instead:

![Graphs](image)

Similarly, for Figure B/10.05, the following graphs show the relationship between eligibility and priority funding:

![Graphs](image)

Example scenarios are illustrated by the graphs in Figure B/10.06, Figure B/10.07, Figure B/10.08 and Figure B/10.09.

![Graph](image)

Figure B/10.06: Query: Does a Probationer obtain priority funding for '#13'?
Appendix B110

Funding request: #13

subject

Research heading: Special

characteristic

Exceptional Circumstance: #1

Previously funded request: #6

Figure B/10.07: Request '#13' is an 'only in exceptional circumstances...' (re: guideline 'b')

Priority funding request: #7

applicant

Probationer

Figure B/10.08: Query: Does a Probationer obtain priority funding for '#7'?  

Funding request: #7

subject

Conference: Concepto '93

location

Place: UK

Figure B/10.09: Request '#7' is a funding request for a UK conference. (re: guideline 'g')
B/11: Pro Forma Browse Graphs

**Drawing sheet title: 'Supertype query'**

```
|type:*b| ← supertype → |type:*a|
```

**Drawing sheet title: 'Subtype query'**

```
|type:*a| ← subtype → |type:*b|
```

**Drawing sheet title: 'Super-relation query'**

```
|rel:*b| ← superrel → |rel:*a|
```

**Drawing sheet title: 'Sub-relation query'**

```
|rel:*a| ← subrel → |rel:*b|
```

**Drawing sheet title: 'True concepts query'**

```
universal_type
```

**Drawing sheet title: 'True relations query'**

```
universal_type:*a ← universal_rel → universal_type:*b
```

**Drawing sheet title: 'False concepts query'**

```
absurd_type
```

**Drawing sheet title: 'False relations query'**

```
absurd_type:*a ← absurd_rel → absurd_type:*b
```
From: Simon Polovina Date of session: Thursday, 18th February 1993

Conceptual Analysis and Review Environment (CARE)
Usability Evaluation Session at Melton Borough Council: Questionnaire

Thank you for participating in the above session. I would be very grateful if you could indicate on this form to what extent you agree or disagree with the statements below, and return same to me. Please select your answers by **one choice only per statement** from the following continuum:

```
-3  -2  -1  0  1  2  3
Strongly disagree   Increasing Indifferent   Increasing ->   Strongly agree
```

(e.g. ‘-1’ means you mildly disagree; ‘2’ that you substantially agree)

Each statement includes the above continuum, so you need only circle your choice. However if you wish to add any further comments, please do so. If you have any questions please do not hesitate to ask me. It would be very helpful for my analysis if you might be able to write down your name and job title, both of which will be kept in the strictest confidence. Thank you again.

Name / Job Title (optional): ____________________________

**Statements:**
*(Please read each one carefully before answering)*

1. I am happy working with computers:
   -3  -2  -1  0  1  2  3

2. I am computer literate:
   -3  -2  -1  0  1  2  3

3. In accounting, computers are an improvement over manual methods because they
   a) provide more accurate information:
      -3  -2  -1  0  1  2  3
   b) provide more timely information:
      -3  -2  -1  0  1  2  3
c) are easier to cope with than manual methods:

\[-3 -2 -1 0 1 2 3\]

d) have helped me with my job:

\[-3 -2 -1 0 1 2 3\]

e) have made my job more interesting:

\[-3 -2 -1 0 1 2 3\]

4. A computer will replace my job within the next ten years:

\[-3 -2 -1 0 1 2 3\]

5. I agree that “If A then B” also means “If not B then not A”:

\[-3 -2 -1 0 1 2 3\]

6. I understand that

\[\begin{array}{c}
\text{B} \\
\end{array}\]

is the same as “not B”:

\[-3 -2 -1 0 1 2 3\]

7. I understand that

\[\begin{array}{c}
\text{A} \\
\text{B}
\end{array}\]

is the same as “If A then B”, and thereby “If not B then not A”:

\[-3 -2 -1 0 1 2 3\]

8. I understand that

\[\begin{array}{c}
\text{F} \\
\text{G} \\
\text{H}
\end{array}\]

is the same as “If F and not G then H”, “If F and not H then G”, and “If not G and not H then not F”:

\[-3 -2 -1 0 1 2 3\]
9. I understand supertypes and subtypes - e.g. Bad Customer << Customer (Bad Customer is a subtype of Customer, or Customer is a supertype of Bad Customer):

10. I understand that

Customer: Mr. Warwound

source

'Keeps-the-VAT' Payment

is an example of a customer who won't pay the VAT element due:

11. From

Customer:*x  Bad Customer:*x

source

'Keeps-the-VAT' Payment

I can see that, given the circumstance in '10.' above, Mr. Warwound would be a bad customer:

12. However should

Bad Customer: Mr. Warwound

have been true I can see that Mr. Warwound cannot be withholding his VAT, or be of any other cause that makes him into a bad customer:
13. I can get to grips with all the dimensions that embodies:

-3 -2 -1 0 1 2 3

14. CARE makes the dimensions of problems more apparent than
a) written reports with their well-known ambiguities and obfuscations:

-3 -2 -1 0 1 2 3

b) 'If-Then' rules:

-3 -2 -1 0 1 2 3
c) my brute intuition:

-3 -2 -1 0 1 2 3

15. CARE is better than spreadsheets for
a) adding rows and columns of figures:

-3 -2 -1 0 1 2 3

b) day-to-day accounting:

-3 -2 -1 0 1 2 3
c) financial accounts:

-3 -2 -1 0 1 2 3
d) management accounts and budgeting:

-3 -2 -1 0 1 2 3
e) strategic high-level issues:

16. I can see the similarity between CARE and structured diagram techniques, such as flowcharting:

17. CARE is better than flowcharts for
a) adding rows and columns of figures:

b) day-to-day accounting:

c) financial accounts:

d) management accounts and budgeting:

e) strategic high-level issues:

18. I perceive that negation as presented in CARE, i.e.

\[
\text{to be the same as negating figures in accounting - i.e } '(<\text{figure}>)' \text{ to give 'Credit' values:}
\]

19. CARE can capture all my knowledge and thereby replace me:

20. It is apparent to me that CARE can only model trivial problems. Real problems make it too complicated to follow:
21. Concepts do not, in reality, always fit into a neat hierarchy of ‘supertypes’ and ‘subtypes’:

   -3   -2   -1   0   1   2   3

22. You would have to tell CARE too much for it to be of any practical use:

   -3   -2   -1   0   1   2   3

23. All problems can be represented by numbers alone:

   -3   -2   -1   0   1   2   3

24. CARE would make my job more interesting:

   -3   -2   -1   0   1   2   3

   (End of statements)

Please write below any other comments you may wish to make:
Figures B/13.01 and '02: Highlighted scope of example linear-form negative contexts in 'cgp - status report', using 'Edit' menu 'Balance' command.
**B/14: Summary of Questionnaire Responses from Melton Session**

Key: Q = Question; n/a = The subject did not answer this question.

<table>
<thead>
<tr>
<th>Q:</th>
<th>1</th>
<th>2</th>
<th>3a</th>
<th>3b</th>
<th>3c</th>
<th>3d</th>
<th>3e</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14a</th>
<th>14b</th>
<th>14c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subjects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>-2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>-2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>2</td>
<td>2</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-2</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>-3</td>
<td>3</td>
<td>3</td>
<td>-3</td>
<td>-3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>-1</td>
<td>3</td>
<td>3</td>
<td>-1</td>
<td>-1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>F</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>-2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

**Mean:**
- 2.3 2.1 2.3 2.3 2.2 -1.8 2.7 2.7 0.3 -0.2 2.5 2.8 2.5 0.7 -0.3 -0.6 -0.4 -0.4

**Median:**
- 2 2 2 2 2 2 2.5 2.5 -2 3 3 0.5 -0.5 3 3 3 3 1 0 0 0 0

**Standard Deviation:**
- 0.5 0.6 1.2 0.5 1 1 1 0.8 0.5 0.5 2.3 2 0.8 0.4 0.8 2.4 1.4 0.9 1.1 1.1

**Minimum:**
- 2 1 0 2 0 1 1 -3 2 2 -3 -3 1 2 1 -3 -2 -2 -2 -2

**Maximum:**
- 3 3 3 3 3 3 3 -1 3 3 3 3 3 3 3 3 3 3 3 3 1 0 1 1

343
<table>
<thead>
<tr>
<th>( \theta ):</th>
<th>15a</th>
<th>15b</th>
<th>15c</th>
<th>15d</th>
<th>16</th>
<th>17a</th>
<th>17b</th>
<th>17c</th>
<th>17d</th>
<th>17e</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subjects' Answers:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>2</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>-3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-3</td>
<td>-3</td>
</tr>
<tr>
<td>B</td>
<td>-3</td>
<td>-1</td>
<td>-2</td>
<td>-1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>-2</td>
</tr>
<tr>
<td>C</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>D</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>3</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>1</td>
<td>-3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-3</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td>2</td>
<td>-1</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td>0</td>
<td>2</td>
<td>-3</td>
<td>3</td>
<td>-2</td>
<td>3</td>
<td>-3</td>
<td>-1</td>
</tr>
<tr>
<td>F</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>-3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>-3</td>
<td>0</td>
</tr>
<tr>
<td>Mean:</td>
<td>-2.2</td>
<td>-1.8</td>
<td>-2</td>
<td>-1.8</td>
<td>0.2</td>
<td>1.5</td>
<td>-0.2</td>
<td>-0.6</td>
<td>-0.6</td>
<td>0</td>
<td>1.2</td>
<td>-2.5</td>
<td>2.2</td>
<td>0.7</td>
<td>1.7</td>
<td>-2.2</td>
<td>-1</td>
</tr>
<tr>
<td>Median:</td>
<td>-3</td>
<td>-1</td>
<td>-2</td>
<td>-1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-3</td>
<td>3</td>
<td>0.5</td>
<td>1.5</td>
<td>-3</td>
</tr>
<tr>
<td>Standard Deviation:</td>
<td>1.1</td>
<td>1.1</td>
<td>1</td>
<td>1.1</td>
<td>1.1</td>
<td>1.6</td>
<td>0.4</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>0.7</td>
<td>1.2</td>
<td>0.8</td>
<td>1.3</td>
<td>1.8</td>
<td>1.2</td>
<td>1.6</td>
</tr>
<tr>
<td>Minimum:</td>
<td>-3</td>
<td>-3</td>
<td>-3</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-2</td>
<td>-2</td>
<td>-2</td>
<td>-1</td>
<td>0</td>
<td>-3</td>
<td>0</td>
<td>-2</td>
<td>0</td>
<td>-3</td>
<td>-3</td>
</tr>
<tr>
<td>Maximum:</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>-1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

344
B/15: Significant Spearman \ Kendall Correlations from Melton Session

Key:
One-tailed test, $p \leq 0.05$.
Two-tailed test, $p \leq 0.05$ (When bold values are disregarded).
Shown as "<Spearman figure> <Kendall Figure>", except where the values are exactly the same. In such instances only one figure is shown.
ns = not significant.

(Tables appear overleaf)
<table>
<thead>
<tr>
<th>Q</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14a</th>
<th>14b</th>
<th>14c</th>
<th>15a</th>
<th>15b</th>
<th>15c</th>
<th>15e</th>
<th>16</th>
<th>17a</th>
<th>17b</th>
<th>17c</th>
</tr>
</thead>
<tbody>
<tr>
<td>3b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>.85%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15d</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>.83%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17d</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>.86</td>
<td></td>
<td>.79</td>
<td>.79</td>
<td>.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>-1.00</td>
<td></td>
<td>.85</td>
<td>.80</td>
<td>.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>-.82</td>
<td></td>
<td>-.85</td>
<td></td>
<td>-.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td></td>
<td></td>
<td>-.87</td>
<td></td>
<td>-.79</td>
<td>.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>-.85</td>
<td></td>
<td>-.85</td>
<td></td>
<td>.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>-1.00</td>
<td></td>
<td>-.85</td>
<td></td>
<td>.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
<td>-.85</td>
<td></td>
<td>.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(continued overleaf)
C/01: "Towards the Thought Machine"

Accountants, as we all know, play key roles in most organisations. They assemble raw data into meaningful information for both external and internal consumption.

This is not a trivial task. Accountants add expert judgment. Without it the data is at best worthless and at worst dangerously erroneous to the various outside interests and the organisation's own decision-makers.

The advent of processing data by computer has had a significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a removal of remarkably tedious tasks.

This is not a trivial task.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a
doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

doubly significant impact on the role of accountants. Many will state this benefit has occurred in the form of achieving more accurate information and doing so in a shorter period of time.

Most of us will also take comfort from more closely felt benefits based on the removal of remarkably tedious tasks.

Trial balances that balance automatically, monthly accounts provided in an instant by computer has had a

The point is that, armed with a computerised spreadsheet like Lotus 1-2-3, this effort now could be performed so easily that it would have been possible to produce the whole item on a monthly basis.

Qualitative aspects

However, computers cannot perform everything. The conceptual roots of current computer software lie in numeric and data-based relationships. The remaining vital qualitative aspects still have to be processed and explained manually.

For instance, computer

Imagine I now present my director with the output of the computerised spreadsheet model. Already I, not the computer, will have arranged the information into what I believe to be the most meaningful way. The results of calculations which I consider important will be highlighted, maybe in summary form on the first page. Less important ones and details will be relegated to subsequent pages or appendices. Insufficient items are not included at all. I design any charts similarly.

The director asks the not untypical kind of question: 'I know these figures will not be accurate because some buildings may be sold or demolished, others purchased or improved. Can you give me a revised report accounting for these possibilities?'

To satisfy this, the report invariably now needs to contain, in words, the reasons why some of the figures no longer follow the general pattern in the spreadsheet.

Adding separate columns alone for these movements is insufficient. The director would focus on them and say 'Tell me how you arrived at these values'. The reasons would reveal explicitly the appropriate responsible officials' (and my) subjective judgments which determine the context from which the figures are derived.

This example highlights how problems require solutions not based on calculations alone. The judgments stem from knowledge gained from our life experiences and professional training. Knowledge is what the computer does not yet have.

Mathematical techniques have attempted to model knowledge. Operational Research (OR) and probability theory are good examples. Russell Ackoff, a key proponent of OR, wrote an article in 1987 entitled President's symposium: OR, a post mortem. The title tells the story.

Herbert Simon, the management scientist, back in the 1960s, recognised that the data needed to ensure that these optimising models were realistic just could not be obtained, or involved im-

TOWARDS THE THOUGHT MACHINE

and 'exception' reports for managers are but a few familiar examples.

I remember my previous employment in a large multinational public company. I was responsible for the financial and management accounts of the property subsidiary which held all the multinational's buildings. Part of the reporting requirement to the top board directors involved producing an enormous 'manual' spreadsheet.

spreadsheets are invaluable as flexible decision support tools but the accountant still has to relate manually the context under which the figures and charts are produced. Its form is usually verbal, or in an attached text-based report, or both.

The closest these areas come to being computerised is by being word-processed. What do I mean by all this? Let us continue with my spreadsheet example above.

Does 'artificial intelligence' pose a threat to the accountant? Simon Polovina believes that it will actually enhance his value.
possible quantities of calculation combinations. He saw real decisions being based on what could be found out or reasoned sensibly, calling these 'satisficing' models.

If that is insufficiently convincing, Goedel, the Austrian mathematician, further back in 1931, demonstrated his 'incompleteness theorem'. This showed that the theory on which mathematics itself is based is incomplete, and so its outcomes can never be completely trusted. In summary, it is best to be approximately right than precisely wrong.

A large body of current evidence suggests another approach involving artificial intelligence techniques as evidenced in practice by the emergence of computer-based expert systems. These instead attempt to replicate the judgmental human reasoning process, or heuristic, in the computer.

We live by heuristics because of the shortcomings of mathematical-like rigidity as indicated above. Say we hear: 'If it rains and I am outside, then I will get wet', or: 'If a customers of ours does not pay us within 30 days, then we give them no further credit'.

We just accept these statements as true because of our experiences and training.

Heuristics can never be precise as they are confounded easily by exceptions: 'If it rains in Yorkshire and I am outside in London, then I will not get wet by the rain in Yorkshire', or: 'If Eastern Company barter our goods for theirs, then we give them 30 days' extra credit to allow for shipping delays in sending their supplies to us'.

These statements will have exceptions in turn.

The computer can utilise heuristics by the rules of logic. Aristotle, the Greek philosopher who lived from 384 to 322 BC, proved that in any 'if-then' scenario the 'then' part is true provided the 'if' part is true. Should a 'then' part be false, so will its 'if' part.

There are usually many 'ifs' to a single 'then'. Extending the customer example above, credit is refused if that customer is bankrupt, or had a poor credit rating from analysis of his accounts, or so on. Hence a customer who has credit is in none of these categories.

The rules also mean a false 'if' does not make a false 'then' which, in fact, is simply not asserted. For instance, 'If Western Company is not bankrupt' cannot mean 'then credit to Western Company is granted', because that customer may fall under some other criterion for credit refusal.

An expert system basically involves 'chaining' together these 'if-then' rules in the same way a human expert would. In particular, an asserted 'then' would cause other 'if-then' rules to operate provided their 'if' part matches the 'then' part in the previous rule.

This can be illustrated by another example rule: 'If a customer is refused credit, then that customer's future purchases must be paid for by cash on sale'.

Hence, by chaining this rule to the earlier non-paying rule it is possible to state: 'If Southern Company has not paid within 30 days, then Southern Company must pay cash for new purchases'.

As humans perform the reasoning anyway, why use expert systems? There are two strands to answering this question. The first is that human experts are rare and that existing human expertise may leave the organisation through a change in employment or death. Hence an expert system helps to obtain, spread and retain this valuable commodity within the enterprise.

The second is overcoming what is known as humans' 'cognitive overload'. Psychologists have long recognised that there is a limit on how much humans can reason simultaneously before they become mentally overloaded. Experiments have shown this as generally between five and nine items.

An expert system is freed from this constraint because computers are not hindered in this direction.

In summary, these knowledge-based systems are computer models of the human experts' heuristics unhindered by personal limits.

Institute Fellowship

The ICAEW was so concerned about the impact of expert systems in accounting that it established a research fellowship in the department of accounting and finance at Southampton University in 1985.

The outcome was the publication of Expert Systems in Accounting, written by Edwards and Connell in 1989. This surveyed the use of expert systems in accountancy and concluded that they will have an important role to play. Expert systems have saved organisations up to many millions of pounds already. Examples of accounting expert systems in use include the following:

- Auditor: Provides assistance in improving the consistency of adequate bad debt provision decisions.
- Expertax: Developed by Coopers & Lybrand in the US, this helps to evaluate the application of new tax laws to the firm's clients. The system incorporated the knowledge of the firm's senior tax partners.
- Clique: Also from Coopers & Lybrand, this helps to minimise the tax liabilities of multinationals companies.
- Planpower: Helps financial planners develop suitable schemes for high-income personal and organisational clients.
- Vatia: A VAT expert system from Ernst & Young. It collects information efficiently and processes it to identify important VAT issues.

In conclusion, however, knowledge-based systems are still mainly being researched and developed, so have yet to create any large-scale effect. A notable sphere of potential application is in management accounting.

What effect will their success have on the accounting technician? One cynical view is that it will become a redundant profession because expert systems, by encoding their expertise, will de-skill the work AT's do.

This that can be avoided. A necessary feature of expert systems is that they must be able to display their paths of reasoning to users. Why? Because the imprecise heuristic fabric on which they are based will need continual affirmation and updating by those conversant with the actual problems being modelled.

Computer scientists are not accountants. Hence an ideal opportunity arises for accounting technicians to acquire this responsibility because they manage the key crossroads of accountancy data, information and knowledge. It will be an irresistible challenge.

The author, an MA, is researcher in the department of computer studies at Loughborough University of Technology.
C/02: "Tomorrow's Spreadsheets: Conceptual Graphs as the Knowledge Based Decision Support Tool for the Management Accountant"

Abstract

This paper presents conceptual graphs as the means by which management accountants are able to evaluate qualitative business problems. Management accountants can, using conceptual graphs, structure concepts and relationships beyond that of numeric and data-based decision support tools such as the spreadsheet, which triumphed because users were conversant with them manually. The same success is possible with conceptual graphs. A small example shows how conceptual graphs can benefit the management accountant. The paper assumes how conceptual graphs can benefit the management accountant. However, the paper presumes a fundamental understanding of conceptual graphs theory.

Introduction

Management accountants strive continually to provide timely, accurate and meaningful performance gauges to business managers. Yet, although current decision support tools such as spreadsheets are flexible, their conceptual roots lie in numeric and data-based relationships. Many important business problems demand solutions based on subjective judgement. Many writers, like Graff (1989), acknowledge the fact that these qualitative evaluations still have to be processed and explained manually.

A large body of current evidence, catalogued by Mahahon (1990), suggests another approach which should be knowledge-based. This is demonstrated by the emergence of accounting expert systems. Despite this, any successes to date have been restricted to solving a specific range of easily definable problems.

Much of the literature, typified by Keravnou and Washbrook (1989), discuss this limitation in terms of overcoming the technical inflexibilities within the various expert system formalisms. Although this is a worthwhile pursuit, there is a better answer to this issue than seeking improved knowledge representation formalisms alone.

Gill (1988) argues convincingly for a user-centred approach in expert system research and development. Technically-centred expert systems attempt to
capture every element of knowledge that makes up expertise. Gill contends this will never be achieved because of the great difficulty in modelling the tacit and implicit parts of knowledge. The model of the user in this scenario is the non-expert. In contrast, user-centred expert systems capture only those easily modelled parts of knowledge and relies on the expert user to provide the remaining more difficult parts of knowledge when interacting with the expert system. However, the danger in following a strictly user-centred line of analysis is the outcomes cease to be knowledge-based systems. Liang (1990) views the user-centred approach as a necessary but temporary stage towards fully fledged expert systems in the multi-disciplinary area of management.

Therefore, an ideal expert system would deploy a powerful enough form of knowledge representation which captures faithfully these more difficult knowledge elements yet remains user-centred. This would be achieved if the advanced representation could be built by expert users directly, according to their own mental models. Such expert systems would gain user acceptance because they would provide supportive instead of prescriptive solutions. Computer spreadsheets are successful because users were conversant with how they worked manually beforehand.

Conceptual graphs have been devised by Sowa (1984) in a remarkably principled way. Knowledge is highly structured by modelling specialised facts which can be subjected to generalised reasoning. This is due to the supporting of advanced knowledge forms which deal with the leading edge topics of inheritance, background knowledge, and numeric and database interfacing. Hence, technically, conceptual graphs are very sophisticated. The distinct advantage of conceptual graphs is that they can be highly user orientated as well. This is because both factual and reasoning knowledge are represented in a visual form which lends itself to being clearly understandable by users. Therefore, conceptual graphs offer human mental model support because they could be exploited by the user directly.

The foregoing leads to the author's research, the thesis of which is conceptual graphs become an important knowledge based decision support tool in management accountancy. A major survey carried out by Edwards and Connell (1990) does not list any management accounting applications of expert systems. Buckner and Shah (1989) recognise that expert systems have yet to be sufficiently developed before they can get to grips with management accountancy problems. Buckner and Shah express sentiments which support the approach taken in this paper. Work carried out by carried out by Garner and Tsui (1985) in auditing; Garner (1987); Billo, Henderson, and Rucker (1990) in engineering analysis; and Slagle, Gardiner, and Han (1990) demonstrate encouraging results in respect of conceptual graphs as a knowledge specification tool.
Management accountants could accept the use of conceptual graphs because they have the potential to be:

1. **Familiar**: Accountants employ procedural flowcharts already as an advantageous visual aid to structuring problems and their solutions. Conceptual graphs can be defined as the declarative equivalent, and hence overcome user resistance.

2. **Simple**: To see if a solution applies to a given problem, management accountants can 'pattern match' general solution graphs onto the specialised problem ones.

3. **User-Centred**: The computer can speed up management accountants' manual graph operations yet retain the user-centred support role as in the computerised spreadsheet.

The purpose of this paper is to show one way the above thesis would be fulfilled. To do this, the paper focuses on one pertinent part of management accounting that cannot be modelled on a spreadsheet. This is "Relevant Costs and Benefits for Decision Making".

**Relevant Costs and Benefits for Decision Making**

Horngren and Foster (1990) advise that all decisions involve predictions. Nothing can be done to alter the past. Any historical costs or benefits are irrelevant, although they may be the best available basis for future values. Bierman, Dyckman and Hilton (1990) add the costs or benefits determined by accounting information systems cannot replace judgement in choosing the best course of action. Coulthurst and Piper (1986) recognised relevant values do not represent techniques but *concepts*, and call for a clearly structured conceptual framework. This paper reveals how conceptual graphs can help by the following basic example. For the benefit of the reader, supporting explanations are given throughout to clarify unfamiliar problem domain terms and relationships.

**Example: Scrupulous Chemicals Company**

Scrupulous Chemicals Company uses the chemical "GGS" to make one of their products. The manager in charge of the department involved provides the following comments:

"As regards the GGS in stock, the accounts show we bought it for £20,000. It would cost £30,000 to replace, and if we don't use it, it's environmentally hazardous, but we can sell it for £16,000 to Mr. Nasty".
On which figure should the Scrupulous Chemicals management base its production decisions?

This query is passed to Scrupulous Chemicals' management accountant who is expected to evaluate such problems. The management accountant thinks this problem can benefit from using conceptual graphs, so decides to structure it accordingly. To begin, the graphs representing the query, Figure 1, and the statement, Figure 2, are drawn, and the type lattice, Figure 3, defined for those types not a direct subtype of "Universal" for the purposes of this example.

The statement graph, Figure 2, may be read as "The GGS #7568 has:

a) the disposal buyer who is the sole trader Mr. Nasty,
b) the characteristic of being an environmental hazard,
c) the historic measure £20,000,
d) the replacement measure £30,000,
e) and the disposal measure £16,000".

The last figure is enclosed in parenthesis in the graph because it is income rather than expenditure. This accords with accounting convention.

The query graph, Figure 1, may be read as "The relevant measure of the GGS #7568 is some £". It is this latter concept for which a referent must be found. Should this not be possible, then none of the three £ measures in the statement graph, Figure 2, is relevant.
These graphs may not appear to match precisely the statement or query sentences. The reason is the structure necessary in forming the graphs requires the tacit and implicit knowledge in the sentences to be made explicit. Mr. Nasty is a referent to a type label. This could have been left as 'legal person', but the management accountant was aware Mr. Nasty was a sole trader. Therefore, the conformity is to the most specialised type label because it contains more knowledge. £20,000 is related by historic measure to the GGS because the management accountant knew accounts figures are always historic.

Of course, relationships such as between 'accounts' and 'historic' could be modelled as well. Once a definition is made explicit, the user can refer to its singularly defining subtype or sub-relational label in an implicit but not ambiguous way. The management accountant is free to design the graphs in whatever way, as user, is thought appropriate. Subsequently, although not shown in this example, such items may in fact be included. Should such actions happen to cause an inconsistency, the user may wish to re-think his or her own mental conceptual model. The overall point of this paragraph is that the conceptual graphs model has demanded the user gives an immediate structure to the implicity and ambiguity of natural language interpretations. The user's model of the problem is not prescribed, rather the user clarifies his or her understanding of the problem. A rule-based representation could 'fudge' these inconsistencies by, say, writing some 'patching up' rules. Conceptual graphs make this much harder to do. This ensures the user is not lead astray.

To continue, an attempt is made to see if the query graph, Figure 1, can be projected onto the statement graph, Figure 2, and thereby assert its truth and discover the relevant value. This cannot be achieved. As the graphs are in their most specialised form, type or relational expansion may be possible. This will reveal underlying explicit knowledge from where there is perhaps common ground for projection to occur. As it happens, the relevant measure relation in the query graph, and the three measure relations and the disposal buyer relation in the statement graph can be expanded to give the redrawn query graph, Figure 4, and statement graph, Figure 5. The extended type lattice, Figure 6, is drawn accordingly.

The expanded query graph, Figure 4, asks if GGS is the subject of a relevant value which has a measure in units of £. Should the query succeed, this value will have an unique referent. The £ measures in this example happen to be unique, but in any realistic scenario this would be unlikely. The individual relevant value referent allows these cases to be distinguished.

Examining the expanded statement graph, Figure 5, which happens to be now in four parts, also reveals detailed underlying knowledge. The graph could be left as one large structure, but this makes it difficult to read. The graph remains valid in this form because they are all coreferent on the concept
Projections can span over more than one graph accordingly. The ' % <words> ' are simply comments.

For each sub-relation of measure in the original graph, Figure 2, the expanded graph, Figure 5, shows that value is determined by an event called a 'GO_poss', which means a change of possession. The value has a £ measure. The earlier 'allows to be distinguished' discussion of value's subtype, relevant value, is also valid for value.

For the replacement and disposal cases, the GO_poss is part of a replacement and disposal transaction event respectively. The GO_poss concepts are part of transactions because a transaction would include some other GO_poss, which would reflect what is exchanged for the GGS. The subject of this GO_poss is usually banknotes, a cheque, a creditor or debtor obligation. The historic case transaction does not appear from the historic measure expansion, but if it did the supertype concept [Transaction] would be attached with the particular GO_poss via a part relation from the former to the latter. This is because historic measure occurs from any type of transaction.

The subject of the GO_poss is the GGS, which is the value subject too. The graph has these relationships in its shape because the management accountant is aware that change of possession realises value. A tradable entity does not have a value in its own right. It acquires value solely because some person or organisation

---

**Figure 4:**
Query graph (expanded)

**Figure 5:**
Statement graph (expanded)

**Figure 6:**
Extended type lattice

Types not linked to supertypes and / or subtypes are direct subtypes of $T$ and supertypes of $\perp$ respectively. Explanations of type labels are provided, as appropriate, in the course of the main text.
is willing to give up something else for it. Once again, conceptual graphs makes the implicit knowledge in a statement such as "GGS has a value of £X" to be made explicit.

Lastly, the GO_poss happens at a past or future point in time. The disposal buyer sub-relation shows the disposal instance's GO_poss subject destination as Mr. Nasty. This establishes the destination is that of the GGS. Simply linking the destination direct to the GGS would give graphs that would show GGS has multiple destinations. Clearly, this is not so. The destination relates to the appropriate GO_poss. Therefore, the relation is correctly drawn.

Another attempt at projection is made, but is unsuccessful as the [ Value ] concepts in the new statement graph is specialised to ['Relevant Value'] in the new query graph. According to Coulthurst and Piper (op cit) a relevant cost or benefit is avoidable by the alternatives in a decision. Including the implicit knowledge this becomes, "a relevant cost or benefit is a value determined by an avoidable 'GO_poss', an avoidable change of possession". Hence, the management accountant draws the appropriate type definition for 'Relevant Value', Figure 7.

The expanded type definition replaces relevant value in the query graph, Figure 4, to give the more expanded query graph, Figure 8. Another attempt at projection is made with Still, it is unsuccessful as the [ GO_poss ] concepts in the expanded statement graph, Figure 5, is specialised to ['Avoidable GO_poss'] in the more expanded query graph, Figure 8.

Type expansion of avoidable GO_poss is not possible because a GO_poss becomes avoidable when one, or more, prevailing contexts are successfully evaluated. This point is elaborated in the following paragraph.

The complex graphs, Figure 9, depict two such avoidable cases. These graphs represent Scrupulous Chemical's environmentally friendly policy. The top graph shows the company can avoid the GO_poss in the future of a tradable entity provided it is false that the tradable entity has a characteristic which is an environmental hazard. The bottom graph shows the company can nevertheless avoid the GO_poss of a tradable entity which is an environmental hazard provided it is true that the GO_poss subject destination is to an environmental supporter.
Taking the top graph, the simple graph in the odd outermost context can be projected into the statement graph, Figure 5. This means insertion and joining can take place such that deiteration can occur in the odd outermost context. A simpler way of stating this is the aforesaid simple graph can project into the statement graph and hence be deiterated. However, it is not possible to deiterate the false environmental hazard. In fact, the opposite characteristic of GGS happens to be true. Therefore, no progress is made here.

As for the bottom graph, the outermost graph can be deiterated completely in respect of the concept [GO_poss : #ii] which relates to the replacement transaction subgraph in the statement graph, Figure 5. The own company type label of Scrupulous Chemicals is a subtype of environmental supporter thereby revealing, amongst other features which distinguish it from being an environmental supporting company, the company’s care for the environment. The co-referent avoidable GO_poss concept is double negated and hence asserted as:

[ 'Avoidable GO_poss' : #ii ]

Considering the bottom graph again, the outermost graph could additionally be deiterated for the disposal measure, but the management accountant does not know about Mr. Nasty's environmental consciousness. After further investigation, the management accountant confirms that 'Mr. Nasty' is no environmental supporter. Hence, the following graph is drawn:

→ [ [ 'Environmental Supporter' : Mr. Nasty ] ]

This also means Mr. Nasty cannot conform to Environmental Supporter. Therefore, the deiteration does not take place.

Incidentally, if Mr. Nasty did happen to be an environmental supporter the following graph would have been asserted instead:
The query graph, Figure 7, is then re-projected into the statement graph, Figure 5. The projection succeeds for the replacement subgraph because the appropriate GO_pos concept conforms to avoidable GO_pos. This yields the result, Figure 10. This can be contracted to give the graph, Figure 11.

This may be contracted down further to the original query graph size. However, the management accountant may wish to make the relevant measure relation more suitably distinct to take account of the relevant value referent, #2. The relational definition 'relevant measure / #2' may be drawn. Using the new definition, the graph, Figure 12, may be produced.

The management accountant now can present the results of this evaluation to the company's management. Any queries on the result itself can be explained by going through the graphs, and any further 'what if' queries can be modelled by the management accountant without immediate recourse to rewriting the whole problem. Such restructuring that may occur will serve not to patch holes in, but clarify further, the management accountant's present conceptual framework.

Conclusion

A criticism of taking a conceptual graphs approach might be they have transformed what appears to be a simple problem modelling exercise into something horrendously complex. This accords with the reservations of Webster (1988). The complexity arises due to the need to explicate presumptions and background knowledge. However, the claim that this approach is user-centred is not contradicted. This is because although the above problem had to make all the appropriate implicit knowledge explicit, this exercise need occur only once. Given an increasing set of graphs, larger problems will require relatively fewer graphs as a graph library is built up by the user. Certain graphs would need re-defining, but this is to be expected in an open, real, world.
Using rule-based approaches may obtain initial results faster, but in the long run cannot capture sufficient knowledge. Patching up rules also conspire to thwart serious meaningful applications in qualitative domains. Predicate logic formalisms can capture much more but, with their style of notation and manipulation, cannot be considered user-centred. Dumb 'friendly front ends' fail to fully utilise the power of the underlying advanced but unfriendly formalisms. Beside providing a front end capability by a direct mapping into first-order predicate calculus, conceptual graphs is an advanced formalism in its own right.

The above small example may appear to be trivial, but it serves to illustrate a way that qualitatively based problems, numerous in management accountancy, could be modelled. Conceptual graphs have paved the way for the management accountant, as a professional, to have a decision support tool to hand that can structure concepts and relationships beyond that of a spreadsheet. Yet, like the spreadsheet, the computerisation of conceptual graphs commences from the manual equivalent. This assures the combination of technical advancement with the user-centred aspects that knowledge-based systems need to succeed.

References


Coulthurst, Nigel; Piper, John (1986a) "The State of Cost and Management Accounting", Management Accounting, April.

Coulthurst, Nigel; Piper, John (1986b) "The Terminology and Conceptual Basis of Information for Decision Making", Management Accounting, May.

Coulthurst, Nigel; Piper, John (1986c) "Framework for Analysis of Decision-Relevant Costs and Benefits", Management Accounting, June.


C/03: "An Introduction to Conceptual Graphs"

AI Expert, 7(5), May, pp. 36-43.
Conceptual graphs were devised by John Sowa from philosophical, psychological, linguistic, and AI foundations in a remarkably principled way. Knowledge in conceptual graph form is highly structured by modelling specialized facts that can be subjected to generalized reasoning. In a comprehensive comparative study carried out by Dallas Webster, conceptual graphs were acknowledged as the richest means of representing and manipulating expert knowledge in a computer model. However, Webster also warned that conceptual graphs seemingly extracted a too severe penalty in terms of the complexity encountered in using them. This fact would explain the relatively limited effect of conceptual graphs in any meaningful application areas, although there continues to be an active interest in this way of modelling knowledge (for instance, the annual workshops on conceptual graphs and the conceptual graph electronic forum on cg@cs.umn.edu. New subscribers are included by sending a request to tjan@cs.umn.edu).

The usability criticism comes up because Sowa's work leaves many loose ends: It was basically a stimulus for further research and development rather than a meticulous programmer's manual. This article attempts to redress the apparent complexity by clarifying the salient aspects of conceptual graphs theory. Included are some elementary algorithms, which indicate how a conceptual graphs program may be implemented.

CONCEPTS AND RELATIONS
Conceptual graphs are based upon the following general form:

This may be read as: A RELATION of a CONCEPT_1 is a CONCEPT_2. The direction of the arrows determines the direction of the reading. If the arrows were pointing
the other way, then the reading would be the same except that CONCEPT_1 and CONCEPT_2 would exchange places. As an alternative to the above "display" form, the graphs may be written in the following "linear" form, which permits input and output using a text-based computer terminal:

(CONCEPT-1] - (RELATION) - [CONCEPT-2]

The period (".") signals the end of a particular graph. Consider the following example:

[Mammal] - (part) - [Trunk].

This reads as: "A part of a mammal is a trunk." Such phrases may create readings that may sound long-winded or ungrammatical but is a useful mnemonic aid in constructing and interpreting any graph. It is easier to state: "A mammal with a trunk."

All concepts have referents, which refer to a particular individual of that concept. (Sowa discusses other kinds of referents such as measures and sets. The use of these and other items as referents has raised many complex issues resulting in controversy among the conceptual graph community. Therefore these are excluded for the sake of clarity.)

Consider the concept: [Mammal: Clyde]. This reads as: "The mammal known as Clyde." This example also happens to show that a conceptual graph can, in fact, consist of only one concept. The referent is a conformity to the type label in a concept. This example shows that Clyde conforms to the type label Mammal.

A concept that appears without an individual referent has a generic referent. Such generic concepts should be denoted as [<Type_Label>: *]. Writing [<Type_Label>] is merely a convenient shorthand. Generic concepts may take up an individual referent (Clyde would have a particular trunk). A unique number referent can suffice to make a concept distinct. Thus the generic concept [Trunk] might become [Trunk: #1234] in respect to [Mammal: Clyde]. This would yield:

[Mammal: Clyde] → (part) → [Trunk: #1234].

Larger graphs may be constructed, as in Figure 1: "A monkey eating a walnut with a spoon made out of the walnut's shell." (The translation may be stated by reference to the conceptual catalogue given in Sowa. The section on relations explains the terms agent, object, instrument, part, and material. Sowa does not distinguish along the lines of, say, nouns as type labels and verbs as relations. It might be argued that constructors of graphs would be aided if they thought in terms of these distinctions. If such guidelines do help particular graph constructors build and clarify graphs, they should be used accordingly.) This graph of the monkey eating a walnut would be impossible to
FIGURE 1.
Sowa's graph, "A monkey eating a walnut with a spoon made out of the walnut's shell."

reproduce in linear form. Therefore, additional devices are employed to overcome this:

[Spoon] -
(instrument) → [Eat] -
(object) → [Walnut] → (part) → [Shell: *x]
(agent) → [Monkey].
(material) → [Shell: *n].

The hyphen shows the relations of a concept are listed on a subsequent line. As the arrows both to and from a relation must flow in the same direction in a conceptual graph, no ambiguity is introduced. The comma terminates that part of a graph that relates to the last hyphen. Any part of the graph following the comma relates directly back to the hyphen before the last hyphen and so on, except there is no need to add a comma to pair up with the very first hyphen. Hence that part of the graph from (object) to [Monkey] relates to [Eat] while that from (instrument) to (material) → [Shell: *x] relates to [Spoon]. Indenting the previous graph clarifies the layout but has no computational effect. The *<whatever> is a coreferent marker for concepts. This shows that each such concept has the same referent. Here, if one of the [Shell: *x] concepts were to acquire an individual referent, then so would the other one. The period terminates the whole graph.

Linear graphs may be written differently but remain equivalent. For instance, this graph may have been written more clearly as:

[Eat] -
(agent) → [Monkey] -
(object) → [Walnut: *y]
(instrument) → [Spoon] -
(material) → [Shell] -
(part) → [Walnut: *y],...  

**TYPE HIERARCHY**

In conceptual graphs, type labels fall into what is known as a type hierarchy. Thereby:

Mammal (Animal.

This means Mammal is a more specialized...
type of the type Animal; that is, Mammal is a subtype of Animal. Alternatively, this can be stated as Animal is a supertype of Mammal. Similarly, the remainder of the hierarchy may be:

Animal ( Entity.
Trunk ( Entity.
Entity ( T.

The type denoted as T means the universal supertype. It has no supertypes and is therefore the most general type. Arrangement of each type into a hierarchy is a powerful feature. Knowledge is built up from degrees of commonality as each subtype inherits the sum of all its supertypes. Furthermore, a subtype can have more than one immediate supertype. For example, consider the concept ['Pet Cat']. This concept has a type label that may be defined as the subtype of both Pet and Cat. Therefore the graphs that make up Pet and Cat would both be inherited by Pet Cat.

A neat representation of the type hierarchy is a form known as the type lattice. The type lattice gives a clear multi-inheritance structure. Each pair of type labels has an immediate minimal common supertype and maximum common subtype in a perfect type lattice. Practical knowledge bases developed in conceptual graph form, however, are far more likely to fall somewhere in between a lattice and a tree hierarchy into degrees of acyclic graph structures. The three hierarchy structures are shown in Figure 2.

A referent of a type label in a concept is also a referent of that type label's supertypes. Continuing with the illustration, as Clyde conforms to Mammal, then Clyde would also conform to:

[Animal: Clyde].
[Entity: Clyde].
[I: Clyde].

These supertypes say less about Clyde than Mammal does. This relationship can be represented in type definitions. A subtype of Mammal, namely Elephant, may be introduced by the following type definition:

type Elephant(n) Is
[Mammal: 'n] - (part) - [Trunk].

"An elephant is a mammal with a trunk." This also means:

Elephant ( Mammal.

The variable x specifies that any individual referent would be coreferent for both Elephant and Mammal. The Type_Label (coreferent) part of a type definition is the specializing concept and the graph below the header is the defining graph. A specializing concept can replace the defining graph and vice versa. Because Clyde is such a mammal, it is now possible to assert:

[Elephant: Clyde] - (part) - [Trunk: #1234].

This graph cannot be contracted further because of the individual referent in the Trunk concept (unless given an individual definition). These are essentially type definitions but with an individual referent such as: individual Elephant(Clyde) is [Mammal: Clyde] - (part) - [Trunk: #1234].

In addition, the form <Type_Label> :: <individual> lists explicitly the most specialized type label conformity for an individual referent. In the example this would be Elephant :: Clyde. It is similarly possible to have definitions for relations. For example:

relation Grandparent(n, y) is
[Person: *n] - (parent) - (Person) - (parent) - (Person: *y].

"A grandparent is a parent of a parent." The type hierarchy might include:

Person ( Animal.
Parent ( Person.
Grandparent ( Parent.

PROJECTON

Projection is the pattern-matching operation for conceptual graphs. Every graph becomes more specialized when it increases the number of concepts, types, and relations within it; acquires a nongeneric referent; or substitutes subtypes for particular types. Therefore, "inside" such a graph there exists the "original" general graph. A general graph is likely to have many specialized variations. Thus the general graph projects into any of these specialized graphs. A projection common to more than one graph is said to be a common generalization.

Projection plays an important role in the inference for new knowledge with conceptual graphs. If a graph happens to project into another, then a particular pattern may have been identified. From this discovery a new graph may be asserted. Projection also

FIGURE 2.
Three hierarchies: tree, lattice, and acyclic graphs.
COMBINING GRAPHS

The joining of graphs into bigger structures provides a coherent structure to the knowledge base. Furthermore, it facilitates inference because more projections can be made into larger graphs.

Maximal join attempts to define the optimal method by which graphs are joined and occurs when graphs are joined on the largest, or maximally extended, projection that is common to them. The act of joining the graphs has the effect of specializing this projection accordingly. That projection now becomes a common specialization from which the remainder of the original graphs can then be “attached” to a new graph. This is illustrated in Figure 3, which is based on “A man loves a woman,” “Simon is located in Loughborough,” and “A woman has blonde hair.” Sowa explains experience, location, characteristic, and object.

Having stated all this, such joining of graphs may lead to invalid results. This obstacle arises because Sowa treats the generic referent as theoretically equivalent to the existential quantifier, 3, in predicate logic. (Conceptual graphs are, in fact, an existential notation. Sowa demonstrates this during a direct mapping between conceptual graphs and first order predicate logic.) 3 means a conditional “there exists an item such that...” For instance, a graph that has [Person], [Person: *x], or [Person: *x] in it means: “There exists a person (or some person) such that the other concepts and relations which make up the attached graph are valid.” However, this comparison is violated when graphs are combined, because any item will be suitable as a referent in a generic concept provided it conforms merely to the type label in the concept. This pays no regard to the conditional statement given by any concepts and relations attached to it. [Person] is treated as an unknown person.

This concept is illustrated in Figure 3, where it is quite possible that a male other than Simon loves Clare, Simon loves a blonde who is not Clare, Simon loves Clare and another blonde, Clare loves Simon and another male, Clare is not a blonde, or any suitable combination of these. Because of such lack of knowledge, joining should occur only when the referents are known to match in the graphs to be joined or when all of one graph can project directly onto another.

INFEERENCE

Inference in conceptual graphs theory is based upon the existential graphs logic of Charles Sanders Peirce (pronounced “purse”). Sowa develops “Peirce logic” to provide a comprehensive inference capability in conceptual graphs. Peirce logic, cited by its founder as the logic of the future, is seen by Sowa as an enhancement of the traditional propositional and predicate logic of Peano, Russell, and Whitehead. Consider the following example:

If Graph 1 then Graph 2.

This may be read as: “If Graph 1 can project into any graphs in the knowledge base, then Graph 2 can be asserted.” Logically, if/then can be rewritten as:

not (Graph 1 and not Graph 2).

This can be written graphically in Peirce logic as:

Knowledge-Base Graphs

Graph 1

Graph 2

This visual form illustrates the contextual domination of graphs over other graphs.
graph is dominated by another graph if the dominated graph is inside what may be considered a negated context ring, whereas the dominating graph is outside of that ring. Here, the knowledge-base graphs dominate Graph 1, which dominates Graph 2. Any graph that projects into a graph that dominates it may be “rubbed out” or deiterated. To assert Graph 2, Graph 1 must project into the knowledge-base graphs. Should this occur, Graph 1 can be deiterated leaving two rings around Graph 2. The term not(not Graph 2) equates to the term Graph 2, so the empty outside ring and the inside ring cancel out, or double negate. This frees Graph 2 out of the context and thereby means it has been asserted as a new graph. A true antecedent in an if/then rule means its consequent is always true. This is the general inference rule of modus ponens and has been demonstrated here using Peirce logic. The linear equivalent of the above is:

\[ \neg \text{Graph 1} \neg \text{Graph 2} \].

The symbol \[ \neg \] means “not.” A \[ \neg \text{...} \] forms a negated context ring. For reference purposes, contexts may be numbered by an integer series according to how deeply they are nested. Graphs that are not inside any context are deemed to be at level 0 (zero). Graphs are deemed to be at level 1 if within a context, 2 if within a nested context, 3 if within a nested nested context, 4 if within a nested nested nested context, and so on. The general form is \[ n \] if within an \[ n - 1 \] nested context. The next diagram should help to illustrate this:

```
0 1 2 3 etc.
```

\[ \neg \text{<A Graph> \neg} \] would mean that a graph is not true; that is, it is false. Therefore it is also possible by the appropriate use of contexts and nested contexts to build a knowledge base consisting of both true graphs, false graphs, and various inferences of those graphs. For the sake of clarity, we will substitute (...) in place of \[ \neg \text{...} \] to denote contexts written in the linear form. Consider the next example:

Graph 1 and Graph 2.

This is merely a case of adding Graph 1 and Graph 2 to the knowledge base because they both are true. Say, however, the example was:

\[ \text{if (Graph 1 and Graph 2) then Graph 3}. \]

In Peirce logic form this would be:

\[ (\text{Graph 1} \text{Graph 2} \text{Graph 3}). \]

Assuming that Graph 1 and Graph 2 existed in the knowledge base, then they can be deiterated and Graph 3 double negated, thereby asserting it as a new graph at level 0. Now say that the knowledge base happened to include the graph (Graph 3) instead. This states that Graph 3 is false. Regarding the previous rule, (Graph 3) can be deiterated from it, leaving:

\[ (\text{Graph 1} \text{Graph 2}). \]

This shows that because Graph 3 is false, then both Graph 1 and Graph 2 are false. It is still possible for either Graph 1 or Graph 2 to be in the knowledge base but not both. If
they were or some derivative that would state they were, this would show there is an inconsistency in that knowledge base. Return to the first if Graph 1 then Graph 2 example:

(Graph 1) then Graph 2)

Should (Graph 2) be in the knowledge base, then (Graph 1) would be asserted. This fact demonstrates another general inference rule of modus tollens, or that if the consequent of an if/then rule is false, then so is its antecedent. The illustration also shows that if the antecedent is false, then the consequent cannot be determined from it. If (Graph 1) was in the knowledge base to begin with, there is no possible way to assert either Graph 2 or (Graph 2) from (Graph 1) alone.

It should be noted that if a graph is false then so will be any of its specializations. If it is false that “Clyde is an elephant,” then “Clyde is an elephant who lives in a wildlife park” must also be false.

As a final example, consider Graph 1 or Graph 2. Logically, or can be rewritten as:

not (not (Graph 1) and not (Graph 2)).

This maps to the Peirce logic form:

((Graph 1) (Graph 2)).

By the above discussed Peirce logic operations, if either Graph 1 or Graph 2 were false, then Graph 2 or Graph 1 would be true respectively.

Figure 4 is an example using conceptual graphs. Graphs that include contexts are known as compound graphs. This example shows that before any projecting graph can be iterated, it must first be specialized and any referents passed onto all other coreferent concepts. In the display form the coreferent marker is a dotted line, called a coreferent link (Figure 4). This figure uses the same elephant example as before and reveals the relationship between type definitions and inference. For the inference to become a type definition there would need to be another compound graph with [Mammal:x] - (part) - [Trunk] at level 2 and [Elephant:*x] at level 1. Then the inference would work both ways as either the defining graph or the specializing type label could be asserted as appropriate. This applies similarly to relation definitions. The additional compound graph required to turn the example inference into a type definition is illustrated by this linear representation:

([Elephant:*x];
[[Mammal:*n] - (part) - [Trunk]]).

The semicolon indicates the end of a particular graph other than the final graph. The period is used only to terminate the whole compound graph.

In summary, Peirce logic shows contexts of knowledge elements visually dominating others. Inference is performed by attempting to reduce those contexts.

**ALGORITHMS**

The algorithms presented in Listings 1 and 2 indicate the nature of program code needed to implement conceptual graphs. This gross simplification is necessary because the
program from which they are taken is far too large to be published here. The operations of projection and projective extent, mentioned in the definitions below, are defined in Sowa.

Definitions

- $Q$ is a query graph, which is any graph that must be proved
- $K$ is a knowledge base
- $\pi(Q,G,\sigma)$ is the projection of a graph $Q$ onto a graph $G$, where $\sigma$ is a list of substitutions made when variables in $Q$ were instantiated
- $\Pi(Q)$ returns the projective extent $E$ of a graph $Q$ onto the knowledge base $K$. $E$ is a set of pairs of the form $(\pi(Q, \text{some graph } g \text{ in } K, \sigma),\sigma)$ where $\sigma$ is a list of substitutions made when variables in $Q$ were instantiated
- $\delta(Q)$ is the denotation operator and returns the pair $(T,E)$, where $T$ is the truth value of $Q$ in $K$. $T$ can take values true, false, unknown and inconsistent
- The symbols $\land$, $\lor$, $\neg$, $\forall$, and $\exists$ have their usual meaning.

Algorithm one (Listing 1) assumes a backtracking mechanism that will successively try different graphs and substitution lists at line 30. Also, the reproof of the graph at line 35 will need to backtrack, as will the recursive evaluation of $q$ at line 26. A loop detection mechanism must be added to a real system. This consists of maintaining a list of graphs that are being proved and that is passed to each incarnation of the algorithm at line 26. Line 20 then tests that the graph $Q$ is not present in this list. If it is, then $Q$ has been proved and its truth value is returned as unknown at this point. Line 24 of algorithm 1 finds a rule that is somehow appropriate to $Q$. Algorithm 2 (Listing 2) clarifies this.

The version of algorithm 2 is adequate for the examples here, but the real system from which this algorithm was taken uses a much more complex version. Once again, a backtracking mechanism is assumed so that different rules will be tried at line 1 and different valid projections of $g$ onto $Q$ will be tried at line 3.

PLACE TO START

The rationale behind this guide has been to present both factual and reasoning knowledge in conceptual graphs in a manner that lends itself to being clearly usable. It is hoped this approach will help to spur the deployment of conceptual graphs as a knowledge-based support tool for tackling practical qualitative problem situations.

SUGGESTED READING


C/04: "Bridging Accounting and Business Strategic Planning Using Conceptual Graphs"

Proceedings of the Seventh Annual Workshop on Conceptual Graphs, 8-10 July, 1992, New Mexico State University, Las Cruces, USA, pp. 203-212.
Bridging Accounting and Business Strategic Planning Using Conceptual Graphs

Simon Polovina

Department of Computer Studies, Loughborough University of Technology (LUT),
United Kingdom LE11 3TU

Abstract. This paper reveals the existence of a gateway between the two seemingly disparate subjects of accounting and strategic planning. This is achieved by examining two important existing methodologies, events accounting and cognitive mapping, and restating them both in conceptual graphs. Also, for the bridge to be viable the framework must be usable by accountants and strategic planners. To achieve this usability a conceptual graph-based conceptual analysis and review environment (CARE) is proposed. From all this, a way forward exists whereby a business can develop and maintain a corporate knowledge-base that permeates throughout more of its diverse activities.

1. Introduction

To enable an organisation to have more control over its affairs, it would be very desirable to bring together into one positive framework two seemingly disparate yet important parts of business activity. The first area, accounting, essentially provides an established basis for controlling numerically-based problems\(^1\) whereas the second, strategic planning, is characterised by fundamental but highly subjective problems that do not lend themselves to accountancy's quantitative techniques. The paper reveals the existence of a gateway between the two by examining an influential methodology in each area, and then restating them both in conceptual graphs [15]. The first methodology, events accounting, is a significant attempt to capture the conceptual basis underlying accountancy [5,9]. The second, based on cognitive mapping, is a leading knowledge-based strategic planning analytical tool [3].

2. Events Accounting

The major benefit of adopting a structured model of a problem is so that such a model, by its inherent nature, draws out all the problem’s relevant parameters from which a solution can be investigated fully. Contrast this with a written or spoken text discussion of the problem where it is well known that ambiguity and obfuscations can occur easily. This 'natural language' interpretation of problems may be the most flexible and easily followed, but without at least a basis in some structured form it can be dangerously erroneous. Hence the emergence of disciplines such as accounting that attempt to model the dynamics of economic activity in a structured way. The model,

\(^1\) Although the author has previously discussed the modelling of non-numeric elements within accountancy [10,11].
of course, must also be structured on a suitably principled basis. Otherwise it will omit or misinterpret the salient issues of the problem situation.

2.1 The Double Entry Bookkeeping Model

It was with the above in mind that the traditional model of accountancy, the bookkeeping model, was developed in the Middle Ages (7). The principle behind this model is economic scarcity. In other words for every benefit a sacrifice has to be made. For example, the benefit of a business owning its office is sacrificing $1,000,000 that could be employed elsewhere; a book prepared by its author researching a new exciting area in semantic understanding may have involved that author deciding against many complex yet important alternatives, such as the costs of, say, not participating in his or her growing family. These 'transactions' occur because the decision-maker makes a value judgement that the benefits outweigh the costs.

The bookkeeping model appears simple but rigorous. Fundamentally, instead of recording one amount per transaction it records two: A 'debit' and a 'credit'. Moreover these amounts are complementary to one another, hence they 'balance' against each other. An accounting 'balance sheet' is merely the aggregate of all these debits and credits. The rigourousness derives from this principled 'double entry' structure so that each benefit is accounted for by a cost and vice versa. Hence every gain is matched to a sacrifice.

2.2 Problems with the Bookkeeping Model

However on deeper investigation the double entry bookkeeping model is unlikely to capture all these economic value trade-offs. Say the business in the first example above decides to sell its office. This transaction can be recorded easily by the elementary bookkeeping entries "DEBIT Cash $1,000,000, CREDIT Fixed Assets $1,000,000". The second, preparing the book, is simply too qualitative to be recorded by the bookkeeping model yet the author may want to know clearly about all the actual costs and benefits of such a transaction. This neglect on the part of the bookkeeping model is elaborated on below.

Errors of Omission. The threshold where the bookkeeping model may break down is perhaps lower than may be thought. Reconsidering the first example about the office, the value of selling the current office may be the purchase of cheaper offices for $500,000. In this case the bookkeeping is basically "DEBIT Fixed Assets $500,000, CREDIT Cash $500,000". Now say, by spending the remaining $500,000 elsewhere, the business generates a revenue of $600,000. On aggregate in the balance sheet the business's money worth then increases by $100,000 (represented primarily as "DEBIT Profit and Loss Account $100,000, CREDIT Reserves $100,000"). However if the value of the current office is retaining key employees through a comfortable work environment then, as in the author example above, the bookkeeping model is inappropriate. Therefore the double entry bookkeeping model is easily liable to make significant errors of omission.

Errors of Commission. Furthermore the bookkeeping model could mislead. Reconsidering the 'preparing the book' example the value may be viewed as the more
easily quantified cost of the author ceasing to conduct consultancy work at $2,500 a
time instead. This revenue would have been recorded by the bookkeeping model on an
ongoing basis. However the book might bring its author satisfaction of a deep desire
for an enhanced reputation amongst peers. Unless this can be translated into a cash
benefit the bookkeeping model would not record these judgements and thereby leave a
'loss' of $2,500. By choosing to author the book the decision-maker qualitatively has
to justify, against the grain of the bookkeeping model's assessment of value, why that
$2,500 has been forsaken even though this may the lesser value item. Therefore the
double entry bookkeeping model, taken too literally, can also readily lead to
significant errors of commission.

2.3 The Nature of the Events Accounting Model

The above problems are familiar to most accountants. The Economist, for instance,
summarises the difficulties accountants have in attaching a monetary value to
'intangible' assets such as product brand names [4]. The events accounting model [5,9]
is intended to overcome the above obstacles. Unlike the bookkeeping paradigm, events
accounting attempts to capture the qualitative dimensions of economic scarcity. The
model is shown by the diagrams that make up Fig. 1. Fig. 1(a) shows the events
accounting model as an entity-relationship diagram [1]. In line with the earlier
discussion about the desirability of structured models, Chen argues that the pictorial
nature of entity-relationship diagrams are particularly useful in structuring problems
qualitatively stated in natural language [2]. Given this actuality, events accounting
represents a powerful means of recording scarcity as more than a monetary measure.
Setting aside its 'dotted' part for the moment, Fig. 1(a) reveals the fundamental links
between an 'economic resource', which means some exchangeable item of value, and
the parties which create the 'economic event' that causes the economic resource to be
exchanged.

Fig. 1(a). The events accounting model in entity-relationship form [7].
2.4 The Events Accounting Model as Conceptual Graphs

Why. Sowa, in response to Chen’s argument, adds conceptual graphs can further structure the dimensions of natural language-based problems [16]. Conceptual graphs extend entity-relationship diagrams by adding the capacity of first-order logic in a pictorial way [17]. It is therefore sensible to transform the events accounting model into conceptual graphs form.

![Fig. 1(b). Transformation of the events accounting model into supertype conceptual graphs.](image1)

![Fig. 1(c). Transformation of the events accounting model into subtype conceptual graphs.](image2)

Restating the Events Model. The conceptual graph in Fig. 1(b) represents this transformation. This graph is basically a conceptual graphs reproduction of the general supertype parts of Fig. 1(a), except that the arguably more definitive term 'event subject' is substituted for 'sale line item'. Fig. 1(c) basically expresses the specialised subtypes of the top diagram. It also refines 'party to' into 'source' and 'destination', and thereby shows the route by which the economic resource changes possession. For both the conceptual graph diagrams, certain aspects in the entity-relationship model are not reproduced to focus on the salient nub of events accounting.

![Fig. 2. Completing the events accounting model by including duality.](image3)

Duality. However the conceptual graphs within Fig. 1 do not yet fully capture events accounting. To achieve this the duality principle of economic scarcity needs to be considered. Duality is defined by the statement "for every benefit there has to be a
sacrifice made". The events accounting model indicates duality in the 'dotted part' of Fig. 1(a). This 'cash receipt pays for the sale' is really a shorthand to make that top diagram concise. For instance 'party to' should also connect to 'cash receipt' because it is also part of the exchange. Fig. 2 reveals this duality in full at the supertype level. Both sides of duality are shown in Fig. 2 by linking the economic events to the same transaction. As already indicated, the 'Economic Resource' concepts may be specialised to any quantitative or qualitative concept describing an item of value. As one value describes the benefit in the transaction, the other value depicts what had to be sacrificed for that benefit. An example scenario is portrayed in Fig. 3.

![Diagram of economic events](image)

**Fig. 3. Apparent duality of selling offices.**

Intangible Values. Fig. 3 develops the earlier example of a business wishing to realise cash by selling its expensive offices, with a view perhaps to purchasing cheaper ones. The diagram shows a $1,000,000' cash benefit to 'OurCo Inc.' for the loss of their high-class offices at 1, Prestige Plaza'. As such, it may easily be recorded by the bookkeeping model. However this may not be the full picture. In addition to the loss of the present offices, there may be an intangible loss due to disgruntled key employees leaving the corporation as a consequence. As discussed earlier, such intangible values remain outside the bookkeeping model. Nevertheless the business would require its inclusion into the sphere of consideration should this factor, say, determine the survival of the business itself. The conceptual graph of Fig. 4, manages to record this qualitative cost.

![Diagram of intangible loss](image)

**Fig. 4. Further cost-side duality of selling offices.**

2.5 Summary: Present Accounting in a Knowledge-Based Form

The events accounting model, in summary, does not restrict the recording of transactions to only those that can be captured by numeric monetary measures. Thereby its wider scope enables qualitative value judgements to be modelled and hence be gauged along with the quantitative elements. In databases, such parameters are defined in data fields which can be then subjected to some selection criteria. Within any knowledge-base, in addition to this 'lookup' facility, there should exist the ability to make adequate inferences for new knowledge. Hence if accounting transactions can be presented in a knowledge-based form they can be subjected to this inferencing.
Peirce logic, which is the inference mechanism associated with conceptual graphs, will be seen to be a powerful part of the pivotal area that performs this role. This area, strategic planning, is addressed in the following discussion on cognitive mapping.

3. Cognitive Mapping

Cognitive mapping is a practical technique used by strategic planners to structure the highly subjective problems that characterise strategic planning [3]. This technique stems from Kelly's theory of personal constructs [6].

3.1 The Nature of Cognitive Mapping

The small cognitive map shown in Fig. 5 continues the office relocation example discussed so far, and demonstrates the two essential elements of cognitive maps. These elements are referred to as 'concepts' and 'links' in cognitive mapping.

Concepts in Cognitive Mapping. Each cognitive mapping concept is represented as one 'emergent pole', which describes one side of the problem, and a 'contrasting pole' which is meant to focus the concept by a meaningful contrast to the emergent pole. In Fig. 5 the left-hand concept contains the emergent pole 'buy down-market offices at 13, Sidestep Row'. The contrasting pole, separated from the emergent pole by '...', is 'retain high-class offices at 1, Prestige Plaza'. Where a contrasting pole is unspecified, as in the case of 'disgruntled key employees', then the contrasting pole is determined to be 'not <emergent pole>'. Hence the contrasting pole for the latter concept becomes 'not disgruntled key employees'.

Links in Cognitive Mapping. Attached to the cognitive mapping concepts are links which are used to demonstrate how poles lead to other poles. The link is the arrow which leads from the 'buy cheaper offices at 13, Sidestep Row... retain current offices at 1, Prestige Plaza' concept to the concept with the emergent pole 'disgruntled key employees' and default contrasting pole 'not disgruntled key employees'.

3.2 Cognitive Mapping as Conceptual Graphs

From the above the down-market offices lead to disgruntled key employees whereas the current offices lead to these employees not being disgruntled. After refining this statement from the elementary form illustrated by Fig. 5 into the more logically advanced conceptual graphs, the choice of offices and its consequence may be remodelled as given by the conceptual graphs of Fig. 6.

Restating the Cognitive Map Concepts. In Fig. 6(a) the pair of specified poles denoting the choice of offices can be restated as a conceptual graph by placing

---

379
each pole into a separate conceptual graph concept and together surrounding them within a Peirce negative context. The single specified disgruntled employee pole and its unspecified 'not' contrast is also restated in Fig. 6(a) although, as can be seen, this graph turns out to be a tautology.

Fig. 6(a). Conceptual graphs for the concepts of the example cognitive map.

Fig. 6(b). Conceptual graphs for the links of the example cognitive map.

Restating the Cognitive Map Links. Fig. 6(b) demonstrates, in conceptual graphs, the cognitive map links as two implications based on the Pierce logic operations of deiteration and double negation. The conceptual graphs show both implications to be generalised, assuming this step is valid, to show that for any down-market offices the key employees located there are disgruntled whilst in any high-class offices they are not (modus ponens). Similarly if such employees are not disgruntled then they are not located in down-market offices (modus tollens).

Why. Conceptual graphs provide the contrast in cognitive map concepts by stating that one pole must be false if the other is true. However when both poles are specified the converse, that if one is false the other is true, cannot be determined. This distinction, which is not clear from the elementary cognitive mapping model, is intended. The business (referred to earlier as 'OurCo Inc.') may, in the event, dispose of 1, Prestige Plaza and obtain offices other than at 13, Sidestep Row or even not any other down-market offices for that matter. Hence it may be quite wrong to assert that they will be at either 13, Sidestep Row or 1, Prestige Plaza. Furthermore conceptual graphs allow each pole to be generalised or specialised to differing degrees so enabling a potential continuum of contrast. This dimension happens to accord more precisely with Kelly's personal construct theory mentioned earlier [6]. There may be, for example, a more general contrasting pole to a more specific emergent one. Ultimately the most general contrasting pole is the 'not <emergent pole>'. As for the cognitive mapping links, the modus tollens inference is arguably not evident in the elementary form given by Fig. 5.

3.2 Summary: Conceptual Graphs Enrich Cognitive Maps

As well as the ability to a) generalise and specialise problem scenarios to the

380
appropriate degree and b) vividly express modus tollens, conceptual graphs further reveal that c) there is no real distinction between the concepts and links of cognitive mapping anyway. As should be evident from a careful examination of Fig. 6, these links turn out to be like emergent and contrasting poles. Hence conceptual graphs remove the arbitrary choice as to what a link or a pole. Moreover strategic planning problem situations can now be captured alongside the accountancy domain through the single medium of conceptual graphs. This opens the way towards suggesting a bridge between the two techniques.

4. Suggesting a Bridge

An accounting methodology that incorporates the events model, rather than merely the bookkeeping model, enables the more qualitative elements of transactions based on economic scarcity to be captured along with the more quantitative elements. Conceptual graphs enhance the entity-relationship model and include a visual method of inference in the form of Peirce logic. It thereby widens an organisation's numeric-based financial record systems into a corporate knowledge-base that can now include the parameters of strategic planning.

4.1 The Technical Bridge

However, given the discussion so far in this paper, it is not expected that inference will play a major role for the qualitative elements in accounting transactions. Such inference becomes significant instead within strategic decision-making where, as also discussed, it in fact plays the major part. Yet presenting accounting events as conceptual graphs because of the inference capability remains worthwhile. This is because their qualitative aspects can then be interrogated and reviewed by the conceptual graphs derived from strategic analysis. For example, should the graph \[ \neg(\text{Disgruntled Key Employee}) \] be asserted as a parameter of a strategic decision then, given the graphs so far for the office location example (and OurCo Inc. Disgruntled Key Employee < Disgruntled Key Employee), the transaction disposing the offices at 1, Prestige Plaza would be blocked. Such operations describe the technical half of the bridge.

4.2 The Usability Bridge

As the nature of much business activity is not only qualitative but also constantly changing, the corporate knowledge-base is unlikely to fully keep up with the tacit and implicit parts of business knowledge [8]. Therefore, to facilitate sufficiently rapid analysis and review, the corporate knowledge-base must be usable by accountants and strategic planners themselves to complete the remaining half of the bridge.

CARE. To achieve the usability bridge the author proposes a human-computer interface which may most aptly be referred to as a conceptual analysis and review environment, or CARE, through which the accountant or strategic planner can interact with the corporate knowledge-base. The author suggests CARE could be conceptual graph-based. This is because a) this paper identified that accountants and strategic planners evidently appreciate the need for structuring problems, b) business information professionals such as accountants use structured diagrams in the general
course of their work [14,18] and c) negative contexts happen to be similar to the way the accountant's bookkeeping model negates numeric values by enclosing them within rounded brackets [7].

Safeguards. There must be, nevertheless, certain safeguards. Terms like $\forall$, $\exists$ and $\neg$, for instance, are unlikely to be understood by these non-logicians and should therefore be avoided. A study conducted by the author discovered that even the linear form of conceptual graphs is too abstract for users [13]. Hence CARE would employ conceptual graphs in display form. In addition, points of the theory itself must be as clear as possible. The author has jointly attempted such a clarification [12]. Ideally, intricate parts of theory should be handled by the machine and be transparent to the domain expert without making the power of CARE too trivial. Allied to this, CARE should prevent the domain expert from drawing incorrectly structured graphs.

Fundamental Problems. Finally there will be, of course, fundamentally unavoidable problems through allowing the user to build more advanced models. For instance, 'A' affects 'B', 'B' affects 'C', 'C' affects 'D', and 'D' affects 'A'. In some cases this may be a valid recursion but in others it may not. This dilemma is especially significant in conceptual graphs given that knowledge can be generalised and specialised at many levels. Such obstacles should be mitigated as far as is possible given that it going to be very difficult to determine valid from invalid cases. For this example some kind of user-warning facility may help. Accountants, through their experience in the use of spreadsheets for instance, tend to be acquainted with the nature of circular arguments albeit at a much more straightforward level.

5. Concluding Remarks

From all the above, conceptual graphs have revealed the fundamental difference between accounting and strategic planning is the former is essentially descriptive in nature and can thereby be captured by knowledge representation alone whereas the latter is prescriptive and requires the dynamics of inference. However the benefit is not one way: Accounting can offer strategic planning a firmer conceptual basis from which its dynamic models can be built. With CARE, conceptual graphs are potentially usable by domain experts themselves. These domain experts would be accountants and strategic planners. Hence a way forward exists whereby a business can develop and maintain a corporate knowledge-base that permeates throughout more of its diverse activities.

Acknowledgements

The author would like to thank Paul Finlay, Loughborough Business School, LUT; John Heaton, Computer Studies, LUT; Chris Hinde, Computer Studies, LUT; and Malcolm King, Loughborough Business School, LUT for their invaluable help towards the content of this paper.

References


383
C/05: "The Economics Of Supply And Demand: An Important Challenge For Conceptual Graphs"

The Economics Of Supply And Demand: 
An Important Challenge For Conceptual Graphs

Simon Polovina 
Department of Computer Studies 
University of Technology 
Loughborough, UK LE11 3TU 
S.M.Polovina@lut.ac.uk 

Harry S. Delugach 
Computer Science Department 
University Of Alabama in Huntsville 
Huntsville, AL 35899 
delugach@cs.uah.edu 

Abstract. Conceptual graphs have been used to model information in many complex domains, but the domain of economics is particularly difficult because its knowledge is based as much on perceptions of people as on physical laws. This paper addresses that problem using as a vehicle one well-known basic economic area: namely, the law of supply and demand. Employing Peirce logic negative contexts, we represent various parts of classical economic theory, e.g., over-supply, over-demand, and equilibrium states. It is shown how tacit knowledge is relevant to the modeling of this information, and why this knowledge requires the conceptual graphs to be built and reviewed by the domain experts directly. Conceptual graph actors are employed to represent relationships between quantities and to represent market forces. Negative contexts are briefly evaluated as a modeling tool. Representing changes in domain assumptions is discussed.

1 Introduction

Conceptual graphs are being used to model information in many complex domains, such as interpreting radiological evaluations, context-searching in documents, playing chess, etc. One of us (Polovina) is using conceptual graphs to model accounting and economic knowledge [1]. Economics is known to be far more extensively based upon the highly elusive dispositions of people rather than definitive physical laws [2,3]. Rhodes illustrates how economics' precarious scientific basis has caused the economist to be the subject of much ridicule [4]. Nonetheless Rhodes also shows that the economist's knowledge remains consequential, because economics is a reality that affects us all [4]. The teaching style of Pool and La Roe's amusing text on understanding economics underscores this actuality, by drawing considerably on the formative, yet imprecise human instincts that their novice readers would already possess [5]. Hence the indomitable conceptual bases that characterize economics provide an excellent proving ground for conceptual graphs [6].

This paper focuses on one economic concept, namely that knowledge comprising what is popularly known as "the law of supply and demand." This aspect of fundamental microeconomic knowledge is a key element in modeling essentially any overall economic theory. As indicated above, it is also a good example of an economics that is well-known by the general public. It therefore represents not only a class of domain-specific knowledge, but also a corpus of "commonsense" general knowledge that should well suit conceptual graphs [6].
With the above in mind, we (a) explore the nature of supply and demand, (b) provide some conceptual graphs models of this area, and (c) identify, from (a) and (b), the kinds of initial but noteworthy challenges for the conceptual graphs technique itself. We are also paying particular attention to the human-understandability of the graphs in capturing economics' conceptual bases, as illustrated below.

The paper is organized as follows. Section 2 discusses the law of supply and demand, with conceptual graphs representations along the way. Section 3 discusses some issues we have identified during the course of this investigation, such as the use of negative contexts and changing domain assumptions. Section 4 concludes the paper.

2 The Law Of Supply And Demand

2.1 Background

Put simply, the law of supply and demand holds that the price of some resource depends upon relationships between the supply of the resource and the demand for the resource. A resource might be some person's time, an unmarried person available for dating, or anything valued by another person. In a supply and demand context, that resource becomes an economic resource. Ijiri clarifies economic resources as objects that (a) are scarce and have utility, and (b) are under the control of an enterprise [7, pp.51-52].

A recent compilation of American cultural information describes supply as "the amount of any given commodity available for sale at a given time," and demand as "the amount of any given commodity that people are ready and able to buy at a given time at a given price." The law of supply and demand is described as follows:

"In classical economic theory, the relation between these two factors determines the price of a commodity. This relationship is thought to be the driving force in a free market. As demand for an item increases, prices rise. When manufacturers respond to the price increase by producing a larger supply of that item, this increases competition and drives the price down. Modern economic theory proposes that many other factors affect price, including government regulations, monopolies, and modern techniques of marketing and advertising." [8, p.434]

This description is our starting point — firstly because it concisely characterizes the important parts of supply and demand, and secondly, because it purports to be a view of popular culture based on "common sense."

2.2 Initial Conceptual Graphs Model

We will attempt in stages to describe supply and demand using conceptual graphs. The graph in Figure 1(a) shows, employing Peirce logic negative contexts, that if there is an economic resource then it has both a supply and a demand. Figure 1(b) shows that if there is an economic resource whose demand quantity \( x \) is less than its supply quantity \( y \), then that economic resource is in a state of over-supply. Similarly, in Figure 1(c), if there is an economic resource whose demand quantity \( x \) is more than its supply quantity, then it is in a state of over-demand.
2.3 The Appropriateness of Conceptual Graphs

A psychological study by Novak and Gowin demonstrated how the structured diagrammatic nature of their simple "concept maps" elicited key concepts and relationships even to young children [9]. Sowa explains however that concept maps, unlike the equally diagrammatic conceptual graphs, cannot express all of logic and natural language for serious analysis [10]. The tough problem domain of economics would certainly require such levels of analysis, but the particular significance of the above psychological basis is that our representations extend beyond strictly logical issues. This is because the model must also enable the economist, accountant or strategic planner to build and review their own knowledge bases in view of the high level of continually shifting tacit knowledge that economics embodies. Hence the above argued readability of conceptual graphs is as important as the graphs' technical power.

The above should become more evident as this discussion progresses but, to give an initial illustration, we debated the choice whether to employ "quantity" or "units" in our graphs. We discovered that a case could be made for either, merely according to individual inclination. As a result we decided that a deep semantic analysis of "quantity" or "units" outright mattered substantially less than what those terms symbolize psychologically to each user through his or her own conceptual graphs models. Of course as corporate knowledge bases evolve there could be some standardization effort amongst the users to gain the benefits of wider scale knowledge. This would, as a consequence, involve standardizing terms as well as graphs. Nevertheless, all would still be as agreed by a consensus of the users, not as dictated by some well meaning knowledge engineer.

2.4 Further Aspects

Figure 1 shows other examples of how conventional terms are used. For example, the terms under-demand and over-supply would have essentially the same meanings. Which one should be chosen? We believe that no a priori decision is possible. An empirical study might show a preference among Western economists for one over the other. Even if such a preference was found, the claim could not be made that one is
somehow "better" or "more natural" than the other; future empirical studies, or studies of economists in other cultures might show a different preference.

The spatial arrangement of these graphs is also significant. The "(supply)" relation and the "(demand)" relation are both in the same spatial location in both Figure 1(b) and Figure 1(c). The authors wonder if there is any escaping a significant degree of similarity between the two. For instance, the term "less than" could have been used instead of "more than" in the right hand graph but there would still remain that visual similarity. Are the conceptual structures thereby telling us something here?

2.5 Capturing Natural Meaning

The (less than) and (more than) relations are of course not merely arbitrary ones. In fact, they are related through our tacit common-sense knowledge. Figure 2(a) captures the natural meaning of "If measure x is less than measure y then y is more than x." Likewise, Figure 2(b) captures "If y is more than x then x is less than y."

Figure 2. Representing Relationships Between Relationships

Figure 3 shows relationships between the notions of over-demand and under-supply, so that the economist may use any terms that are convenient. Figure 3(a) represents "If over-demanded economic resource a then not over-supplied economic resource a and vice versa". Figure 3(b) represents "If under-demanded economic resource a, then not under-supplied economic resource a and vice versa." Figure 3(c) represents "If over-demanded economic resource a, then under-supplied economic resource a." Figure 3(d) represents "If under-demanded economic resource a then over-supplied economic resource a."
Appendix C105

The above would chain together backwards, forwards, or appropriate combinations thereof. The above simple examples vividly show that Peirce logic makes all these relationships much more evident to the domain expert than if-then rules. It may also suggest alternative viewpoints to the economist, once he or she is shown some alternate representations that are relevant.

2.6 Incorporating Actors

To continue, economists are interested in more than just whether or not there is a state of over-supply or over-demand, but also the actual measure of difference and the consequences of being in either of those states. Figure 4 uses actors to explicitly show the functional relationship between the demand and supply quantities.

Figure 4(a) characterizes the situation of an over-supplied resource. The demand quantity $x$ is less than the supply quantity $y$, as denoted by the relation (less than). The actual amount of over-supply is obtained by the functional relation $<$subtract$>$ between $x$ and $y$ which determines $z$ to be the over-supply quantity of resource $a$. Similarly, Figure 4(b) characterizes the situation of an over-demanded resource. Figure 4(c) characterizes the equilibrium situation. Since in this situation, there is no difference between the demand and supply quantities, they can be represented by the same quantity $x$, which is also the equilibrium quantity.

Figure 4 illustrates one powerful feature of conceptual graphs. The same graph can represent qualitative relationships (e.g., less-than) as well as quantitative relationships (e.g., the difference between two quantities). For some purposes the qualitative relationship will suffice; for other purposes, the measure of over-supply (or over-demand) may be desired.
Figure 4. Representing Quantities Of Demand And Supply

Figure 5 reveals that if an economic resource is in over supply then its price decreases or less units of it are produced. Note that the term "price" is used instead of "money." This is because price may be derived from a non-monetary exchange, and that money itself is anyway fluid through exchange rate movements and inflation. Of course even this seemingly clear-cut point should not be treated as a completely hard and fast rule, for the reasons of user tacit awareness already outlined. Figure 5 shows how the state of having an over-supplied economic resource causes the price and quantity to decrease toward their equilibrium amount. Figure 6 shows a similar result for the over-demanded economic resource. Both Figures 5 and 6 show the use of actors to represent the effect of market pressure.
Figure 5. Representing Over-supply

Figure 6 describes the over-demand case. As with over-supply there is the effect on price or units produced, albeit now an increase not a decrease.

Figure 6. Representing Over-demand

The above graphs as a basis from which more refined graphs would need to be constructed perhaps according to particular situations. Suggestions as to ways the graphs might need to be modified are discussed later on, under the next section of "Further Issues."
3 Further Issues

In the course of developing these ideas, we have encountered several further issues that have implications to the larger conceptual graphs community. This section summarizes some of those issues we have recognized.

3.1 Negative Contexts

Inferencing in conceptual graphs employs negative contexts, adapted by Sowa from the logic of Charles Sanders Peirce [6]. To illustrate, the rule “P \Rightarrow Q” could also be expressed as “\neg (P \land \neg Q)”. This transformation may appear to be initially counter-intuitive to non-logicians, but Peirce relied on converting other logical relationships into AND and NOT form to obtain the visuality of his logic. To bridge the gap, Sowa consequently proposed explicit relations in conceptual graphs such as one for “implication” [6]. Therefore in the linear notation:

\[
[P] \rightarrow (implication) \rightarrow [Q]
\]

essentially means:

\[
(not) \rightarrow [P] \rightarrow (not) \rightarrow [Q]
\]

Farques et al. go further and dismiss Peirce logic completely, replacing this aspect of conceptual graphs with “if-then” rules on the grounds that Peirce logic is unnecessarily complicated [11]. However one of us (Polovina) is currently exploring the value of retaining Peirce’s negative contexts. For instance, Peirce logic explicates even the simple modus tollens relationships obscured somewhat by the procedural nature of an “if-then” implication. Hence, say, what if it is false that quantity “x” is less than “y”, then quantity “y” is more than quantity “x” would be asserted as false. Careful examination of the graphs presented in this paper will reveal such and more involved “what if” interrelationships not evident from if-then rules alone. Furthermore the method of negation in Peirce is arguably similar to that in the accountant’s bookkeeping model, where figures are negated by surrounding them in brackets. For example the complementary double entry of “$3,000” is “($3,000)”. This and other similarities have been discussed in previous work [1].

3.2 Changes in Domain Assumptions

Not surprisingly, there is much more to supply and demand problems than we have discussed. For example, for the over-demand case already denoted by Figure 6, a slightly more complex situation is described in Figure 7. From Figure 6, it can be seen that over-demand leads to either an increase in price or units produced. However it may be that these two parameters are fixed, say by government policy. Therefore a third possibility can be introduced that states a certain amount of demand for that economic resource could remain unsatisfied instead.
Moreover there are instances where supply and demand do not follow the expected pattern, such as the decreasing of the price of an economic resource may turn out to decrease its demand. This may be because dropping the price of a product may cause customers to believe it is now second-rate and thereby avoid buying it. Hosking reports on the concern of the luxury fragrance houses that large discount health and beauty retail chains wish to sell those houses' perfumes at a substantial discount of up to 50%. Hosking notes that the advertising media take the houses' attitude so seriously that they will not accept advertisements by the discount retailers, for fear of losing the lavish advertising business of the houses [12], Becker [13], and The Economist magazine [14, p. 85], describe in a similar vein the highly-fashionable, or "in" commodities which, as their price increases, their demand increases. However this only occurs if demand is never satisfied, otherwise the price and quantity demanded drops again.

The above dynamic well exemplifies how the pervasive yet elusive tacit/implicit knowledge in a problem domain (here in terms of what is presently in fashion) affects the fundamental methodology that attempts to model that domain, and underpin the reasons why is it important that the models are built by the domain experts directly. Figure 8 is presented lastly as one thought-provoking way towards how such scenarios might eventually be captured by an economist using conceptual graphs.
4 Conclusion

This paper has shown how some basic common knowledge, namely, the law of supply and demand, may be represented naturally through conceptual graphs. We relied heavily on negative contexts to convey relationships between facts, and upon actors to convey functional relationships. We took a highly user-centered approach with economists, accountants or strategic planners building and reviewing conceptual graphs that are familiar to them. We found that even some rather esoteric marketing notions may thus be effectively modeled. Nonetheless we believe that further empirical studies are needed on the understandability of these representations by practicing domain experts. Such studies will provide further evidence for or against the claims made here.

We made much use of conceptual graph actors. Since economics deals with processes and many cause-and-effect relationships, we believe that actors are an appropriate way to represent this information. In particular, we note the usefulness of actors in representing both quantitative relationships and even the somewhat elusive notion of market pressure. Further work is needed to be able to specify precisely the behavior of actors in our graphs.

Although we would accept that this paper has only touched upon the economics of supply and demand, let alone economics in general, the discussion reveals that this problem domain gives conceptual graphs not merely a rich but also a meaningfully real avenue to vent the technique's worth. The graphs presented within this paper illustrate the possible directions in which conceptual graphs-based modeling could proceed. An economics knowledge-base that usefully captured the endlessly baffling dimensions of economic activity would be quite useful, and clearly bring the formalism employed to the forefront of eminence. The chance offers conceptual graphs to take that much-awaited lead.
Acknowledgements

One of us (Polovina) would like to thank Chris Hinde, Computer Studies, LUT for his help towards the content of this paper.

References


