A study of the design expertise for plants handling hazardous materials

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/2134/7503](https://dspace.lboro.ac.uk/2134/7503)

Publisher: © A.R. Bunn

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/
A Study of the Design Expertise for Plants Handling Hazardous Materials

by

Adrian R. Bunn

A Doctoral Thesis submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of the Loughborough University of Technology

October 1987

© A.R. Bunn 1987
BEST COPY

AVAILABLE

Variable print quality
This thesis is dedicated to my parents
for providing me with the opportunities
and encouragement for this work

"An expert is a man who has made all the mistakes
which can be made in a very narrow field"

Niels Bohr

1885 - 1952
ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor F. P. Lees for valuable guidance and discussion throughout the course of this work.

I would also like to thank Dr. P. K. Andow and Professor T. A. Kletz for their help, together with colleagues and friends in the Department of Chemical Engineering and the Computer Centre.

I also acknowledge the financial assistance given to me from the Science and Engineering Research Council.

Finally, I would like to thank Helen for her help, support and encouragement over the last 2 years.
# Contents

## 1 Introduction

1.1 Plant Design
   1.1.1 Design of Plants Handling Hazardous Material

1.2 Safety and Loss Prevention
   1.2.1 Loss Prevention and Plant Design

1.3 Computer Problem Solving

1.4 Expert Knowledge
   1.4.1 Expertise
   1.4.2 Knowledge

1.5 Expert Systems

1.6 Project Objectives
   1.6.1 Problem Identification

## 2 Expert System Fundamentals

2.1 Applications of Expert Systems
   2.1.1 DENDRAL
   2.1.2 MYCIN
   2.1.3 PROSPECTOR
   2.1.4 R1
   2.1.5 Applications in Chemical Engineering

2.2 Architecture of Expert Systems
   2.2.1 The Knowledge Base
   2.2.2 User Interface

2.3 Facilities
   2.3.1 Control
   2.3.2 Questions
   2.3.3 Explanations

2.4 Validation
   2.4.1 Consistency
   2.4.2 Completeness
   2.4.3 Soundness
   2.4.4 Precision
2.4.5 Usability
2.5 Building Expert Systems

3 Knowledge Representation and Acquisition

3.1 Knowledge Representation
  3.1.1 Semantic Nets
  3.1.2 Logical Expressions
  3.1.3 Object-Attribute-Value-Triplets
  3.1.4 Production Systems
  3.1.5 Frame Systems

3.2 Knowledge Acquisition
  3.2.1 Elicitation
  3.2.2 Sources of Knowledge

3.3 Representing Uncertain Knowledge
  3.3.1 Bayesian Logic
  3.3.2 Fuzzy Logic
  3.3.3 Certainty Factors

4 Problem Solving and Search

4.1 Problem Solving
4.2 Direction of Search
  4.2.1 Forward Chaining
  4.2.2 Backward Chaining
4.3 Search Procedures
  4.3.1 Depth-First Search
  4.3.2 Hill-Climbing
  4.3.3 Breadth-First Search
  4.3.4 Best-First Search
  4.3.5 Branch-and-Bound Search
  4.3.6 Generate-and-Test
  4.3.7 Heuristic Search
4.4 Backtracking

5 Languages, Environments and Shells

5.1 Languages
  5.1.1 Prolog
5.1.2 Lisp 74
5.2 Environments 81
  5.2.1 Poplog 81
5.3 Expert System Shells 82
  5.3.1 EMYCIN 82
  5.3.2 Micro-Expert/Savoir 83
  5.3.3 Expert 84
  5.3.4 ESP/ Advisor 84
  5.3.5 Expert-Ease 85
  5.3.6 Ex-Tran7 86

6 Problem Solving Tools Used

6.1 Introduction 87
6.2 Classifiers 87
  6.2.1 Introduction 88
  6.2.2 ID3 88
  6.2.3 Limitations of ID3 96
  6.2.4 Expert-Ease 96
  6.2.5 Ex-Tran7 97
6.3 Production Systems 98
  6.3.1 Theory 98
  6.3.2 BAGGER 99
6.4 Advice Giving Systems 103
  6.4.1 Micro-Expert 103

7 Design of Plants Handling Hazardous Material

7.1 Introduction 109
  7.1.1 Design Theories 109
  7.1.2 Design Problem Characteristics 112
7.2 Topics Investigated 113
  7.2.1 Candidate Topics 113
  7.2.2 Emergency Isolation Valves 116
  7.2.3 Fire Protection of Storage Tanks 117
  7.2.4 Flare Systems 119
  7.2.5 Fugitive Emissions 120
  7.2.6 Pressure Relief and Blowdown 121
  7.2.7 Static Electricity 122

iii
7.2.8 Expertise in Candidate Topics 124
7.3 Short Studies 124
  7.3.1 General 124
  7.3.2 Fugitive Emissions 125
  7.3.3 Static Electricity 128
7.4 Summary 130

8 Hazard Identification

8.1 Introduction 131
8.2 Hazard Identification 131
  8.2.1 General 131
  8.2.2 Coarse Scale 132
  8.2.3 Expertise in Coarse Scale Hazard Identification 133
  8.2.4 Fine Scale 135
  8.2.5 Expertise in HAZOP 136
8.3 Fault Propagation 138
  8.3.1 Fault Trees 138
  8.3.2 Fault Propagation 139
  8.3.3 Alarm Analysis 140
8.4 Computer-Aided Hazard Identification 141
  8.4.1 Computer-Aided Fine Scale Hazard Identification 142
  8.4.2 Deficiencies of the Computer Aid 146
  8.4.3 Expert System Features 147
8.5 Expert Systems and Hazard Identification 148
  8.5.1 Expert Systems and Coarse Scale Hazard Identification 148
  8.5.2 Expert Systems and Fine Scale Hazard Identification 148
8.6 Summary 149

9 Fire Protection of Storage Tanks

9.1 Overview 151
9.2 Aspects of Storage 152
  9.2.1 Segregation 152
  9.2.2 Bunds 153
11.2.1 Physical Design 219
11.2.2 Flare System Awareness Aid 219
11.2.3 Hazards in Flaring 227
11.3 An Account of Flare Literature 228
  11.3.1 General 229
  11.3.2 Specific Aspects 229
11.4 Discussion with an Expert 232
11.5 Rules for Flare System Design 235
  11.5.1 Rules Derivation 235
  11.5.2 Rules Summary 237
  11.5.3 Conflicts 245
11.6 Flare System Expertise and Design Strategy 246
  11.6.1 Flare System Design Expertise 246
  11.6.2 Flare Design Strategy 250
11.7 A Prolog Program for Flare System Design 252
  11.7.1 Program Output 253
11.8 Summary 258

12 Pressure Relief and Blowdown

12.1 Introduction 259
  12.1.1 Legal and Code Requirements 259
  12.1.2 Some Definitions 260
12.2 Aspects of Pressure Relief Design 262
  12.2.1 Causes of Overpressure 262
  12.2.2 Determination of Individual Relieving Rates 265
  12.2.3 Pressure Relief Valves 267
  12.2.4 Relief Disposal Systems 269
  12.2.5 Location and Position of Relief Valves 270
12.3 Account of the Literature 273
  12.3.1 General 273
  12.3.2 Particular Cases 274
  12.3.3 Relief Devices 275
  12.3.4 Alternatives to Pressure Relief 276
12.4 Pressure Relief Design 277
  12.4.1 Alternatives 277
  12.4.2 System Definition 277
  12.4.3 Pressure Relief Situations 278
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4.4 Overpressure Sources</td>
<td>278</td>
</tr>
<tr>
<td>12.4.5 Pressure Protection Principles</td>
<td>279</td>
</tr>
<tr>
<td>12.4.6 Pressure Protection - Individual Solutions</td>
<td>280</td>
</tr>
<tr>
<td>12.5 Design Rules</td>
<td>280</td>
</tr>
<tr>
<td>12.5.1 Rules for Relief Valve Selection</td>
<td>280</td>
</tr>
<tr>
<td>12.5.2 Rules for Pressure Relief in General</td>
<td>282</td>
</tr>
<tr>
<td>12.5.3 Rule Types</td>
<td>285</td>
</tr>
<tr>
<td>12.6 System Decomposition</td>
<td>286</td>
</tr>
<tr>
<td>12.7 Example Problem - Benzene Plant Line Diagram</td>
<td>288</td>
</tr>
<tr>
<td>12.7.1 List of Vessels</td>
<td>288</td>
</tr>
<tr>
<td>12.7.2 Pressure Breaks</td>
<td>290</td>
</tr>
<tr>
<td>12.7.3 Rules for Decomposition</td>
<td>291</td>
</tr>
<tr>
<td>12.8 Pressure Relief Design Strategy</td>
<td>292</td>
</tr>
<tr>
<td>12.8.1 Pipework</td>
<td>293</td>
</tr>
<tr>
<td>12.8.2 Vessels</td>
<td>294</td>
</tr>
<tr>
<td>12.8.3 Outline of the Design Process</td>
<td>294</td>
</tr>
<tr>
<td>12.8.4 Subsidiary Topics and Design Expertise</td>
<td>295</td>
</tr>
<tr>
<td>13 Problem Characteristics</td>
<td></td>
</tr>
<tr>
<td>13.1 Introduction</td>
<td>298</td>
</tr>
<tr>
<td>13.2 Problems Studied</td>
<td>298</td>
</tr>
<tr>
<td>13.2.1 Fire Protection of Storage Tanks</td>
<td>298</td>
</tr>
<tr>
<td>13.2.2 Emergency Isolation Valves</td>
<td>300</td>
</tr>
<tr>
<td>13.2.3 Flare Systems</td>
<td>301</td>
</tr>
<tr>
<td>13.2.4 Pressure Relief and Blowdown</td>
<td>303</td>
</tr>
<tr>
<td>13.2.5 Hazard Identification</td>
<td>304</td>
</tr>
<tr>
<td>13.3 Summary Table</td>
<td>305</td>
</tr>
<tr>
<td>13.4 Problem Characteristics</td>
<td>306</td>
</tr>
<tr>
<td>13.4.1 Characteristics of Problems Studied</td>
<td>306</td>
</tr>
<tr>
<td>13.4.2 Characteristics of Design</td>
<td>307</td>
</tr>
<tr>
<td>13.4 Expert Systems for Design</td>
<td>308</td>
</tr>
<tr>
<td>14 Conclusions</td>
<td></td>
</tr>
<tr>
<td>14.1 Characteristics of Candidate Topics</td>
<td>309</td>
</tr>
<tr>
<td>14.3 Characteristics of Design</td>
<td>310</td>
</tr>
<tr>
<td>14.3 Expert System Tools</td>
<td>310</td>
</tr>
</tbody>
</table>
14.4 Highlights of the Study

15 References

15.1 Chapter 1
15.2 Chapter 2
15.3 Chapter 3
15.4 Chapter 4
15.5 Chapter 5
15.6 Chapter 6
15.7 Chapter 7
15.8 Chapter 8
15.9 Chapter 9
15.10 Chapter 10
15.11 Chapter 11
15.12 Chapter 12

16 Appendices

A ID3 Example
B Prolog 'BAGGER' Program
C Prolog Program Listing for Storage Tank Fire Protection
D Ex-Tran7 Listings for EIV Installation
E Micro-Expert Listing for EIV Installation
F Prolog Program Listing for Flare System Design
# Index to Figures

<table>
<thead>
<tr>
<th>Figure Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Anatomy of an Expert System</td>
<td>23</td>
</tr>
<tr>
<td>4.1 Transformation of CAB to ABC by Search</td>
<td>57</td>
</tr>
<tr>
<td>4.2 Depth-First Search</td>
<td>59</td>
</tr>
<tr>
<td>4.3 Hill-Climbing</td>
<td>61</td>
</tr>
<tr>
<td>4.4 Breadth-First Search</td>
<td>63</td>
</tr>
<tr>
<td>4.5 Branch-and-Bound</td>
<td>65</td>
</tr>
<tr>
<td>5.1 Prolog Tree Representation</td>
<td>73</td>
</tr>
<tr>
<td>5.2 CAR and CDR Operation</td>
<td>78</td>
</tr>
<tr>
<td>6.1 Decision Tree for ID3 Example</td>
<td>91</td>
</tr>
<tr>
<td>6.2 Experiment with 'n' Equally Likely Outcomes</td>
<td>92</td>
</tr>
<tr>
<td>8.1 Mini-Fault Tree</td>
<td>140</td>
</tr>
<tr>
<td>9.1 Storage Tanks and Vessels</td>
<td>155</td>
</tr>
<tr>
<td>9.2 Fixed Roof Tank with Floating Deck</td>
<td>156</td>
</tr>
<tr>
<td>9.3 Perimeter Sprinkler for Coned Roof Tank</td>
<td>159</td>
</tr>
<tr>
<td>9.4 Typical Deluge System</td>
<td>160</td>
</tr>
<tr>
<td>9.5 A Sprinkler System for an LPG Storage Sphere</td>
<td>161</td>
</tr>
<tr>
<td>9.6 Algorithm for Storage Tank Fire Protection</td>
<td>167</td>
</tr>
<tr>
<td>10.1 Possible Approaches to EIV Installation on an Existing Plant</td>
<td>187</td>
</tr>
<tr>
<td>10.2 General Arrangement of Piping to a Spherical Vessel</td>
<td>188</td>
</tr>
<tr>
<td>10.3 Block Diagram for EIVS</td>
<td>207</td>
</tr>
<tr>
<td>11.1 Typical Flow Scheme of an Elevated Flare Installation</td>
<td>220</td>
</tr>
<tr>
<td>11.2 Typical Flow Scheme of a Ground/Elevated Flare Installation</td>
<td>221</td>
</tr>
<tr>
<td>11.3 Graphical Representation of Controlled Blowdown</td>
<td>226</td>
</tr>
<tr>
<td>Showing How Blowdown can be Controlled to Avoid Initial Peak Flow</td>
<td></td>
</tr>
</tbody>
</table>
11.4 Base of Flare Stack

12.1 Typical Pressure Relief Valve Installation
12.2 Typical Pressure Relief Valve Installation on a Process Vapour Line
12.3 Benzene Plant Line Diagram
## Index to Tables

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 'BAGGER' Knowledge</td>
<td>100</td>
</tr>
<tr>
<td>9.1 Attributes and Values for Storage Installations</td>
<td>180</td>
</tr>
<tr>
<td>10.1 EIV Data for an Olefin Plant</td>
<td>189</td>
</tr>
<tr>
<td>10.2 Expert-Ease EIV Rule</td>
<td>192</td>
</tr>
<tr>
<td>10.3 Rules Application to EIV Data</td>
<td>198</td>
</tr>
<tr>
<td>10.4 Revised EIV Data</td>
<td>200</td>
</tr>
<tr>
<td>10.5 Revised Expert-Ease Induced Rule</td>
<td>203</td>
</tr>
<tr>
<td>10.6 Ex-Tran7 Induced Rule</td>
<td>205</td>
</tr>
<tr>
<td>11.1 Discussion with an Expert</td>
<td>232</td>
</tr>
<tr>
<td>11.2 Rules for Flare System Design</td>
<td>237</td>
</tr>
<tr>
<td>11.3 Types of Flare Rules</td>
<td>244</td>
</tr>
<tr>
<td>11.4 Expertise in Flare Rules</td>
<td>245</td>
</tr>
<tr>
<td>11.5 Expertise for the Design of a Flare System</td>
<td>249</td>
</tr>
<tr>
<td>11.6 A Strategy for Flare Design</td>
<td>250</td>
</tr>
<tr>
<td>12.1 Utility Failures</td>
<td>264</td>
</tr>
<tr>
<td>12.2 Decomposition of the Benzene Plant Line Diagram</td>
<td>288</td>
</tr>
<tr>
<td>13.1 Summary of Principal Topics Studied</td>
<td>305</td>
</tr>
</tbody>
</table>
1 Introduction

This project is a study of the expertise in the design of chemical plants which handle hazardous materials. It has been approached by studying various topics in hazardous plant design together with the methods employed in expert systems. In this opening chapter we give an introduction to design, particularly for hazardous plants, safety and loss prevention, computer problem solving, expert knowledge and expert systems. In the final section, the objectives of the work are outlined.

There are various works which deal with engineering design as a subject in its own right. These include Simon (1975) and Turner (1984), and Rudd and Watson (1968) who deal with design with particular reference to Chemical Engineering. In this chapter design is considered generally, and more specific details are given in Chapter 7.

1.1 Plant Design

The design of a large plant or plant extension is a complex task which takes some time to complete and involves a number of stages and experts of different disciplines.

Design evolves from research and development which defines the requirements and constraints of the system. Also considered at this early stage is the economic viability of the project in terms of production costs and sales returns. The decisions taken early on are crucial and will leave little scope for alteration later on, as these may be expensive and time consuming. Also to consider in the first stages is the safety of the plant with the concept that it is better to eliminate a hazard rather than devise a means of controlling it in mind. The safety of the plant is therefore determined largely by the quality of the design from the outset.

The design process can be generalized in terms of knowledge of the characteristics of particular chemicals, operations, and processes etc. This kind of knowledge is
essential for process design and for hazard identification.

The design team should be properly organized, since the team will be made up from various different disciplines, such as chemistry, mechanical engineering, electrical engineering as well as chemical engineering. Individual responsibility should therefore be well defined and the nature of each aspect of the design that one person is concerned with should not take them outside of their domain of expertise.

Stages in the design will be outlined in detail later in this work, but as a broad indication, they are:

Research and development
Process design -
  process flowsheet
  detailed process design
Engineering design and equipment selection

Lees (1980) states that design is an iterative process since as more information becomes available and as constraints and opportunities are recognized, modifications to the design may be needed. Further, the project must be scheduled and coordinated properly, and for this, use is made of the techniques of Critical Path Scheduling (CPS) and of Project Evaluation and Review Technique (PERT).

Other general aspects to the design of a plant are listed below:

Design experience
Design information
Design standards and codes
Design communication and documentation
Hazard identification
Reliability assessment
Design modification
Computer aided design
1.1.1 The Design of Plants Handling Hazardous Materials

It is useful at this stage to describe design from a hazards viewpoint. We mentioned above that it is better to eliminate the hazard rather than try to control it using a special device. This concept is known as inherently safer design. However, some hazards cannot be avoided, particularly if we consider that some chemicals are more flammable than others, or more toxic and some unit operations are more hazardous than others. Therefore the degree of hazard is a function of the process chosen and the inventory present on the site, since by limiting the inventory, the process can be made inherently safer. The disaster at Flixborough illustrates this well. The scale of the explosions was due to the holdup of large quantities of flammable liquid at high temperature and pressure. The subsequent enquiry recommended that consideration be given to reducing the inventories on process plants.

Other aspects which will make a plant inherently safer include the selection of the process, since some processes are inherently more operable than others, and fail safe design, which refers to the design of control and solenoid valves etc., in the event of a utility failure such as electricity or instrument air. Another aspect is so-called 'second chance design'. This means that there is a second line of defence. Features here include plant layout, pressure system design and alarms and trips. It can be illustrated by normal features of plant design. For example, bunding and drainage facilities, pressure relief and blowdown systems on pressure vessels, isolation of loss of containment, the use of double mechanical seals on pumps and by strict following of maintenance and operation procedures, such as purging again after a delay in lighting a furnace.

The design of engineering equipment such as reactors, distillation columns and furnaces together with an awareness of the operating conditions and the utilities available all go toward the successful completion of a plant design.
1.2 Safety and Loss Prevention

The phrase 'Safety and Loss Prevention' has taken on a new meaning in the chemical and process industries in the last 20 or so years, and particularly since the Flixborough incident of 1974. Traditionally, safety was often only considered on a plant as a remedial action, i.e. the fault was put right once an accident had occurred. In recent years, the action has tended towards the reverse of this. The emphasis now is on making plants safe from the moment they start operations and on designing plants which are intrinsically safe, i.e. safe by their very nature. There are many reasons for the change in attitude, namely that plants have got larger, with more quantity of dangerous materials in operation, Government legislation, companies own moral regard to safety for their workers, the environment and the public, and the heavy financial losses that are incurred when a catastrophes occurs. This is where the term 'loss prevention' derives from. It is an insurance term meaning the financial loss caused by accidents. It is a forward looking process - one that anticipates hazards and their effects before they occur. It is an all-embracing term that includes all areas of the plant - from first designs to operations, all personnel - from the designers to operators and maintenance workers, and to all departments - from Safety to Research and Development. Because chemical and process plants are bigger today than ever before, the need for loss prevention is greater than ever before.

Safety and loss prevention should be of prime consideration in plant design and should be considered at the earliest possible time in the designing of a plant. The reasons for this have been outlined above, but also under the Health and Safety at Work Act (1974), company management is under legal obligation for the safeguarding, as far as is reasonably practicable of the health and welfare of employees and the public where they are affected. The safeguarding of the plant itself is not only the responsibility of the management, but also the employees.
Lees (1980), has characterized loss prevention under the following headings:

1) A concern with depth of technology and associated major hazards
2) An emphasis on management
3) A systems rather than trial and error approach
4) A concern to avoid loss of containment resulting in major fire, explosion or toxic release
5) The development of techniques for the quantification of hazards
6) The principle of risk criteria and the quantification of hazards
7) The development of techniques for the quantification of hazards
8) The use of the techniques of reliability engineering
9) The principle of independence in critical assessment and inspections
10) The planning of emergencies
11) A critique of traditional practices or existing codes or regulations where these appear out-dated by technological change

1.2.1 Loss Prevention and Plant Design

Loss prevention in plant design comes under the following general headings (Coulson and Richardson, 1985):

1) Identification and assessment of hazards
2) Process control - prevention of deviations
3) Hazard control - containment
4) Loss limitation - pressure relief

Many aspects of safety on process plant can be considered at the design stage. Equipment has to be specified and selected. Designers consult relevant British Standards or codes of practice. Other appropriate codes relevant to petroleum plant are published by the American Petroleum Institute and the Institute of Petroleum, in the Refinery Safety Codes.
Also, each individual company will produce its own codes, although these will generally be confidential.

The hazards met in design must be effectively dealt with. This is also the stage where expert design can reduce the need for complex and costly control and protective devices. Kletz (1984) makes this point in his book published by the IChemE on Cheaper, Safer Plants. The point is made that simplifying the design can lead to increased safety.

Important aspects of safety in plant design are listed generally by the IChemE (1984). These are:

1) Hazard and Operability Studies - an effective technique for examining plant and identifying hazards at an early stage in the design. All aspects of the plant can be considered by questioning the process under key headings such as "more-of" (flow, temperature, pressure), "none" (flow, temperature, pressure), etc. A detailed study of HAZOP is given also by Kletz (1983). More on HAZOP is discussed in Chapter 8.

2) Reactor Design - potentially a major source of hazard particularly when exothermic reactions and reactions containing large amounts of flammable material occur.

3) Plant Layout - a very important aspect, where it is necessary to provide adequate space between items of plant, storage etc., and good roadways for fire-fighting appliances. Also to consider here is the danger to the public of a plant situated in a densely populated area. A comprehensive discussion of plant layout is given by Mecklenburgh (1985).

The above aspects are just a small selection of safety and loss prevention in plant design. We will go into more detail later, but extensive discussions can be found in Lees (1980).

This section has emphasised the need for forward planning when designing process plant. This is a fundamental aspect in
this project and one of the prime aims is to make the designing of safety into a process easier from an earlier stage. If the engineering design team can use a computer containing expert knowledge before going into the calculations, then the team will be all the more aware of the hazards inherent in the process.

1.3 Computer Problem Solving

To solve any kind of problem, either manually or by using a computer, the solver needs to have a clear idea and statement of the problem to be solved. This is particularly true of computers which are programmed in the conventional sense using such languages as FORTRAN and BASIC. Traditionally, computers were very large number-crunching devices able to solve mathematical problems of increasing complexity as time has gone by. Today, computers are able to solve very complex mathematical problems very quickly indeed. However, computers are now able to solve text oriented problems, where calculations are not an inherent part of solving the problem. Here, yes/no type answers are needed in answering relevant questions in order to work through to a solution to the problem. These kinds of problems are about the environment in which we live such as medicine, engineering design classification and diagnosis, problems that computers traditionally have not been able to solve. The so called 'creative computer' is still unable to solve our economic and social problems, but some would argue that this is not such a bad thing.

The level of difficulty of a problem must also be considered. Getting a robot to guide a spacecraft to the moon is a relatively straightforward operation, but to get the robot to go to the shop and buy a newspaper is a difficult problem. Some would argue that for a computer to solve such problems of the real world, then it would have to be intelligent. This is not quite true, because a computer will only do what the programmer asks it to. However, for a human to solve some arithmetic, that person will use his intelligence. If a machine does the same problem, in a fraction of the time, then it
too is intelligent. The argument appears never-ending, but the point we are making is that for a computer to solve problems that it normally requires specialists to solve, it has to be able to appear to mimic the expert in its line of reasoning. The computer will still need to be programmed, but by using specialist people, and not in the conventional sense.

Programs generally need a clear problem definition and a start and an end to the program in order to find a solution. Steps are set out in a clear and logical order for the computer to follow. Fifth Generation programs, and particularly expert systems (see section 1.5) are not written in this way. This is because the types of problems are different. They are not clear-cut in both definition and solution. Languages (such as Lisp and Prolog) are designed in order to solve such problems. Using these languages, it is possible to write one line of program code and get the compiler to make sense of it. Also lines of program can be inserted at any point in the program without upsetting the way the program will solve the problem. A logical start or end to the program is also not necessary in the conventional sense.

The types of problems that Fifth Generation programs solve, then, are quite different in nature from what we are used to. Such problems are often not fully defined have many different solutions, are difficult to solve for a non-expert and are characterized by uncertainty. In order to solve such problems, large amounts of knowledge are needed, making structured languages inappropriate for problem solution.

Problem selection becomes very important in this field. Feigenbaum (1984) states:

"...problem selection - the selection of a domain in which to attempt the building of an expert system - is an art."

Feigenbaum goes on to say that the problems must be chosen so that they reflect the state of the art in knowledge enginee-
ring. If the problems fit precisely then there are no difficulties, but if they are slightly beyond present capabilities, then the art is pushed. If they are well beyond what other people are doing then time and effort are wasted and little is accomplished.

1.4 Expert Knowledge

1.4.1 Expertise

An expert in one particular field is considered as a specialist, one who will be at the top of his profession and one who is recognised as having much specialised knowledge about his domain. For example, a medical doctor is considered an expert, particularly if he specialises in one particular area of medical science. Expertise consists of knowledge, understanding problems in the domain, and skill at solving these domain problems. Because experts are knowledgeable, they achieve outstanding performances that few people could emulate. However if computer programs could embody an expert's knowledge, then they too will achieve outstanding performances. If such programs are going to achieve success and appear to 'think' like the expert, then not only do they have to contain the knowledge, but also the problem-solving methodology. Added to this, difficult and interesting problems generally do not have tractable and algorithmic solutions, since important tasks originate in complex physical or social domains.

1.4.2 Knowledge

Knowledge can be divided into two parts - that of the facts of the domain and that of heuristic knowledge. Facts domain knowledge consists of facts and theories that can be found in textbooks, literature, papers or in the classroom. Equally important is heuristic knowledge which is the knowledge of good practice and good judgement, and is central to the task of solving expert problems. Hayes-Roth et.al. (1983) refer to this as 'hidden knowledge', since it is rarely
written down, and is knowledge acquired by the expert after years of experience. It is usually unique to one particular expert, and is in the form of 'rules-of-thumb' or heuristics. Heuristics enable the human expert to make judgements and educated guesses. They are used for recognising promising approaches to problems and to deal effectively with incomplete or errorful data.

Heuristics are a very important aspect of expert knowledge, and are central to the theme of this study. Interpreting and copying this kind of knowledge from the human expert is recognised as one of the biggest bottlenecks in the designing of expert systems (Feigenbaum, 1984). The reasons for this are that heuristics are by nature vague, unexplainable, and difficult to extract from the expert. Experts often use heuristics without even realising it, and their basis for using them is often because it worked the last time they tackled such a problem. Another problem with the use of heuristics is representation in the computer program. The two problems of acquisition and representation are dealt with in more detail later. Lenat (1983) has presented work on the nature and use of heuristics.

1.5 Expert Systems

Expert systems are a branch of the general area of artificial intelligence. Artificial intelligence is about making machines behave in a way similar to that of humans. Winston (1984) has given the following definition of artificial intelligence:

"Artificial intelligence is the study of ideas that enable computers to be intelligent".

Applied artificial intelligence has applications in natural language processing, robotics and expert systems, along with computer chess playing and puzzle-solving. Artificial intelligence has its background in the 1940s when a prominent British scientist, Alan Turing, stated that there was a need
for a machine which was based on logical operators such as 'and', 'or' and 'not' (Harmon and King, 1985). Today, many countries are concerned with A.I. development, and particularly expert systems work. Notably, the United States, Britain, the E.E.C. and Japan with their Fifth Generation Program are concerned with work in this and related fields.

Knowledge-based expert systems provide the means for computers to aid people with analysing problems and making decisions. There are many commercial applications undergoing development and many in existence already, notably MYCIN, DENDRAL and PROSPECTOR. Typically, expert systems are helping doctors diagnose diseases, geologists locate ore deposits and general trouble-shooting in many different fields.

An expert system has been defined as:-

"...an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution."

Feigenbaum
(in Harmon and King, 1985)

Traditionally, an expert system was built by debriefing a recognized human expert so that his expert knowledge could be captured and coded into computer programs. This is termed knowledge engineering, and knowledge engineers are usually employed for this task. Also, other knowledge acquisition techniques are used. Such techniques involve 'teaching' a system expertise by feeding in rules or examples of any domain into an existing structure. Such systems are called shells. Other systems contain knowledge of a difficult decision-making situation that is quite useful, but not equivalent to that of a human expert. Machine learning is another promising method of expert system development, in which the program acquires knowledge from past experiences. Aspects of machine learning include classification systems and learning systems, and these receive attention in this work.
Expert systems are usually built using specialist artificial intelligence languages, such as Prolog or Lisp, or by using a shell, which meets the requirements of the particular domain. As explained in the previous section, these languages do not run in the same way as languages such as FORTRAN or BASIC. Artificial intelligence languages are symbolic, are highly interactive and are capable of being programmed to give mid-run explanations.

Expert systems are generally made up of a knowledge-base, in the form of rules together with an inference engine, which is a program which works out the logical sequences of all the rules taken together. Rules can be unambiguous, such as 'IF this AND that THEN some result', or they can be more vague involving probabilities, such as 'IF (to some degree) this AND (to some degree) that THEN (to some degree) result'. The machine works through the rules, asks the user for information and then gives its conclusions.

Expert systems and inference procedures will be discussed in more detail in later chapters.

1.6 Project Objectives

The main objective of this project is to gain a better understanding of the expertise involved in the design of plants handling hazardous materials. Our aim is to develop methodologies in certain aspects of safety in the process industries with a view to improved methods of computer aided design, including expert systems.

There are already a number of large computer aided design packages commercially available, but these tend to deal with the detailed design calculations involved for plant equipment, such as reactors, heat exchangers and distillation columns. There is little available which specifically deals with hazards work, apart from reliability and availability programs and fault tree programs. Since the key element in hazard identification is to identify the key problems before the
detailed design, there is obviously a need for computer aided design programs in this area. These could be used as front-end programs, using existing packages as back-end programs.

Another point to note is that generally speaking we are not dealing with new plants built on green-field sites, but existing plants where modifications and extensions are taking place.

1.6.1 Problem Identification

It is indicated above that the most beneficial contribution to this field is to aid the identification of hazards early on in a design project. This involves a coarse scale hazard identification technique. This is what happens in manual plant design, so a complete computer aided design approach would have to have an equivalent of this function.

There are a number of ways of approaching this problem. One is to try to list the generic hazards that always tend to occur. We could also investigate case histories, one in which the hazard was found during the design, one where accidents were reported. A third approach is to look at the index-based methods such as the Dow and Mond indices and try to develop generic features on which these are based. Also, we have tried to concentrate an approach to specific design tasks.

An important point to note is that the work reported here is concerned with various aspects of the design problem. The expertise is therefore from literature sources. If the project was concerned with the development of a single expert system, then the work would have probably proceeded in a different way. We are concerned with the type of knowledge used by designers, and on its representation, rather than knowledge elicitation.

Earlier sections of the introduction have emphasised the characteristics of the design process, design solutions and of expert systems. It is at this point that we should bring the
subjects together and justify the potential for design heuristics for plants handling hazardous materials. Such design is not about lengthy calculations involving iterations and convergences, but about encapsulating expertise from the literature, texts, examples and the experts. The characteristics of this knowledge seem to show that one of the most promising approaches to problem solving is in the development of methodologies with the idea of building expert systems.
In Chapter 1, expert systems were introduced. In this chapter, the components of an expert system are explained, including the method of solving problems, the knowledge base, and facilities characteristic of most 'classical' existing systems, such as explanations. The first section deals with fields of application of expert systems today together with applications in chemical engineering.

2.1 Applications of Expert Systems

When experts are in great demand, but in short supply, a computer based consultant can help to amplify and disseminate the needed expert. Expert systems capture practical knowledge, hard to pin down and rarely found in text books. Below are three areas where expert systems have been proven and are advantageous, as given by Wiess and Kulikowski (1984).

a) Medicine - Here, expert systems are generally used for a second opinion or to qualify a doctor's suspicions. As specializations grow and problems become more complex, expert systems are playing an ever increasing role.

b) Oil Exploration - There has long been a shortage of expert well-log analysts. The idea of capturing the expertise of the best analysts in a computer model is an attractive alternative to training new specialists and retraining existing ones.

c) Equipment Repair - Again, there is a shortage of trained experts, particularly in the area of computer repair. As technology becomes more complex, the problem becomes progressively worse. The dynamic component of the expertise involved is hard to capture in manuals, making expert systems very attractive.

Sell (1985) states that there are over 50 expert systems in existence, although there is some doubt as to whether some of them can be thought of as expert systems (Andow, 1985).
There are, however, four systems that are considered to be 'classical', and these are described below.

2.1.1 DENDRAL

Bramer (1984) states that DENDRAL was the first expert system. It is probably still the best known successful system. The project as a whole includes three programs, HEURISTIC DENDRAL, META-DENDRAL and CONGEN, and was developed at the Stanford Heuristic Programming Unit by Feigenbaum, Buchanan and others, in association with the Stanford Mass Spectrometry Laboratory from 1965 onwards. The system is in daily use in universities and industry (Hayes-Roth, et.al. 1983).

The purpose of DENDRAL is to derive chemical structures from data available to physical chemists. A mass spectrometer is used to determine the constituent atoms of a compound and their relative frequencies. The physical chemist must then determine the arrangement of the atoms in the molecule. No algorithm is available to do this.

The constraints of the spectrograph are fed to the CONGEN (CONstrained structure GENerator) part of the program, where rules are employed to arrive at a list of possible candidate structures. These structures are then compared with the original spectrograph.

The HEURISTIC DENDRAL part eliminates lengthy and error prone work by the chemist in order to come up with a final structure. META-DENDRAL contains high level rules and is used to examine data and discover rules for determining molecular structure. Rules are discovered then refined before being released as the working set. META-DENDRAL is an example of an inductive inference system as described by Quinlan (1982).

Sell (1985) gives an indication of the success of the DENDRAL project:

"DENDRAL is a success story. The results derived
from its use are cited in over 50 scientific journals, which attests not only to its usefulness but also to its scientific credentials... The number of its users was expanding so rapidly that in 1983 a separate company was set up for its distribution and continued enhancement."

2.1.2 MYCIN

Developed from 1975 onwards by Edward Shortliffe and others, the MYCIN project, like DENDRAL was based at the Stanford Heuristic Programming Project in collaboration with the Stanford Medical School. MYCIN is a medical diagnostic program for blood and meningitis infections, and also recommends effective drug treatment on the basis of interactive dialogue with the physician. It contains more than 400 rules of the IF <situation> THEN <action> type. Associated with these is a degree of certainty indicating the level of confidence in the rule. Bramer (1982) gives the following example of a rule:

RULE 85

IF:
1) The site of the culture is blood, and
2) The gram stain of the organism is gramneg, and
3) The morphology of the organism is rod, and
4) The patient is a compromised host,
THEN:
There is suggestive evidence (0.6) that the identity of the organism is pseudo-aeruginosa.

The certainty factor, 0.6, is a probability-like value in the range 0 to 1.

A physician's task in tackling a problem of this kind is to establish four facts:

1) Does the patient suffer from bacterial infection?
2) What organism is responsible?
3) Which drugs may be appropriate?
4) Which drugs do I administer?

MYCIN is designed to aid all four decisions on the basis of patient data and test results, reaching a conclusion about all four questions. A line of reasoning can be displayed and references to articles and publications relevant to the case.

MYCIN uses 'backward chaining' (see section 4.2.2) and an implicit AND/OR tree. Medical diagnoses ('goals') are linked with user-supplied data ('leaf-nodes') to make sure that the questions are "focussed" towards a particular hypothesis. A model of inexact reasoning is used to propagate certainty factors through the tree. Bramer (1982) gives an example session with MYCIN.

MYCIN has shown to be as good as specialists in the field of blood disease diagnosis. However, it has not found regular use in the medical fields. There are various reasons for this, (Sell, 1985) but it is thought that experts do not want to use a machine for tasks that they are well capable of performing themselves. Other reasons may be that MYCIN requires a large machine, takes 20-30 minutes per consultation and it is ignorant of a patient's case history. MYCIN has, however, had favourable attention as a teaching aid, mainly due to its explanation facilities and its use of references related to the field.

2.1.3 PROSPECTOR

This was developed by Duda, Hart and others and is an aid to geologists searching for ore deposits and for the evaluation of the mineral potential of large geographic areas. It was developed at the Stanford Research Institute from 1978 onwards (Duda, 1979). Like MYCIN it is a conversational system and uses several geological models. These include three different sandstone models, uranium deposit models and porphyry copper models.
The knowledge in PROSPECTOR is in the form of an inference network of relations between field evidence and geological hypotheses. Users are able to give a certainty factor about evidence on the scale -5 (indicating evidence is certainly absent) to +5 (indicating evidence is certainly present). Three kinds of evidence are present in the inference network:

1) Logical Relations - Uses standard Boolean connectives of AND, OR and NOT. They indicate that the truth value of a hypothesis depends totally on those assertions that define it.

2) Plausible Relations - Bayesian probability theory is used to determine changes in the probabilities of a number of related hypotheses. Plausible relations are used for general cases where the proving or otherwise of a relation provides evidence to support the change.

3) Contextual Relations - Used to express necessary conditions that are needed to be established before assertions can be utilized in the reasoning process.

PROSPECTOR is said to be a very accurate system. Sell (1985) quotes an example when the models were submitted to tests against known sites of exploration and against the judgement of experts. PROSPECTOR was found to be in agreement to within 7%. It is a remarkably cheap system, with one consultation costing only a few dollars.

2.1.4 R1

R1 (also known as XCON) is said to be a very successful expert system (Sell, 1985). It was developed by McDermot and workers at the Carnegie-Mellon University with funding from the Digital Equipment Corporation (DEC). R1 is a knowledge based VAX configuration system, and is used for aiding DEC staff to work out customer requirements when buying a VAX computer system. When the VAX was first marketed, the customer had a great choice of equipment, depending on his particular requirements. The orders had to be translated into configura-
tions which were plausible. Some equipment needed to be added, such as cables, power supplies and cabinets, other equipment needed greater specification such as translating disc storage requirements into disc units and controllers. Thus, much detail was required, along with an extensive knowledge of the equipment available with the constraints that it had to observe. R1 is able to deal with this kind of problem. At first the system had 400 rules, although this has now grown to 4000. DEC believes that R1 is capable of out-performing people, and 80 more staff would be required if R1 was not available.

R1 is interesting in the context of this work, and it is worth noting some of its features. Out of the systems described above, R1 is the only one that has a design element. The system is a forward chaining one which uses production rules of the 'if-then' format. Conflict resolution plays a part when two rules fire at the same time. In searching for a complete solution, R1 forges ahead and hardly ever backs up. The reason for this is that the rules generally embody enough constraint to prevent going into blind alleys.

Summary

The four systems described above are all very important in expert system research and development. They have shown that expert systems, and the techniques that they employ, are capable of solving real world problems, and they have also given models for other applications to follow. Many systems are based on them, including the so-called 'shells', i.e. the toolsets which make expert system development easier (see section 5.3).

2.1.5 Applications in Chemical Engineering

Expert systems applications in chemical engineering cover a wide range of problems (Banares, 1985a), from pure derivation types to production types. Most problems of the real world lie somewhere between these two extremes. In derivation
problems, the solution is found by a search, whereas in produc-
tion problems, the constraints of the solution are provided,
and the solution generated. Banares feels that as chemical
engineers, we normally come across problems of the derivation
side, although this is not the case with design problems.
Control, diagnosis, monitoring, and repair are all derivation
type problems. The section below gives examples of working
systems in chemical engineering, and is biased towards the
derivation type of problem.

FALCON - Described by Banares (1985a), FALCON is a knowledge
based system under development to help diagnose faults on
process plants. Observed effects of the plant are input to
FALCON, and the output is a series of faults that could be to
blame, with associated probabilities. Both causal and
production rule methods are used.

CONPHYDE - Described by Banares et. al. (1985b), CONPHYDE is a
system for physical property prediction. Data such as composi-
tion, concentration and physical conditions of a chemical
mixture are used. The system has about 37 heuristic rules,
involving six different equations of state and nine activity
coefficients.

PICON - Described by Moore et. al. (1984), PICON is a system
for monitoring and controlling industrial processes. Knowledge
comes from the expert plant operator and the expert process
engineer. PICON is one of the few systems which uses heuristic
knowledge together with 'deep' knowledge (in the form of
mathematical models).

Work by Niida et. al. (1985) has shown some of the appli-
cations of expert systems in process engineering. They state
that the main problem in process engineering is to synthesize
and analyse process systems and related sub-systems by taking
consideration of various criteria, such as annual costs,
operability, safety and maintenance. Specific applications of
their work include a cause-effect analysis of a pressure
relief system. By answering yes/no questions, the user can
establish whether or not there is a possibility of overpressure occurring on a plant item. The system uses backtracking in order to reach a conclusion. This particular application appears to be a fairly straightforward and simple one.

Other work on applications in chemical engineering comes from a variety of sources and workers. They are discussed briefly below. Sachs et al. (1986) have developed a system for relieving the cognitive load on users of information systems generating large amounts of dynamic data, called ESCORT. The system, which is in real time, provides advice to process plant operators to help them handle and avoid crises. Also relevant here is the work of Nelson and Jenkins (1985). They are concerned with an expert system for operator problem solving in process control. Reference is made to the Three Mile Island incident, and they explain how their Response Tree expert system could help reactor operators select a response for an emergency condition. Further, Mills (1984), Haspel (1984) and Donoghue (1984) have described work on control and fault diagnosis in plant engineering.

Lu and Motard (1985) describe an expert system which uses heuristic and evolutionary rules to find an optimal solution for computer aided flowsheeting, and AIDES (Siirola et al. 1971) is a system for process design synthesis. Finally, two other systems which are worthy of mention are REACT (Govind and Powers, 1981) which generates synthetic routes to industrial chemicals and HEATEX (Banares, 1985a) which is a network for reducing energy requirements in process streams.

2.2 Architecture of Expert Systems

The section above shows that there are many kinds of expert systems available in a wide variety of fields. They vary in terms of system design and capabilities. Bramer (1982) has said that the reason for this is partly because the term 'expert system' is not yet precisely defined. However, most important systems have many features in common. Figure 2.1 shows the general anatomy of an expert system.
Figure 2.1 Anatomy of an Expert System
A computer program, written in the more conventional types of languages such as BASIC or FORTRAN has been described by Forsyth (1984) as having:

a) Data
b) Algorithm

which together make up the program. The two key features of an expert system can be compared:

a) Knowledge Base
b) Inference Engine

The two kinds of architecture are clearly similar, but different enough to have significant consequences. Together with the knowledge base and inference engine, expert systems also have other desirable characteristics which make the use and interpretation of such programs straightforward:

a) Limited to one area of expertise
b) Ability to give explanations
c) Able to acquire new rules and update old rules
d) Use of certainty factors

If it was not for these qualities, then computer systems using expert knowledge could not be called expert systems.

2.2.1 The Knowledge Base

The knowledge base is typically made up of facts, rules and assertions. The facts are described as short-term knowledge, and they may change during the course of a consultation. Rules are more permanent and are used to generate new facts, hypotheses and reach conclusions or assertions based on facts provided by the interactive user. One of the great advantages of this kind of user interaction is the fact that it is possible to add to the rule set or update and refine existing rules. The knowledge base is creative and flexible, and directly affects the program performance in reaching decisions.
and conclusions. The so-called inference engine is the method by which the knowledge is controlled, and is therefore, a very important aspect of an expert system. This receives attention later on in this chapter.

There are various ways of representing knowledge in expert systems. The most straightforward method of encapsulating rule-of-thumb knowledge is in production rules. These have the IF-THEN format. Other knowledge representation schemes include decision trees, semantic nets and predicate calculus. These are described in detail in Chapter 3, along with the acquisition of expert knowledge.

2.2.2 User Interface

The third main component of an expert system is the user interface. Michie (1982) points out that this should not be regarded as an optional extra. Michie has warned about possible dire consequences of systems which do not operate within the 'human cognitive window', i.e., whose actions are vague and inexplicable.

The ease of providing a user interface is a point in favour of rule-based programming, and the classical systems such as MYCIN show this well. At any point the user can quiz the system as to why the system made a given deduction or asked a certain question. Explanation is usually enabled by the system retracing the reasoning steps that led to the question or deduction. Forsyth (1984) sums up the importance of a user interface:

"If we are to avoid a succession of Three-Mile-Island type disasters or worse, then our expert systems must be open to interrogation and inspection. In short, a reasoning method that cannot be explained to a person is unsatisfactory, even if it performs better than a human expert."
2.3 Facilities

2.3.1 Control

The control of an expert system is divided into two parts:

Inference -
  Modus Ponens
  Reasoning about uncertainty

Search -
  Backward and forward chaining
  Depth first and breadth first search

In this section, we will concentrate on the inference mechanism. Search is dealt with in Chapter 4.

There are two reasons why an inference mechanism is needed. They are:

1. A rule-based system needs a method of deciding where to start. The rule base can be very large, and there is little point in scanning all the rules in every consultation.

2. Conflicts may arise in the rule base when alternative lines of reasoning come up with the same results. It may be that more than one rule will satisfy the user's requirements. The inference system must be able to decide on which one is best.

A common rule for deriving new facts from rules and known facts is called modus-ponens. This means that if a rule states "if A then B" then it is valid to conclude that if A is true then B is true. This means that in most cases, if the rule is simple then reasoning on it is easily understood. Also, when B is known to be false then it is valid to conclude that A is also false.

Another task of the inference mechanism is to deal with uncertain or incomplete information. When an expert works, he
frequently deals with cases for which some information is missing or is uncertain. If an expert system is to act like an expert, then the inference mechanism must be able to deal with such information. There are a number of ways of achieving this, two of which are described below.

In most cases a rule will fail if all its antecedents cannot be proved true. If the clauses in the rule are connected by 'and', and if one or more of the clauses are unknown then the rule will fail. Alternatively, if the clauses are connected by 'or', then a piece of unknown information need not stop the rule from succeeding.

Another way to overcome unknowns is to provide knowledge in the knowledge base about unknown information. Experts are familiar with providing advice with incomplete information so this expertise can be extracted from the knowledge engineer. This can be done using certainty factors, and uncertain facts will lead to uncertain conclusions.

2.3.2 Questions

The following question types have been found to be most useful in existing expert systems: -

a) Yes/No - It has been found (Sell, 1985) that users of expert systems prefer not to encode answers if it can be avoided. Users prefer to answer with 'y' or 'n' rather than '1' or '2'.

b) Free Answer - Any answer is accepted. This may be useful for accepting a client's personal details.

c) Multiple Choice - This is the kind of question that expert system shells ask, with a choice of answers shown to the user, the user then selecting the answer that he wants.
2.3.3 Explanations

Sell (1985) lists two items that are present in every explanation:

a) the explicandum - that to be explained
b) the explicans - the explanation

There are three types of explanation:

a) Interpretive - gives and expands on the meaning of terms.
b) Descriptive - gives an explanation of a process or structure, by stating facts, relations, criteria etc.
c) Reason-giving - explains by stating laws, codes of practice, causes etc.

An expert system should be able to provide all three types of explanation. It is believed that, at the moment, no expert system can provide explanations as good as those of the expert. The expert and the system differ in that a system cannot take into account the background of the inquirer, and therefore cannot adjust its explanations accordingly.

2.4 Validation

One of the most important aspects of all expert systems is validation. In this respect, expert systems are similar to conventional programs. All programs need to be validated and have the question 'does it work?' asked. Sell (1985) states that the need for validation is more important than in conventional programs, since expert systems are based on inexact reasoning and heuristics. Another reason for validation is the fact that expert systems are often judgemental, and it is difficult to pronounce when a judgement is correct or not.

An expert system can be regarded as valid if it is free from contradiction, can approach any problem within its domain, if it can reach the correct answer and if it can be used with relative ease. In summary, the five basic requirements of
any expert system are:-

Consistency
Soundness
Precision
Completeness
Usability

Each of these will be dealt with in turn in the following sections.

2.4.1 Consistency

Consistency for an expert system demands that it should produce similar answers to similar questions regardless of the circumstances. One of the great assets of expert systems, or computer systems in general, is that they never get tired, or function better according to the day of the week. Consistency is certainly reasonable to expect from an expert system, since if it were inconsistent it would be unusable. Validating a system for consistency is a difficult task. There is no established method of achieving this, although tests can be performed that will increase the confidence in a system. One such test would be to ensure that no two rules arrive at different conclusions given the same conditions.

2.4.2 Completeness

Here, the requirement is that the knowledge base is sufficiently knowledgeable to cover any problem in the domain. Whether this requirement is reasonable will depend on the system. In classification, we would expect all the problems to be solved, but in the case where hundreds of solutions may exist, e.g. chess end games, the requirement is unreasonable.

2.4.3 Soundness

A system is sound if it comes to the right conclusions, or, the conclusions with which the expert agrees. This is a
reasonable requirement and is simple to test for.

2.4.4 Precision

For a system to be precise, the conclusion produced should have a certainty appropriate to the particular problem. This is again a reasonable requirement, since a system should neither be too confident nor too pessimistic. After all, the system might be dealing with the health of a patient, or the financial position of a company. Testing for precision is similar to testing for soundness, with the expert providing test cases, but it must also be practical, e.g. how inaccurate can an answer be and still remain acceptable?

2.4.5 Usability

If expert systems are to make an impact, then they have to be user friendly, just as conventional software must be. However, with expert systems, there is far more scope for ambiguities to creep in, especially if the user is asked to give a degree of certainty in answer to a question, e.g.

"How certain are you that chemical 'X' is toxic (-5...+5)?"

The answer to this might not in fact reflect the toxicity of material 'X'. A better way might be:

"How toxic is chemical 'X'?"

1. Non-toxic
2. Toxic - (but can be handled in a confined space with adequate ventilation)
3. Very toxic (could injure in a confined space)
4. Highly toxic (not to be handled without adequate protection)

So, usability is reasonable and could be critical to the outcome of the system. Testing for usability would take the
form of trainees using the system while being observed by the system builders, before the system goes onto the market.

2.5 Building an Expert System

In this chapter, we have outlined the main features of expert systems. The next chapter describes how the knowledge for such a system is acquired and represented. Acquisition and representation of knowledge, along with validation are the key to building an expert system. As yet, there does not appear to be a specific methodology for building expert systems, although Attarwala and Basden (1985) have produced work on a methodology for constructing an expert system. Their opening paragraph begins:

"Construction of expert systems has so far been seen as a craft or an art, not a science."

They question the traditional method - that of extracting problem solving rules from the expert and encoding them, saying that two experts in the same field may well have a different set of rules of thumb for solving the same problem. Attarwala and Basden propose constructing expert systems from causal models. Their reason is that when experts are asked to give a line of explanation, they do so in terms of cause and effect.

Waterman (1986) gives a more formal approach to expert system construction, although the headings he uses still leave much to the designer's whim. He states that expert system development can be viewed as five phases:

- Identification
- Conceptualization
- Formalization
- Implementation
- Testing

Identification involves both the knowledge engineer and the
expert determining the type and scope of the problem, and how the problem can be characterized. Conceptualization involves deciding what concepts, relations and control mechanisms are needed. Formalization involves deciding in what form the rules are to be represented and what environment to use, and implementation is the knowledge engineer turning the formalized knowledge into a computer program. Testing of the system was dealt with in section 2.4.

Although there is no formalized method of constructing expert systems, there are certain procedures that are common to all such developments. It is a very important area if expert systems are to have any impact, in that if the knowledge is not represented in a suitable form, or the type of control strategy is wrong, then the expert system will not perform efficiently, and will not emulate the expert. Some would argue that there is a need for an expert system to build an expert system, but that, at the present time that is a "catch twenty two" situation. With the present state of the art, the steps in expert system development are:

- Problem definition
- Knowledge acquisition
- Knowledge representation
- Implementation
- Validation

This is a general procedure, but one involving all aspects of development. One danger with the technique of Attarwala and Basden is that, if taken too far, the causal model might take over from an expert's rules of thumb, which would not benefit the system.
3 Knowledge Representation and Acquisition

In this chapter the techniques of representing knowledge in expert systems are described in turn, followed by an account of the rather more complex process of knowledge acquisition.

3.1 Knowledge Representation.

There are two different methods of representing knowledge. As a program, knowledge is represented procedurally and as data, knowledge is represented declaratively. Early representation schemes were dominated by procedural representation techniques, notably GPS (General Problem Solver) by Newell and Simon (1972). They were a natural off-shoot of conventional programs, and were highly efficient. However, as computer power increased, the emphasis shifted to declarative representation for artificial intelligence programs. These procedures have the advantage of ease of understanding, ease of modification and significant clarity, even though they are slower than procedural techniques. For expert systems, the advantages of declarative representation far outweigh that of procedural. The list below outlines the principal declarative methods: -

Semantic Nets
Logical Expressions
Frames
Production Systems
Object-attribute-value-triplets

Of these, semantic networks and production systems have found favour with expert system workers. More comprehensive accounts can be found in Harmon and King (1985), Barr and Feigenbaum (1981) and Sell (1985).

3.1.1 Semantic Nets

Semantic nets were developed by Quillian in 1968, and are
said to be the most general of representation techniques, and
the oldest in artificial intelligence. The method is used in
the system PROSPECTOR.

A semantic net is a series of objects called nodes connected by arcs or links. The nodes represent objects, concepts
or situations of the domain, and the arcs represent the
relations between them. There are no absolute constraints to
apply to semantic network systems, or to show how nodes and
links are named. There are however some conventions used:

Nodes - are used to represent objects and descriptors. Objects
can represent physical or conceptual items and descriptors
provide additional information about the objects, such as
their present state.

Links - may represent any relationship. Links commonly used
are:

Is-a - used for class-instance relationships, such as the
name of an object.

Has-a - used for identifying nodes that are properties of
other nodes.

Some links define the state of objects in nodes, called
definitional links, others capture heuristic knowledge and
provide reasons.

Advantages semantic nets offer are their flexibility to
define new nodes and links, and inheritance such that one node
can inherit the properties of other nodes. Harmon and King
(1985) point out that the concepts of semantic nets, links,
nodes and inheritance are all related to research into how
humans store information.

Semantic nets are very popular in artificial intelligence
for representing knowledge. However, most work in this area
now involves an extension of the idea in the form of frames.

34
It is stated by Barr and Feigenbaum (1981) that the simple idea of having nodes that represent objects in the world and links that represent the relations between the objects cannot be pushed too far. Problems may arise when the network databases become too large, and there is argument as to what a node really means, how the passage of time may be represented and the way to represent an idea. Current research in this area is attempting to deal with these and other issues.

3.1.2 Logical Expressions

Logic has been used by philosophers to represent knowledge since the time of the ancient Greeks. Logic is to do with the formal treatment of knowledge, and it has now developed into the application of computer programs that can reason. Here, we deal with formal logic - that of propositional logic and predicate calculus.

Propositional Logic

Propositions are statements of fact that are either true or false, and, when linked by connectives such as 'AND', 'OR' and 'NOT' are called compound statements. Propositional logic is concerned with whether these statements are true or false. The five connectives are represented as (Barr and Feigenbaum, 1981):-

\[
\begin{align*}
\text{AND} & \quad \land \quad \text{or} \quad \& \\
\text{OR} & \quad \lor \\
\text{NOT} & \quad \neg \\
\text{IMPLIES} & \quad \Rightarrow \\
\text{EQUIVALENT} & \quad \equiv
\end{align*}
\]

An example of their use is: -

\[X \land Y \text{ is TRUE and } Y \text{ is TRUE; otherwise } X \land Y \text{ is FALSE}\]
Rules of inference are also used in propositional logic. Inference rules allow the deduction of a new sentence from previously given sentences, and if true, then the new sentence will be true. This means that if the sentences \( X \) and \( X \land Y \) are true, then we can infer that \( Y \) is true.

**Predicate Calculus**

As it stands, predicate logic is not very useful for artificial intelligence work. In order to express knowledge, we need to be able to say whether propositions are true or false, and also to be able to say something about objects and the relation between objects. Predicate calculus is an extension of propositional logic, and uses predicates to write assertions which describe objects, e.g. the assertion "is_red(fire_engine)", meaning that the fire engine is red. A predicate is either true or false and more than one argument is possible, e.g. "is_smaller_than(2,3500)", meaning that 2 is smaller than 3500.

Functions are also used in order to make predicate calculus more understandable. Functions may have a fixed number of arguments and they can also be true or false, and can also return objects related to their arguments, e.g. uncle_of(mary) would return a name such as john. Another useful addition is the predicate equals. In logic, \( X \) and \( Y \) are equal if and only if they are indistinguishable under all predicates and functions. In symbolic form:

\[
X = Y \iff \text{for all predicates, } P \ P(X) = P(Y),
\]

\[
\text{and also for all functions, } F \ F(X) = F(Y).
\]

where 'iff' means 'if and only if'.
This is a form of first-order logic, and plays an important role in the artificial intelligence language Prolog.

Logic provides a method of asserting facts, taking the form of logical statements consisting of predicates and values. Harmon and King (1985) state that the use of logic in artificial intelligence is more popular in Europe and Japan, but is becoming more popular in the United States. It is a very powerful technique of representing logic.

3.1.3 Object-Attribute-Value-Triplets

Object-attribute-value-triplets are used in the expert system MYCIN. Objects can be physical or conceptual entities, attributes are properties associated with the objects, and the values describe the size, colour, shape or quantity of a particular object. This is a specialization of the semantic network approach described previously, in which only two simple relationships are used as links. The 'has-a' link is used for the object->attribute, and the 'is-a' link for the attribute->value link, e.g. a bank loan 'has-a' rate of interest, and 12% 'is-a' rate of interest.

There are two types of knowledge that can be represented in object-attribute-value-triplets. Static knowledge has generic attributes and is unchanging, and dynamic knowledge changes from case to case. Objects are ordered in a type of graph called a tree with the most valuable attribute at the root which is used as a starting point for reasoning and for obtaining information. For dynamic trees, the root is the main object with attributes and values further down the tree.

Object-attribute-value-triplets can also handle uncertainty. Certainty factors represent the confidence that we have in a fact, a piece of evidence or a conclusion. In MYCIN, certainty factors range from -1.0 to +1.0, with -1.0 indicating that the fact is false, and +1 true. More on uncertain information will be described in section 3.3.
3.1.4 Production Systems

Production systems were first proposed by Post in 1943, and an excellent review of production systems is given by Davis and King (1977). The systems used by today's artificial intelligence programs bear little relation, however, to the original formulation. The term production system is used to describe several different systems based on one general underlying idea - the notion of condition-action pairs, called production rules, or just productions.

Production systems are seen by some (Newell, 1972) not as simply a convenient paradigm for approaching psychological modelling, but rather as a methodology whose power arises out of its close similarity to the fundamental mechanisms of human cognition. In this way human problem solving behaviour can be modelled easily and successfully by a production system because it is in fact being generated by one:

"We confess to a strong premonition that the actual organization of human programs closely resembles the production systems organization...We cannot yet prove the correctness of this judgement, and we suspect that the ultimate verification may depend on this organization's proving relatively satisfactory in many different small ways, no one of them decisive."

Newell, 1972

Components

Production systems consist of three main parts:

1. Rule Base - of production rules
2. Data Structure - called the context
3. Interpreter - controls the systems activity

In the simplest form, a rule is an ordered pair of symbol
strings, with a left hand side (LHS) and a right hand side (RHS). The rule set has a predetermined total ordering. The context is simply a collection of symbols, and the interpreter operates by scanning the LHS of each rule until one is found that can be successfully matched against the database. At that point, symbols matched in the database are replaced with those found in the RHS of the rule, and scanning either continues with the next rule or begins again with the first.

**Productions**

A production rule is a statement cast in the form "if this CONDITION holds, then this ACTION is appropriate". The IF part is called the condition part and states the conditions or constraints that must be present for the production to be applicable. The THEN part, called the action part, is the appropriate action to take. A production system whose condition part is satisfied can 'fire', i.e. have its action part executed by the interpreter.

Winston (1984), states the five primitive operations that are allowed in production systems:

1. **Write** - A production can write a new item into a short term memory.

2. **Note** - A production can note by moving items from their existing place to the front.

3. **Mark** - Used to prevent a goal description from re-activating the same production over and over again.

4. **Send** - A production can request new information

5. **Receive** - A production can place a message at the front of the short-term memory.

Typically, production systems consist of many hundreds of productions in their rule bases.
The Context

The context, or data/short-term memory buffer, is the focus of attention of the production rules. It may be a simple list, a very large array, or, more typically, a medium-sized buffer with some internal structure of its own.

For systems designed to be knowledge-based experts, the context contains facts and assertions about the world. For example, the MYCIN system uses a collection of four tuples, consisting of an associative triple and a certainty factor. In the DENDRAL system, the context consists of complex graph structures which represent molecules and molecular fragments. Structures are built up by assigning numbers to each atom of a molecule and by describing chemical bonds by a pair of numbers indicating the atoms they join. A further example is the LISP70 system, which is a "token stream" approach. The context is a linear stream of tokens which are accessed in sequence. Each production is matched against the beginning of the stream, and if the rule is invoked, characters are either added, deleted or modified. This is said to be a very efficient type of system.

The context is the sole storage medium of all state variables of the system. This is very different from procedurally-oriented languages, where there is no provision for separate storage of control state information.

Interpreter

There are many different types of interpreters in existence. They are basically select-execute loops, in which a rule applicable to the state of the context is selected and executed. The action the rule performs results in a modified database, and the selection phase begins again. Since the first rule that matches the context is chosen, this cycle is often referred to as the 'recognize-act'. A complete re-evaluation of the context is made every time a rule is selected because a new rule is chosen based on the current
Production systems operate in cycles. Each cycle examines the productions in a manner specified by the interpreter to see which are appropriate and could fire. If more than one is found appropriate, a single production must be selected from among them (called the conflict set), and one is fired. This process of selecting from the conflict set is called conflict resolution. The three phases of the cycle are:

1. Matching
2. Conflict Resolution
3. Action

Conflict Resolution

As stated above, it is possible for more than one production to fire in each cycle of operation. The system needs to choose from the conflict set. This is where the basic cognitive traits such as action-sequencing, attention-focussing, interruptibility and control are realised. There are several approaches to conflict resolution, all tried and tested:

1. Specificity Ordering - Suppose the conditions of one triggering rule are a superset of the conditions of another. Use the rule with the superset on the ground that it is more specialized to the current situation.

2. Rule Ordering - Arrange rules in one long priority list. The triggered rule appearing earliest in the list has the highest priority. Others are ignored.

3. Data Ordering - Arrange all possible data items in one long priority list. The triggering rule is that of the highest priority data.

4. Size Ordering - Assign the highest priority to the triggering rule with the toughest requirements - where toughest means the longest list of constraining conditions so
that such a rule will always fire first.

5. Recency Ordering - Consider the most recently used rule to have the highest priority or consider the least recent to have the highest priority - at the designers discretion.

6. Context Limiting - Reduce the likelihood of conflict by separating the rules into groups - only some of which are active at any one time. Have a procedure that activates and deactivates the groups.

It is believed that no simple conflict resolution strategy can be completely satisfactory. Also, such strategies affect two important characteristics of production system - sensitivity, the ability to react quickly to changes, and stability, the ability to carry out relatively long sequences of actions.

Advantages and Disadvantages of Production Systems

Production systems have been used in a wide variety of problems, and are probably the most widely used of all knowledge representation techniques. Examples of applications include medical diagnosis, speech understanding, and mineral exploration. Even though these represent a large and diverse subject area, there are features of them, that are both good and bad that can be generalized:

Advantages

1. Modularity - One obvious quality of production systems is that individual productions in the rule base can be manipulated in several ways. They can be added, deleted or changed independently without altering the method and working of the inference system, or affecting any of the other rules. Rules only communicate via the context data structure, which means only the performance of the system will change. This makes the creation of the data base much easier even for very large systems, since it will be known what a proposed rule will mean
in whatever situation. This quality is also a recognized necessity of an expert system.

2. Uniformity - The knowledge used in production rules has to be by nature very uniform in structure. This is to the advantage of production systems. All the information must be encoded into the rigid structure of the productions. This means that the knowledge can be easily understood by other users, or by another part of the system, compared to the free form of semantic net or procedural representation schemes, for example.

3. Naturalness - Important kinds of knowledge are expressed very easily in production systems. By this, we mean statements about what to do in certain situations are ideally and naturally encoded into productions. Further, these kinds of statements are used most often by experts when explaining how to do their job.

Disadvantages

1. Inefficiency - Good modularity and uniformity are gained by production systems at the expense of efficiency. Problem-solving by this method results in high overheads, since the match-action sequence conveys all its information via the data-structure after every cycle. It is also difficult for production systems to take larger steps in their reasoning or to make them more responsive to predetermined situations.

2. Opacity - It is hard to follow the flow of control in solving a problem. The reasons for this are the isolation of each production, and the uniform size of productions - there is no subroutine hierarchy. Function calls and subroutines would help to make the flow of control easier.

Domains for Production Systems

The following is a list of areas which production systems can and have been used.
1. Domains in which the knowledge is diffuse as opposed to domains consisting of concise unified theory.

2. Domains in which processes can be represented as a set of independent actions as opposed to domains with dependent sub-processes.

3. Domains in which knowledge can be easily separated from the manner in which it is to be used (e.g. a classification taxonomy), as opposed to cases in which representation and control are merged.

4. If the task can be viewed as a sequence of transitions from one state to another in a problem space we can model this behaviour with production systems. Each transition can be represented by one or more production firings.

Production systems are very good at capturing certain kinds of knowledge for problem-solving. This is knowledge about what to do in a specific situation, and is held in a manageable representation scheme. The production system has advantages over declarative knowledge representation schemes, mainly in the modularity of the rules. Also, the way productions are structured is similar to the way we, as humans, would talk about how to solve certain kinds of problem.

3.1.5 Frame Systems

A frame system is a knowledge representation method which associates features with nodes representing concepts or objects. The features are described in terms of attributes (called slots), and their values.

In artificial intelligence, frames refer to the special way of representing common concepts and situations. Minsky (1975), originated the frame concept:

"A frame is a data structure for representing a stereotyped situation, like being in a certain kind of living room or
going to a child's party. Attached to each frame are several kinds of information. Some of this information is about how we use the frame. Some is about what can happen next. Some is about what to do if these expectations are not confirmed."

Frames are organized in much the same way as a semantic net. In fact both semantic nets and frames are considered to be frame-based systems. Frames are networks of nodes and relations organized in a hierarchy, where the top-most nodes represent general concepts and the lower nodes more specific instances of the concepts.

In frame systems, the concept at each node is defined by a collection of attributes (e.g. name, colour, size), and values of those attributes (e.g. smith, red, small), where attributes are called slots. Each slot can have procedures (arbitrary pieces of computer code) attached to it, which are executed when the information in the slot (the value of the attribute) is changed. Each slot can have any number of procedures attached to it. Three useful ones often used are:

1. if - added procedure - executes when new information is placed in the slot.
2. if - removed procedure - executes when information is deleted from the slot.
3. if - needed procedure - executes when information is needed from the slot, but the slot is empty.

Attached procedures can monitor the assignment of information to the node, making sure that appropriate action is taken when values change. As their structure suggests, frame systems are useful for problem domains where expectations about the form and content of the data play an important role in problem solving, such as understanding visual scenes or understanding speech.
Using a Frame System

The user types in a title which is inserted into the topic slot of the next empty node. Things then start automatically.

1. The if-needed procedure attached to the topic slot executes since a value was inserted into the slot. This procedure searches a database associated with the system to find the associated fact which is inserted as a value into the next slot of the frame.

2. The if-added procedure attached to this slot name executes since a value was just inserted. A message is needed, but a value for this message is not there.

3. The if-added procedure having looked in the next slot for this information activates the if-needed procedure attached to that slot. The if-needed procedure finds the relevant information in the database, this value is then inserted into the slot.

4. The if-added procedure attached to the slot from (2) then finds that another piece of information is needed. This could be a default value.

And so on.

Organizing Knowledge and Expectations

Frames, then provide a structure or framework, in which new data are interpreted in terms of concepts acquired through previous experiences. The organization of this knowledge facilitates expectation-driven processing, i.e. looking for things that are expected based on the context one thinks one is in.

e.g. A simple frame for the generic concept of a chair might have slots for number of legs and style of back. A frame for a particular chair has the same slots, but the contents of the
slots are more fully specified:

**CHAIR Frame**

- **Specialization-of:** FURNITURE
- **Number-of-legs:** an integer (default = 4)
- **Style-of-back:** straight, cushioned...
- **Number-of-arms:** 0, 1, or 2

**JOHN'S-CHAIR Frame**

- **Specialization-of:** CHAIR
- **Number-of-legs:** 4
- **Style-of-back:** cushioned
- **Number-of-arms:** 0

By supplying a place for knowledge, and therefore creating the possibility of missing or incomplete knowledge, the slot mechanism permits reasoning based on seeking confirmation of expectations - "filling in the slots".

### Some Frame Applications

Application of frame driven systems have been described by Bobrow et al. (1977), and by Goldstein and Roberts (1977). Bobrow describes a system called GUS (Genial Understander System) which is a frame driven dialog system. It is one of a series of experimental computer systems intended as part of a program of research on language understanding, written in Lisp. Its first role was that of a travel agent, giving details of possible flights to or from a particular place. Its components are a morphological analyzer, a syntactic analyzer, a frame reasoner and the language generator. Most of the frames in GUS are created during the process of reasoning, although some do exist in the initial database. Bobrow gives the following example of a frame for a date of the year:
An instance frame for November 14th would be:

```
[ISA DATE
  MONTH NOVEMBER
  DAY 14 ]
```

The slot labelled MONTH specifies that only a NAME can be used as a value, as with all the other slot names. Hence GUS can only interpret a standard set of type terms such as names, integer, list and string.

GUS consists of other procedures, but only the reasoning system, i.e. that of frames, is relevant here.

NUDGE (Goldstein and Roberts, 1977) is a frame system developed for the office scheduling domain, although it does have wider applications. It is written in a knowledge representation environment called FRL-O, which evolved from generalizations of the property list representation. NUDGE contains a hierarchy for practices concerning information transfer for people in various roles involved in the transfer, plans governing the transfer, and demands on time, space and personnel. The hierarchy is about 5 levels deep and includes about 100 objects. An extensive description of NUDGE is given by Goldstein and Roberts.

3.2 Knowledge Acquisition

Before knowledge can be represented for an expert system in one of the methods outlined above, the knowledge engineer must acquire the knowledge from the expert. It is recognized throughout the artificial intelligence field that knowledge
acquisition is one of the bottlenecks of expert systems development (Feigenbaum and McCorduck, 1984). The problem is how to acquire the knowledge so important for automatic problem solving in a way in which the computer eases the transfer of expertise from experts to the symbolic data structure that makes up knowledge representation. Currently, knowledge is acquired in a slow and painstaking way, generally with the knowledge engineer working with the expert. Feigenbaum and McCorduck go on to say:

"Right now (and it can't be emphasised often enough) the problem of knowledge acquisition is the critical bottleneck in AI."

Knowledge is essential to the workings of an expert system. The knowledge must be complete and free from errors and must be continually updated if the system is to be useful for some years to come. Sell (1985) describes knowledge as a "slippery concept". Indeed, philosophers have been trying to define it for thousands of years. In practice, knowledge is treated as rules, facts, reasons, heuristics, and truths that experts find useful in solving problems of a particular domain. The power of an expert system lies in its knowledge base, the rest of the system being a manipulation program in order to find the particular requirements of the problem to be solved.

3.2.1 Elicitation

Sell (1985) lists the following as important operations of elicitation:

- Extracting knowledge and making it easy to manipulate and scrutinize
- Making the knowledge explicit and giving detail to make it clear
- Recording it in symbolic form
- Verification - checking the symbolic form with the original knowledge
The process of elicitation identifies small packets of knowledge which must be organized into a form which is unified. This will depend on how the knowledge is used in the system. The order of the rules could also be important, especially if conflict is likely to arise.

Once the knowledge is obtained and organized it must be made available to the system. Knowledge bases are usually encoded into two different forms - one external and one internal. The external knowledge base is for human needs and is readable. The internal one is in code form, suitable for the machine.

3.2.2 Sources of Knowledge

There are three main sources of knowledge - namely literature, experts and examples. Additionally, there are three main bases of scientific knowledge - scientific laws, experience and models. The knowledge we require is knowledge of the domain that will help to solve the problem. Typically, knowledge that allows us to predict what will happen next or why something has happened. For this, some expression of order of the knowledge is required, and the most useful base for this is probably scientific laws. These can be obtained from either literature or experts, but more commonly from literature. However, the majority of expert systems built to date have been built in areas that are not as well defined as those based on scientific law. The laws in these areas are not codified or found in any written form, so the expert has to be quizzed by the knowledge engineer for his own personal laws. As as been pointed out before, the expert may find it difficult to verbalise his knowledge, making acquisition very difficult.

This type of acquisition has come in for some criticism (Sell, 1985), mainly because the knowledge engineer is not using the expert to his full capabilities. The expert may not be aware that rules exist, and the knowledge and heuristics that he uses is at a more subconscious level. Another approach
is to get the expert to produce and clarify examples of situations in the domain. This technique has been used to some degree of success in the tool 'Expert-Ease', in which examples are fed into the system and, through a process of induction, rules generalising those examples are produced. This area comes under the heading of machine intelligence.

3.3 Representing Knowledge

One of the features of expert systems is their ability to handle knowledge which is of an imprecise nature. This could be an inherent part of the knowledge base or of the answers given by the user, e.g. "don't know", or answers with a probability attached to them, e.g. "How certain are you that...?". This is uncertain knowledge, and the inference engine of the system must have a mechanism for dealing with uncertainty. There are a number of methods for doing this, including fuzzy logic, Bayesian logic, multi-valued logic and certainty factors. Many schemes have been tried, and all appear to work to some degree. Here, we outline Bayesian logic, fuzzy logic and certainty factors.

3.3.1 Bayesian Logic

Bayes theory has been widely used in expert systems, and is used in the mineral exploration program, PROSPECTOR and its subsequent shell, Micro-Expert. The theory is based on the conviction that for any event, irrespective of how unlikely it might be there is an a priori probability that it could be true. So, even if the event is totally untrue, it's prior probability would be zero, thus giving the basis of a calculation as if there was a probability there. In terms of probability, let \( P(H) \) be the prior probability of some hypothesis. Given an item of relevant evidence, \( E \), then \( P(H:E) \) is the following probability of the same hypothesis. By definition :-

\[
P(H:E) = \frac{P(H \& E)}{P(E)} \quad \text{and} \quad P(E:H) = \frac{P(E \& H)}{P(H)}
\]
Rearranging,

\[ P(H \mid E) = \frac{P(E \mid H) P(H)}{P(E)} \]

Detailed discussions of Bayes' Theorem in expert systems have been written by Naylor (in Forsyth, 1984) and by Townsend and Feucht (1986). A point worthy of note here is that to find the probability of a conclusion, the probabilities of the given facts must be independent of each other. It is almost impossible to establish a knowledge base in which the certainty of all the rules is independent.

3.3.2 Fuzzy Logic

Fuzzy logic was derived by Lotfi Zadeh in 1965. His basic idea was to extend classical Boolean logic into real numbers. In Boolean logic, 0 represents falsity and 1 truth. This is the case in fuzzy logic too, although fractions between 0 and 1 are used to represent partial truths. e.g.

\[ p(\text{blond hair}(X)) = 0.5 \]

indicates that the proposition that X has blond hair is 50% false and 50% true. Fuzzy logic also has notation for the AND, OR and NOT operators for combining non-integer truth values.

\[ p_1 \text{ AND } p_2 = \text{MIN}(p_1, p_2) \quad \text{(smaller)} \]

\[ p_1 \text{ OR } p_2 = \text{MAX}(p_1, p_2) \quad \text{(greater)} \]

\[ \text{NOT } p_1 = 1 - p_1 \quad \text{(inverse)} \]

A shortcoming of fuzzy logic, as indicated by Forsyth (1984), is the mapping or membership function. Suppose that a man is 35. How true is the statement that he is old? It could be 40%, 50% or 60%. It is up to the user of fuzzy logic to decide on such matters and there will be many such functions in which an arbitrary decision is required. Another failing of fuzzy
logic is its inability to weigh up disparate or possibly conflicting sources of evidence.

Fuzzy logic has found some applications in expert systems. The system Reveal is a decision support system, and overcomes some of the disadvantages by allowing the user to modify various mapping functions so as to find out if the variations are critical or not. The shell that is the successor to Micro-Expert, Savoir, also uses fuzzy logic in order to handle facts with values such as 'unlikely' or 'very probably'.

3.3.3 Certainty Factors

Certainty factors were used in the MYCIN system so that some of the failings of standard statistics could be overcome. Each rule has a certainty factor associated with it. To calculate the certainty factor of a deduced fact, the certainty factor of the fact, X, and that of the rule, Y, are combined in the relation :

\[ X + Y - XY \]

Certainty factors can then be carried through a series of deductions, and as more rules are used with success the certainty factor will increase, approaching 1, i.e. certainty. The formula above is used because some of its properties appear to reflect how an expert handles evidence.

This chapter has reviewed the 'classical' methods of representing knowledge and some of the techniques of knowledge acquisition. Not all the techniques are used in practice in this work, but it is useful to be aware of other techniques for comparison purposes.
4 Problem Solving and Search

4.1 Problem Solving

One of the most powerful parts of an expert system is in the methods used for searching a large space consisting of many alternatives. This is a characteristic feature of artificial intelligence work, which is in contrast to large numerical calculations where a well defined solution path exists, and where large amounts of data can be processed.

Problem solving in artificial intelligence involves the search for a solution through a state space by the application of operators. The state space is the possible states in the problem solution and consists of an initial state, a goal state and intermediate states. The solution path is the path consisting of all states that lead from the initial state to the goal state. Problem solving strategies that are domain independent are referred to as weak methods, whereas expert systems are considered to be strong problem solvers, since they employ domain knowledge in the solution strategy.

Basic search methods can be considered under two headings - those of direction of search, and search procedures. An important issue in the design of knowledge based systems is the type of search procedure used, i.e. the order in which the rules are scanned for triggering. The decisions involved for designing an expert system are the direction of search and the search method. Control procedures such as these are normally part of the inference engine of an expert system.

There are two other aspects that must also be considered. In starting a search through a problem space, the knowledge system must have some way of deciding a starting point. This will determine whether the search uses forward or backward chaining. The efficiency of a search can be improved using heuristics to resolve conflicts if multiple conflicts arise and for eliminating paths that are not useful.
4.2 Direction of Search

There are two kinds of search that are used in knowledge based system; forward chaining and backward chaining. The type of search that a system uses will depend on what type of problems the system is solving.

4.2.1 Forward Chaining

Forward chaining is also referred to as forward search, bottom up, data driven or antecedent driven. A system is a forward chainer if it works from an initial state of known facts to a goal state, i.e. a goal is concluded by considering the data available. All the facts are input into the system and the system works out the most appropriate hypothesis or goal state that fits the facts. The main disadvantage of this type of search is that it requires all input data as possible facts for all the conditions, and hence can be wasteful of both human time and computer time. Often, not all the facts are known or are relevant. Forward chaining is useful in situations where there are a large number of hypotheses and few input data. It is believed that the forward chainer has its best use as part of an embedded, or larger system, rather than as a conversational system. The system DENDRAL uses forward chaining to very good effect.

4.2.2 Backward Chaining

Backward chaining is also known as backward search, consequence driven, top down, goal driven or hypothesis driven and can be considered to be a validation process. The backward chainer tries to support a goal state or hypothesis by checking known facts in the context. If the facts in the context do not support the hypothesis, then the preconditions needed for the hypothesis are set up as subgoals. It is effectively a search in the state space going from the goal state to the initial state by the application of inverse operators, and is effectively a depth first search (see section 4.3.1). The system MYCIN uses backward chaining very effectively.
Searching in both directions has some attractions. In many real world problems, two way search is combined with heuristics to reduce the branching factor at each level of the search. The result could be a situation where the two solutions "pass" in the search space, because one of the heuristics has not worked as intended, and has eliminated the required path. Also worthy of note is that we are unlikely to know how many steps are involved in a solution path and thus we may not know that the passing problem is occurring. The result would be a deeper and deeper search whilst the best solution has been missed.

It is evident that the direction of search chosen for a particular problem must be appropriate to that problem. DENDRAL, which uses forward search, works forwards thus generating many possible solutions for identifying an unknown compound starting with mass-spectroscopy data. MYCIN's staring point is the symptoms and the results of tests performed on the patient. It then works backwards to generate possible causes of the infection. In both cases, the mechanism chosen is appropriate. Andow (1984) gives the effects on each system if the opposing search direction were applied. If DENDRAL used a backward search, then it would guess one of the many possible compounds and check to see if the spectrogram was consistent. If MYCIN used forward chaining, it would choose one of the many possible infections and compare it with the symptoms. Both would be wasteful of time and resources.

Thus, the choice of search direction is problem dependent. Andow (1984) states that in practice most expert systems of repute use forward chaining and he questions whether this is a random choice, research workers bias or whether this reflects the type of problem that we regard as "expert".

We should, in this project consider how experts handle design problems and decide which procedure is best - whether it is either forward or backward chaining, a combination of both, and whether we can solve all design problems using these methods.
4.3 Search Mechanisms

Search problems are one of the main areas of artificial intelligence work, and deal with situations in which one choice leads to another. There are many types of search procedure, from simple, basic methods to more complicated methods such as problem reduction and theorem proving. Search procedures are used to find paths from starting positions to goal positions and can be used under forward or backward chaining. If the search path is thought of as a tree, then the points in the tree are called nodes and the connections between the nodes are called branches. One node is the ancestor of another or the descendant if there is a chain of branches between the two.

We can demonstrate the principles and concepts of strategies for exploring alternatives with the following example given by Bratko (1986). Consider the diagram in Figure 4.1. The problem is to find a plan for rearranging a stack of blocks. The rules of the game are:

- Move only one block at a time
- Move when there is no block on top
- A block can be put down or put on top of another block

![Figure 4.1: Transformation of CAB to ABC by Search](image)

FIGURE 4.1 Transformation of CAB to ABC by Search
A sequence of moves must be established in order to accomplish the given transformation. It is a choice of alternatives. Our first move is to put block C down, then the choice is:-

Put A down, or

Put A on C, or

Put C on A.

This illustrates two types of concept - that of problem situations and that of legal actions transforming the problem situation into other situations. These lead on to form a graph, or a state space. The nodes of the graph correspond to problem situations, and the arcs correspond to legal transitions between states. The problem of finding a solution plan is equivalent to finding a path between the initial situation and some specified final situation (the goal node).

In this section, we deal with simple procedures that have been used extensively in artificial intelligence. They include depth-first search, together with hill-climbing, breadth-first search, and best-first search. These are used to find paths from starting positions to goal positions when the length of the discovered paths is not important. Procedures which find the shortest path are more complex and include the British Museum procedure, branch and bound and discrete dynamic programming. Useful work has been published in this field by Winston (1984) and Hayes-Roth (1983) and in the specific area of applications in engineering design, by Maher, et.al. (1984).

4.3.1 Depth-First Search

This method is one which dives deeply into the search tree. It works on the assumption that one path is as good as any other. Depth-first search picks an alternative at every node in the search space and works forward from that alternative. Other alternatives at the same level are ignored provi-
FIGURE 4.2 DEPTH FIRST SEARCH
ding that there is a possibility of reaching the goal state using the original choice.

Consider Figure 4.2. The first action would be to go to the bottom of the tree along the left-most branches (assuming a left-to-right order in choosing alternatives). This leads to terminal node C without encountering a goal node G. So, the next step is to backup to the nearest ancestor node with an unexplored alternative. This is B, which leads to eventual success at G, despite another dead-end goal at D.

4.3.2 Hill-Climbing

One disadvantage of depth-first search is that it can go off on the wrong trail and waste a lot of time, e.g. if node C had been the gateway to a vast subnetwork then depth-first search would have exhaustively searched this before finding the right solution. This performance can be improved by ordering the choices so that the most promising is explored first. This is called hill-climbing, and involves quality measurements that turn depth-first search into a more efficient procedure. A heuristic measure, or objective function is used to order the choices. This is shown in Figure 4.3. Search efficiency is improved spectacularly by ordering the choices so that the most promising are explored first. The procedure is as described above, but each node has a number assigned to it which indicates straight line distances to the goal node. The better the heuristic measure is, the better the hill-climbing will be.

Winston gives some examples of hill-climbing in parameter optimization:

The picture on a television set has deteriorated over a period of time. The tuning, colour, tint and brightness controls must be adjusted for a better picture. In this example there are various knobs, each of which interacts with the others to determine the overall picture quality. Winston also describes the various problems associated with parameter opti-
FIGURE 4.3 HILL CLIMBING
4.3.3 Breadth-first Search

As we have seen in depth-first search, the tree is traversed downward until success or failure occur. Instead of moving downward through the levels, all the alternative clauses are tried first in breadth-first search. The result is that control flow scans across each level, moving to the next lower level only after trying all the nodes at that level, i.e., all alternative solutions are tried concurrently rather than one at a time. Breadth-first search pushes uniformly into the tree, and eliminates back-tracking. Breadth-first search places a substantially greater demand on memory usage than depth-first search because it carries so many parallel paths.

Breadth-first search is shown in Figure 4.4. This shows that downward motion proceeds level by level until a goal is reached. Node D is checked after A. The procedure then moves on, level by level, discovering the goal node G on the fourth level down from the root level.

4.3.4 Best-first Search

Best-first search is a modified form of breadth-first search. Here, all solutions found by breadth-first are evaluated and ordered so that more "promising" ones are considered first. This can only work if a reliable evaluator is available. The least likely paths are either explored later in the search or are discarded completely if exhaustive search is not needed.

Best-first search expands the best partial path in the search space. Motion towards a goal is from the best open node so far, no matter where it is in the partially developed tree. The path found by best-first search is likely to be shorter than with other search methods because the search always moves forwards from the node that seems closest to the goal node. So, best-first search works by expanding the best partial
Figure 4.4 BREADTH FIRST SEARCH
path. Winston (1984) draws the following analogy. Suppose a team of mountaineers are seeking the highest point in a mountain range. They maintain radio contact, move the highest subteam forward at all times, and divide subteams into sub-subteams at path junctions.

This method combines the advantages of depth-first and breadth-first in that easy solutions are not missed and the method can get deep into the search space quite quickly.

4.3.5 Branch-and-Bound Search

One way to find optimal paths with less work is by using branch-and-bound search. During a search there are many incomplete paths contending for further consideration. The shortest one is extended one level, creating as many new incomplete paths as there are branches. These new paths are then considered along with the remaining old ones, and again, the shortest is extended. This repeats until the destination is reached along some path. Since the shortest path was always chosen for extension, the path first reaching the destination is certain to be optimal. Consider figure 4.5. Suppose an optimal solution is desired for the net shown in Figure 4.5a. Looking at the first level, in Figure 4.5b, the distance from S to node A is clearly less than the distance to B. Following A to the destination at the next level reveals that the total path length is 4, as shown in Figure 4.5c. This means that there is no point in calculating the path length for the alternative path through node B since at B the incomplete path's length is already 5 and hence longer than the path for the known solution through A.

4.3.6 Generate and Test

The state space search may sometimes be formulated as generate and test. Here, the search is divided into two parts, that of a generator of possible solutions, and that of a tester, which prunes solutions that fail to meet some constraints. A generator is complete if it is capable of genera-
Figure 4.5 Branch and Bound Search

In branch-and-bound search, the node expanded is the one at the end of the shortest path leading to an open node. Expansion continues until there is a path reaching the goal that is of a length equal or shorter than all incomplete paths terminating at open nodes. A sample net is shown in (a), along with partially developed search trees in (b) and (c). The numbers beneath the nodes in the trees are accumulated distances. In (b), node A might just as well be expanded, for even if a satisfactory path through B is found, there may be a shorter one through A. In (c), however, it makes no sense to expand node B, because there is an incomplete path to the goal that is shorter than the path ending at B.
ting all possible solutions. It is nonredundant if throughout the generation, it produces each solution only once.

The distribution of knowledge between the generator and the tester is important, since putting as much knowledge as possible in the generator can lead to a more efficient search path. Hierarchical generate and test does this, and allows pruning of possible solutions that are only partially specified. When a partial description is pruned, an entire class of solutions corresponding to the description is eliminated from the generation process. Pruning rules are applied early in the generation process.

This strategy is only applicable if appropriate tests can be formulated. Typically, in design there is no unique solution, so there is therefore no absolute test for a solution. It may be applicable in the preliminary design phase if the testing is changed to a ranking stage, so that the relative value of possible solutions is determined.

4.3.7 Heuristic Search

The amount of time and storage space available for a search may place a restriction on the search mechanism used. Both depth-first and breadth-first searches are exhaustive of these facilities (although hill-climbing does help in the former), because although a solution may eventually be reached, too many nodes may have been visited to make it practical, particularly with large problems.

Often, it is possible to extract domain specific knowledge in order to guide the search and thus reduce the demands on time and space. This knowledge is referred to as heuristic information and search procedures using it are called heuristic search methods. The advantage is that as soon as a satisfactory solution is reached, the search will stop.

One method of executing a heuristic search is in a "best-first" order, in which a domain dependent function is used to
determine which branch of the search tree to expand. An example has been suggested by Nilsson (1980) for the 'eight-puzzle'. This contains eight numbered tiles with space for a ninth. The object is to move the tiles up, down, left or right (i.e. 'operators') until the goal state is reached which has the numbers in the correct numeric order. Nilsson suggests the following evaluation function for this puzzle:

\[ f(n) = d(n) + w(n) \]

in which \( d(n) \) is the depth of the node \( n \) in the tree and \( w(n) \) counts the number of misplaced tiles. The function is intended to give an estimate of the computational effort required in pursuing a path. Nodes are tried in increasing order of their \( f \) values. Generally, evaluation functions are difficult to characterize, as described by Hayes-Roth (1983):

"To be useful, evaluation functions must characterize the solution space adequately, which generally requires a substantial amount of knowledge..."

Evaluation functions are said to behave well if they reliably and monotonically indicate an optimal path to a goal. In many real problems, well-behaved evaluation functions are elusive. Often, one must seem to move away from a goal in order to achieve it.

Important aspects of expert problem solving include the direction and the method of search, both being highly problem dependent. The direction of search will generally be dependent upon the number of possibilities that need to be considered. The search procedure will depend on whether all solutions are required, the effects of missing solutions and the availability of an intermediate evaluator.

4.4 Backtracking

Backtracking is also an important aspect of problem solving in artificial intelligence. The need for backtracking
cannot be ignored in engineering design, as it is unusual that the first solution considered will satisfy all the applicable conditions. Some kind of backtracking must be incorporated in a knowledge based expert system. The degree of backtracking will depend on the particular application. Backtracking is conceptually simple, but provides a very powerful search mechanism that can be useful in expert system problems.

Backtracking is a type of problem reduction that is applicable to problems that can be subdivided into a tree of fixed subproblems. In a number of practical problems, however, it may not be possible to decompose problems into a fixed set of subproblems. A number of alternate subproblems may exist. In backtracking, the problem solver backs up to other nodes, at the same level as the starting node, if no solution is found along the current path. So, backtracking consists of reviewing what has been done and attempting to resatisfy the goals by finding alternative ways to satisfy them.

Backtracking is an inherent part of the programming language Prolog. In other languages, the user would need to construct the loops himself in order to achieve similar results. In Prolog, backtracking can be initiated automatically, or manually if the user is not satisfied with the solution that Prolog has found. It can also be controlled, by using the 'cut-fail' combination, which is important, as uncontrolled backtracking could lead to unnecessary usage of computer time.
5 Languages, Environments and Shells

This chapter describes some of the facilities available for building expert systems. They come in three forms - languages, environments and expert system shells. Examples of each are given in this chapter. In the languages section, we have concentrated on the 'official' languages of artificial intelligence, i.e. Prolog and Lisp. It should not be forgotten, however, that the more conventional languages of computing can also be used for building knowledge-based systems and expert systems. Such languages include BASIC, FORTRAN and PASCAL, with PASCAL probably the one which is used more often. Some of the early expert systems were written in such conventional languages, and work by James (1984) has shown artificial intelligence applications in BASIC.

5.1 Languages

5.1.1 Prolog

Prolog is very much the European language of artificial intelligence, as opposed to Lisp, which is that of the United States. Prolog is also the language that the Japanese have chosen for their ICOT (Institute for New Generation Computer Technology) project.

Schlobolm (1984) states that Prolog is gaining in popularity over Lisp for symbolic computation, and artificial intelligence programming. The argument over which is better rages on, and takes up a disproportionate amount of space in computing journals. In this project, we have concentrated on Prolog, since the properties of Prolog appear to be more suited to the kinds of problem-solving being tackled.

Prolog was developed in around 1970 by Alain Colmerauer in Marseilles, France (1983), and was brought to Edinburgh University where a DEC10 compiler/interpreter was developed. Edinburgh was instrumental in spreading Prolog to other institutions and now there are various implementations available.
It has since been used in such areas as natural language understanding, expert systems and symbolic mathematics.

Prolog stands for PROgramming in LOGic and uses a series of logic constructs in order to solve problems and can be described as a descriptive language as opposed to a prescriptive language such as BASIC, FORTRAN or PASCAL. These, so-called 'conventional' languages use algorithms in order to solve tiresome calculations that would take humans many hours to solve. Prolog solves problems involving objects and relationships using facts and rules. As described by Clocksin and Mellish (1984), Prolog is about:

- declaring facts about objects and their relationships
- defining some rules about objects and their relationships
- asking questions about objects and their relationships

Clocksin and Mellish describe Prolog as a practical and efficient implementation of many aspects of "intelligent" program execution. Features include recursion, back-tracking and built-in predicates.

Facts, Rules and Questions

It is possible to declare facts about the world in Prolog by defining predicates, e.g.

lives(house, john).

Meaning that 'john lives in a house'. Other examples could be:

hydrocarbon(propane) 'propane is a hydrocarbon'
atomic_number(gold, 79) 'atomic number of gold is 79'

A good example which shows questions, rules and facts is that of a typical family. e.g.
One rule would be:

\[
X \text{ is the sister of } Y \text{ if :-} \\
\text{X is female and} \\
\text{X has mother } M \text{ and father } F \text{ and} \\
\text{Y has the same mother and father as } X
\]

In Prolog, this is written:

\[
sister(X,Y) :- \\
female(X), \\
parents(X,M,F), \\
parents(Y,M,F).
\]

By declaring a knowledge base of facts concerning a family, it is possible to question the system about who the sister of someone is and whom the parents are. Note that the variables M and F are not matched, which means they are uninstatiated variables. They will then match anything when it is necessary to satisfy the goal parents(X,M,F).

We will define the following facts:

\[
female(helen). \\
female(clare). \\
parents(helen,jane,john). \\
parents(clare,jane,john).
\]

Suppose that we now ask the following question:

\[
?-sister(helen,clare).
\]

Here, X will become instantiated to helen, and Y to clare. As a result, Prolog attempts to satisfy the goal 'is female(clare)?'. This is true from the list of facts, so the goal is successful. Next, Prolog searches for the goal parents(helen,M,F), where M and F will match any arguments since they are uninstantiated. A fact that fulfills the requi-
rements is parents(helen, john, clare) and therefore the goal succeeds, and M is instantiated to jane and F to john. The new goal is parents(clare, jane, john) which is successful, and hence the goal sister(helen, clare) is true.

Lists

One of the main features of Prolog is its ability to handle lists. A list is a sequence of items that can be written in Prolog as :-

[peter, football, george, cricket].

Lists can be represented as trees which are combined with the use of a functor, e.g.

```
date
     /  \
    28   Nov
         /   \n        /     1986
```

This is written in Prolog as :-

```
date(28, Nov, 1986)
```

The first element of a list is called the head of the list, and the remaining parts the tail of the list. So, in the first example, 'peter' is the head of the list, and the tail is :-

[football, george, cricket].

72
This is a binary tree, and in Prolog, lists are handled as a special case of binary trees. The [ ] notation signifies the empty list. It is possible to carry out operations on lists. These are:

1. Checking whether an object is an element of a list, i.e. membership.

2. Joining together two lists, i.e. concatenation, to obtain a third list.

3. Adding/deleting to and from lists.

Here, we will describe the membership clause. It can be written in two clauses - one a fact, the other a rule:

\[
\begin{align*}
\text{member}(X,[X|\text{Tail}]) & . \\
\text{member}(X,[\text{Head}|\text{Tail}]) & : - \\
& \quad \text{member}(X,\text{Tail}).
\end{align*}
\]

Which states that, \( X \) is a member of the list if \( X \) is the head
of the list or X is a member of the tail of the list.

The other main elements of Prolog are matching, backtracking and arithmetic. Backtracking is possibly the most useful. Prolog, in its search for a solution will go through each possible path. If a goal cannot be satisfied, backtracking will be initiated. This consists of reviewing what has been done and attempting to re-satisfy the goals by finding an alternative way to satisfy them. Furthermore, if the user is not content with an answer, backtracking can be initiated by typing a semicolon when Prolog informs the user of a solution. Uncontrolled backtracking can be inefficient, so the 'cut' is used to control and prevent backtracking were necessary.

This section is only an introduction to Prolog. Examples of its use can be found in later chapters, and more detailed accounts can be found in Clocksin and Mellish (1984), Bratko (1986), and Coehlo et al. (1980).

5.1.2 Lisp

The other main language of artificial intelligence is Lisp, which takes its name from LISt Processing. It is a much more established language than Prolog, having been around since the early 1960s. It is similar to Prolog in that it models some kind of human cognition, but Lisp is primarily concerned with symbols in the form of lists, and their manipulation. Like Prolog it is highly interactive with the user, in that it reads what is typed in, evaluates it and prints the result to the terminal.

The aim of this section is to give a basic introduction to Lisp. More comprehensive discussions can be found in O'Shea and Eisenstadt (1984), Winston and Horn (1984), and Hasemer (1984).

Terminology

Any set of symbols which are legal to Lisp is called a
Lisp s-expression, which stands for a Lisp symbolic expression. Also used are atoms and expressions, an atom being a single word, and an expression being any set of legal symbols including two or more brackets. e.g.

```
alan
(jump)
(+ 2 7)
```

To the last expression, Lisp would respond with 9. The '+' sign is called a procedure and is the basic entity which specifies how something is to be done. It can be defined by the user or produced by Lisp itself. Procedures provided solely by Lisp are called primitives.

Suppose we need to remember that certain children are friends of someone called Tom. To identify the group, we can use the word FRIENDS. If Alan, John and Jane are friends of Tom, Lisp can remember this by the following:

```
(SETQ FRIENDS '(ALAN, JOHN, JANE))
```

SETQ binds (ALAN, JOHN, JANE) with FRIENDS.

If we now type

```
FRIENDS
```

Lisp responds with

```
(ALAN, JOHN, JANE)
```

We could also have a list of enemies

```
(SETQ ENEMIES '(NIGEL, PETER, ROBERT)
```

If NIGEL then becomes a friend, we can change this by:
(SETQ ENEMIES (REMOVE 'NIGEL ENEMIES))
(SETQ FRIENDS (CONS 'NIGEL FRIENDS))

The expressions REMOVE and CONS are Lisp for addition and subtraction of expressions.

Thus, we now have :

ENEMIES
(PETER, ROBERT)
FRIENDS
(NIGEL, ALAN, JOHN, JANE)

In Lisp, we can also create functions, e.g.

(DEFUN NEWFRIEND (NAME))
(SETQ ENEMIES (REMOVE NAME ENEMIES))
(SETQ FRIENDS (CONS NAME FRIENDS)))

So that the previous NEWFRIEND status of NIGEL can be effected more simply by typing :-

(NEWFRIEND 'NIGEL)

List Manipulation

Suppose we have an expression (sports cars are fast). We may like to cut off the first part of the element giving (cars are fast), or it may be necessary to add on a new first element, such as (small sports cars are fast). In order to manipulate expressions in this way, we use the primitives CAR, CDR, APPEND, LIST and CONS.

CAR

The first element of the list given as its argument is returned by using CAR :-

(CAR '(SPORTS CARS ARE FAST))
returns

SPORTS

Equally,

(CAR '(ABC))

returns

A

and

(CAR '((AB) C))

returns

(AB)

CDR

This primitive returns all but the first element of a list:

(CDR '(SPORTS CARS ARE NICE))

returns

(CARS ARE NICE)

Equally,

(CDR '(ABC))

returns

(BC)

and

(CDR '((AB) C))

returns

(C)

CDR will always return a list - unlike CAR. The diagram in Figure 5.2 illustrates this.

CAR and CDR may be used together. To pick out the second
element of a list, CDR is used first, then CAR is used. The following might seem plausible:

\[(\text{CAR} (\text{CDR} (A B C)))\]

However, Lisp does not know where the specification of what to do leaves off and where the data to be manipulated begins. Lisp assumes that A is some sort of procedure. We get over this problem by using an evaluation inhibiting signal in the form of a single quote character, Hence

\[(\text{CAR} (\text{CDR} '(A B C)))\]

returns B.

When complicated procedures are involved, and it is necessary to dig out some item from deep inside an expression, composite primitives can be used, e.g.

\[(\text{CADR} '(ABC)) \quad ((\text{CAR} (\text{CDR} '(A B C)))\]

Care is needed in the use of A's and D's in the composite primitives. The order of appearance is the inverse of the order of application.

![FIGURE 5.2 CAR and CDR operation](image-url)
Adding to Lists

The primitives APPEND, LIST and CONS can be used to construct lists, e.g.

\[(\text{SETQ } L \ 'A B)\]
returns
\[(A B)\]

\[(\text{APPEND } L \ L)\]
returns
\[(A B A B)\]

LIST makes a list out of arguments and each argument becomes an element of a new list, e.g.

\[(\text{LIST } L \ L)\]
returns
\[
((A B) \ (A B))
\]

and CONS (standing for CONStructor) takes a list and inserts a new first element, e.g.

\[(\text{CAR } (\text{CONS } 'A \ '(B C)))\]
returns
\[A\]

and

\[(\text{CDR } (\text{CONS } 'A \ '(B C)))\]
returns
\[(B C)\]

Other significant primitives are :-

LENGTH - counts the number of top level elements in a list
REVERSE - turns the top level of a list around

LAST - returns a list containing only the last element of an argument

Recursion and Iteration

There are two ways in programming of doing things repeatedly, one is recursion, and the other iteration. Recursion and iteration are examples of control structures. Recursion can be implemented by having a line of Lisp code which makes things happen again inside itself. Executing, say, SHORTEN, involves also executing an inner copy of the function, within which there occurs another copy and so on. Recursion does not involve executing the same copy of the function more than once. An example of recursion can be seen with the following example :-

```
(DEFUN SHORTEN (1)
  (COND ((NULL 1) NIL)
        (T (PRINT 1)
           (SHORTEN (CDR 1)))))
```

This shortens a list by one until the list is empty. The second line says 'if the list is empty do nothing'. When the third line is reached, a new version of the function is set up with different input data to work with. When the evaluator reaches the same point in the new shorten, the same happens again. This continues until the list is empty, and COND returns nil.

An iterative function contains a loop and, unlike recursion, only one copy of the function is ever executed, i.e. iteration occurs all at one level, whereas recursion goes progressively deeper. Iteration involves the following steps :-

1. Bind some variables
2. Test the variables to see if the exit condition applies.
If so, exit, if not goto (3)
3. Change the values of the variables
4. Goto (2)

The above explanation is only a brief introduction to Lisp. The works mentioned previously should be consulted for a more fuller discussion of Lisp. The next section deals with some applications of Lisp.

Applications of Lisp

Human thinking involves a small amount of reasoning using a large amount of knowledge. Knowledge representation is, therefore, important - using the vocabulary of symbols and conventions for arranging them. Lisp has been used extensively in representation areas. Lisp has also been used for speech and vision understanding. It is very difficult to understand how people hear and see. Progress is being made in this area using Lisp, despite the fact that much arithmetic is involved. Finally, Lisp is very popular for symbol manipulation. This is because Lisp is highly interactive and it has become very powerful for writing large programs.

5.2 Environments

Environments for building expert systems are different from shells and languages in that they are more versatile and can be used for many different applications. The only true environment is that of POPLOG, although more problem specific environments exist such as OPS5 (for production systems) are in existence. Here we shall concentrate on POPLOG.

5.2.1 POPLOG

POPLOG is an integrated, interactive multi-language environment for development of all kinds of artificial intelligence programming. It was developed by the Cognitive Studies Programme at Sussex University, and is a combination of the languages POP-11, Prolog and Lisp. The core language is POP-
11, which is similar in power to Lisp. POPLOG contains incremental compilers for Prolog and Lisp. The Prolog and POP-11 are integrated in POPLOG and can call Prolog procedures and vice versa. Also, the same screen editor, VED, can simultaneously be used to manipulate both sorts of files, taking appropriate default actions. Thus, in a complex design it is possible to implement modules in whichever language is more suitable. Combining a conventional artificial intelligence language with a logic programming language allows programs to have the best of both worlds. The editor also allows for mixing different languages such as PASCAL, FORTRAN or C. These can be linked dynamically to POPLOG. A useful account of POPLOG has been written by Sloman, Hardy, and Gibson (1983).

5.3 Expert System Shells

Along with languages and environments, the other way of building expert systems, and probably one of the easiest, is to use an expert system shell. Shells are expert systems without the knowledge. They contain the control mechanism in which the knowledge is manipulated, and structure in which to hold the knowledge. All the user has to do is provide the system with the expert's knowledge in the form of rules. There are an increasing number of shells available today, with many different ways of representing knowledge and controlling the deduction process. It is critical that the user picks the right shell for the job. In this section, we describe some of the better known shells, and also shells that have had relevance in this project.

5.3.1 EMYCIN

EMYCIN, as its name implies, derives directly from the expert system, MYCIN, and stands for "Empty MYCIN". It is appropriate therefore for developing a consultation system, which can ask for data about a situation and it can give an interpretation. Hayes-Roth et.al. (1983) suggest that EMYCIN is good for fault diagnosis, where there are many input measurements. The MYCIN inference engine is used,
and the rules must be represented in the MYCIN rule language. This is usually in the form of production rules which can also cope with certainty factors. As would be expected, the control mechanism is the same as that of MYCIN, i.e. backward chaining. It can also give explanations and trace and has debugging facilities.

5.3.2 Micro-Expert/Savoir

Micro-Expert is a shell based on the expert system PROSPECTOR. It is written in PASCAL, and is available on many types of machine from microcomputers through to mainframes. Cox (1984) gives a useful account of the development of Micro-Expert. The user of the shell does not use PASCAL directly, but the Micro-Expert advice language. The knowledge is formulated in the form of a tree with the goal at the top of the tree. At the bottom level of the tree are questions that will be asked during a consultation. Rules are in between these two levels, which are needed to prove the goal given the user responses. When the model is formulated in the tree format, it has to be coded into the advice language. The advice language has its own compiler, which creates a "rules file" and a "text file".

Micro-Expert has other features including the input of numbers, use of Bayesian type calculations, some arithmetic capability and the ability to expand on questions asked of the user. The system has found extensive use for many applications, and is a convenient and rapid way of setting up an expert system.

A direct descendant of the Micro-Expert shell is Savoir, which was produced by the same company as Micro-Expert. It is again written in PASCAL and the knowledge base is written in the Savoir language. Again, it has its own compiler, which is then used by the run-time system. Savoir has many improvements on the original Micro-Expert shell, notably the facility for explicit control of the reasoning process via "actions", which Savoir call "demons".
Since Micro-Expert has been used in this work, a fuller explanation of the system is given in section 6.4.1.

5.3.3 Expert

Expert is an example of a rule-based language as opposed to a true shell. It is a skeletal knowledge engineering language for rule-based representation. Its main characteristics include a forward chaining control scheme which can be used for diagnosis or classification type problems, a method for dealing with uncertainty and an efficient and portable code. Support for the user comes in the form of user interface facilities such as explanation, acquisition and consistency checking. Expert is implemented in FORTRAN and can operate on DEC and IBM computers. Its main applications have been in medicine.

5.3.4 ESP/ADVISOR

ESP/ADVISOR is produced by Expert Systems International of Oxford. Its basic function is the controlled output of text. It is not a shell in the true sense, rather a text animation system, in which expertise already recorded in regulations or instructions is presented as a knowledge base. The knowledge engineer creates a knowledge base source file using an editor such as WordStar. The file contains the domain knowledge written in the ESP/ADVISOR knowledge representation language. The program is checked and a Prolog knowledge base file is produced. The "ESP" program is run to perform the consultation. A menu of available knowledge bases is displayed for the user to choose from. The Prolog knowledge base file is accessed, questions are asked of the user, inferences are made and conclusions drawn accordingly. Propositional logic is used, meaning that the system is entirely deterministic, all conditions being evaluated to true or false. The line of reasoning for a consultation starts at the top of the knowledge base and proceeds down attempting to satisfy conditions as it finds them. The system chains backwards through the condition's antecedent rules to parameters whose values are known.
or can be established by asking the user a question. Applications of ESP/ADVISOR have included the regulations and procedures an employer has to follow when taking on a new employer (P.A.Y.E.).

5.3.5 Expert-Ease

Harmon and King (1985) give a useful account of Expert-Ease. It is sold by Intelligent Terminals Ltd., and has been used extensively in this project. It is an example driven system, in which the knowledge is represented in the form of examples and a one-rule decision tree algorithm. It was the first expert system building tool for a Personal Computer, and therefore attracted much attention on its introduction in 1983. The knowledge is entered by the user in the form of attributes and values which represent cases in the domain. The attributes are important properties of the domain which represent it thoroughly. The attributes are parameters which have values of specific cases. With each example is a class value, which distinguishes the example from others. From the examples, Expert-Ease is able to establish the general case for the domain, and give a rule which gives a class value for any case. Consider, for example, the case where the decision is to fit an emergency isolation valve to a piece of process plant. These are the classes, and known cases are fed as examples to Expert-Ease, giving values to such attributes as material, equipment, temperature etc. After induction, the user can query the system to see if the item he is designing needs an emergency isolation valve or not.

Expert-Ease uses Quinlan's ID3 algorithm in order to induce rules (Quinlan, 1979) in which the most discriminatory attribute is established in order to define a class value (see section 6.2.2). The 'intelligence' of Expert-Ease is in its ability to rearrange attributes in the decision tree in such a way that the questions sub-divide the tree in the most efficient way. Expert-Ease can be used by experts without the mediation of a knowledge engineer.
5.3.6 EX-TRAN7

Ex-Tran7 is a FORTRAN77-based expert system builder. There are two main components, the "Driver" which is the inference engine and "ACL-TRAN" which is a rule induction system based on Quinlan's ID3 algorithm. ACL-TRAN is an interactive tool for entering examples of decisions from the domain, and inducing rules from them, which can be output in FORTRAN code. Rules can be entered directly if required. The Driver runs the expert system consultation session. It uses an overall controlling script, the "problem text file", to determine the organization of the complete knowledge base, and supply natural language text for the question and answers. The expert system makes decisions by accessing the rules induced by ACL-TRAN. EX-TRAN7's workings are similar in nature to that of Expert-Ease, in that the rules are induced in the same way, and are in the form of trees.

These last two expert system tools have also been used in this work and receive further explanation in Chapter 6.
6 Problem Solving Tools Used

6.1 Introduction

This chapter contains the background theory of the problem solving tools used in this work. They include the rule induction technique used by Expert-Ease and Ex-Tran, a working example of a forward chaining production system written in Prolog, and the theory of Micro-Expert. These systems were introduced in Chapter 5, but here we go into more detail, and explain how they work. The production system is based on an example given by Winston (1984), for bagging groceries in the correct order, and is called 'BAGGER'.

6.2 Classifiers

Classification is an area of artificial intelligence which comes under the general heading of machine intelligence, and deals with systems which learn from past experiences. In this work, we are interested in an algorithm called Iterative Dichotomiser Three (ID3), developed by Quinlan (1979). ID3 is a rule induction method, and its background and theory are discussed in this section. Rule induction methodologies are essentially systems which can be applied to data from a number of classes of chemicals, products, situations etc. and which induce rules about the domain for differentiating between the classes.

Machine learning is the general term applied to computer systems which over time automatically improve their performance as a result of experience. The learning process consists of the ability to acquire new declarative knowledge easily. The computer must then organise this knowledge so that the program can induce, i.e. go from the particular to the general. Induction is seen as one of the key goals for artificial intelligence researchers, and is one which is receiving more attention as a research topic in its own right.

The learning process, or algorithm, attempts to achieve one
or more of the following:-

1. Allow a wide range of problems to be tackled
2. Find accurate solutions
3. Give answers at less cost
4. Simplify codified knowledge

Machine learning can potentially be applied to many domains, but recent work (Forsyth, 1984) has shown that the best results have been obtained from classification systems.

6.2.1 Induction

As mentioned above, induction is the process of going from the particular to the general, with emphasis on the use of examples. The examples are used to infer the structure and explanation of the problem. The relevance of induction to expert systems lies in the way that the knowledge is acquired. At present, most expert systems are built by using the knowledge obtained from experts after many hours of discussion and interviewing. The acquisition of domain specific knowledge is one of the main bottlenecks in the building of intelligent knowledge based systems. The reason for this is that the expert is being called upon to perform a task that he is not used to, and as a result, his complete knowledge set may suffer from a lack of detail, particularly heuristics.

Induction provides us with a technique to assist the knowledge engineer. The expert should still be consulted, since he can come up with the basic concepts, new ways of viewing objects in the domain, and provide the rules of thumb. This kind of knowledge, in the form of examples, can then be passed to an induction based algorithm.

6.2.2 ID3

The knowledge discovered by ID3 is in the form of decision trees for differentiating objects of one class from another. The basic algorithm on which ID3 is built is related
to work done by Hunt et. al. in the mid-1950s, and is called the Concept Learning System (Hunt et. al. 1966). The nature of Hunt's work was to find something out about concept learning, with the emphasis on what an intelligent device must be able to do, i.e. to classify. The idea was that ID3 would develop classification rules, given a set of examples. This then gives a means of being able to identify examples of the concept, which is essentially what a classification rule is.

ID3 takes examples of a known class (the example set) described in terms of a fixed collection of properties called attributes. In a specific example, an attribute has a value assigned to it. To illustrate this, we will use an example quoted by Quinlan (1982). We can describe a person in terms of the attributes height, colour of eyes, and colour of hair. Height could have values short or tall, eyes could have values blue or brown, and hair values dark, blond or red.

Typically, an object could be described as:

\[
\text{Height} = \text{tall}, \quad \text{Hair} = \text{blond}, \quad \text{Eyes} = \text{brown}
\]

With each of these examples is associated a known class, such as 'plus' or 'minus'. ID3 creates the decision tree which maps each instance to its correct class. An object is classified by starting at the root of the decision tree, finding the value of the given attribute, taking the branch of that value, and continuing in the same fashion until a leaf is reached.

e.g. If we had some more examples of the above instances, in a collection of C objects:

\[
C =
\begin{align*}
\text{Short, Blond, Blue} & + & \text{Short, Dark, Blue} & - \\
\text{Tall, Blond, Brown} & - & \text{Tall, Dark, Blue} & - \\
\text{Tall, Red, Blue} & + & \text{Tall, Blond, Blue} & + \\
\text{Tall, Dark, Brown} & - & \text{Tall, Blond, Blue} & - \\
\text{Short, Blond, Brown} & - & \text{Short, Blond, Brown} & -
\end{align*}
\]
These examples can be represented in a decision tree as shown in Figure 6.1. As can be seen, it is necessary only to know the value of the hair and eyes attribute in order to classify an object. This type of rule-former will always work, so long as there are not two examples which are the same, but classified differently. To make the algorithm more realistic, i.e. so that we can classify objects which were not used in its construction, we need some way of selecting the most useful attribute to form the root of the decision tree, particularly if there are many attributes. This is where the work by Quinlan (1979) differed from that by Hunt. Quinlan used an information theoretic approach, the theory of which can be found in standard texts, e.g. Raisbeck (1963). The aim is to reduce the expected number of tests needed to classify an object, by finding the attribute with the greatest discriminatory power. Each example as defined by the most discriminatory variable is partitioned by the next most discriminatory and so on until the leaf is reached.

As stated above the theory is used in order to find out the value of some message containing information. The next section outlines this theory.

If we flip a coin, there are two possible outcomes. The information associated with such an experiment will be defined as a measurement of information - 'one bit'. Then, the information associated with 'n' equally likely outcomes is precisely:

$$\log_2 n \text{ bits}$$

If we test this for an experiment whose outcome is certain, then the information associated is:

let \( H = \log 1 = 0 \)

i.e., the outcome is a foregone conclusion.

If there are 8 equally likely outcomes, then
Figure 6.1 Decision Tree for ID3 Example
\[ H = \log 8 = 3 \]

i.e., there is 3 times as much information that can be gained as that associated with flipping a coin.

Suppose there is an experiment where the outcomes are not equally likely. Assuming the following situation: -

![Diagram showing a source with n equally likely outcomes, grouped into upper group n1 and lower group n2, with significant output being one of two messages having probabilities p1 and p2.]

**FIGURE 6.2** Experiment with n equally likely outcomes

Where the experiment has 'n' equally likely outcomes, grouped into upper group n1 and lower group n2,

\[ n = n1 + n2 \]

Assume that we want to know whether the message generated is of the upper or lower group. Therefore, the significant output is one of two messages having probabilities:

\[ p1 = \frac{n1}{n1+n2} \quad \text{and} \quad p2 = \frac{n2}{n1+n2} \]
for the upper and lower groups respectively.

The information associated with one message among 'n' equally likely messages is

$$\log_2 n$$

for \( n_1 \) equally likely messages, the information is

$$\log_2 n_1$$

This occurs for a proportion \( n_1/n \) of the time. For one message of \( n_2 \) equally likely messages, the information associated is

$$\log_2 n_2$$

and this occurs for a proportion \( n_2/n \) of the time.

In order to find out how much information is associated with the upper and lower messages, we subtract the excess information associated with the 'n' equally probable outcomes:

$$H = \log_2 (n) - (n_1/n) \log_2 (n_1) - (n_2/n) \log_2 (n_2)$$

i.e.

$$H = -p_1 \log_2 (p_1) - p_2 \log_2 (p_2) \quad \text{--- 1}$$

With a known set \( C \) of objects, the probabilities given by equation (1) can be approximated by relative frequencies, so that \( p_1 \) becomes the proportion of objects in \( C \) with class 1.

We can write \( M(C) \) to denote this calculation of expected information content of a message from a decision tree for a set \( C \) of instances. Define \( M(\emptyset) = 0 \). Let \( A \) be the attribute. Consider the possible choice of \( A \) as the attribute to test next. A partial decision tree will look like :-
Attribute A :

A1 → C1

A2 → C2

...........

An → Cn

and the expected information content is

\[ B(C,A) = ((\text{prob. that value of } A \text{ is } A_i) \times M(C_i)) \]

- replace probabilities by relative frequencies. The suggested choice of attribute to test next is that which "gains" the most information, i.e. for which

\[ M(C) - B(C,A) \]

is maximal.

An example of a calculation for information content is given in Appendix A.

Quinlan gives some idea as to the reliability of this approach :

"This method of choosing the next attribute to test has been used in a substantial number of different experiments, and does seem to give compact decision trees."

Quinlan (1983).

ID3, then, allows the user to start with a small set of examples and to expand these until the set of examples gives a comprehensive coverage of the situation, thereby allowing it to induce rules from domains otherwise too complex. This working example set is made up of those examples used to
induce a rule at any one stage of the inductive process. This is called the 'window'. The ID3 algorithm can be outlined as follows:-

1. Initialize the system with a random set of situations of user determined size (the window or the working set).
2. Deal with any clashes in the working set
3. Induce a rule from the working set
4. Mark exceptions to the current rule that occur outside the working set
5. If no exceptions are found go to 8
6. Include a user determined number of exceptions in the working set
7. Print details of the iteration
8. Print rule and end session

A clash occurs if two or more examples have the same attribute values but different class values. Finding the clash involves marking the points in the working set where the clashes arise. The user is informed that the current set of attributes is not adequate to classify the working set and the attribute set must be changed. An exception is where there is an example that conflicts with the rule induced. The process of iteration is complete when a rule has been formed that contains no exceptions and is correct for the whole of the working set. Quinlan (1983) gives some results of his findings on using ID3 for a large set of objects. A non-trivial classification problem involving 14 attributes and some 2000 objects was tested, resulting in a decision tree containing 48 nodes. Also, only 4 iterations were needed to find the correct decision tree, and interestingly, only a small fraction of the 2000 objects were needed to develop a correct decision tree. Quinlan points out:

"These features...enabled ID3 to discover correct decision trees for some large classification problems."
6.2.3 Limitations of ID3

1. Rules cannot be probabilistic
2. Each example has equal weighting - two examples the same will have the same effect as one.
3. Examples that clash cannot be dealt with

6.2.4 Expert-Ease

A brief introduction to Expert-Ease was given in section 5.3.5. There we stated that it is an example driven system, based on attributes, values and classes. The system uses the induction system derived by Quinlan. The theory of this was discussed in the last section. Expert-Ease has been used in this work for the emergency isolation valve problem.

When using the program, the first stage is to give it some examples of the problem you wish it to work on. From them, a rule is induced which distinguishes between classes of different objects. In the second stage, these rules can be applied to new data that you do not know anything about. In this respect, Expert-Ease is a machine learning type of system.

Expert-Ease uses five main screens. The file screen shows the names of all the problems held on the disk; the attribute screen is used to define the variables to be used in any given problem; the example screen is used to input examples from which the system will build up its discriminatory rule set; the rule screen is used to display the rule which the system has developed for you; and the query screen is used for a question and answer session with you and other users.

The variables are described on the attributes screen and then some examples are given on the example screen. The rule induced may then be viewed on the rule screen. An enquiry system is then generated automatically by Expert-Ease.

Examples can be either logical or integer values, and up
to 31 variables are allowed. It is possible to give the system up to 30,000 examples to work from.

6.2.5 Ex-Tran7

Like Expert-Ease, Ex-Tran7 induces rules from sets of examples provided by the knowledge engineer. The system uses an enhanced version of ID3 and the trees induced are organized according to the information theoretic measure of the discriminatory power of each attribute.

Ex-Tran7 is a more powerful tool than Expert-Ease, and provides the user with more facilities for enhancing the expert system. The tool was introduced in section 5.3.6, where the two main components, 'ACL-TRAN' and 'Driver' were mentioned. The sequence of actions taken by Ex-Tran7 is :

1. ACL-TRAN

a) Accepts examples of conditions and resulting decisions from an existing file, or from the user interactively, and induces a rule in tree format from the examples, or reads a rule in tree format from a file.

b) Converts the tree format rules into Fortran code.

c) Produces Fortran Code for the intermediate subroutines which link all the individual rules to make a homogenous knowledge base.

2. Fortran Compiler

a) The Fortran source files are all compiled - one for each sub-problem and one for the intermediate sub-routines.

b) All the compiled files are linked together with the Driver object file.
3. Driver

Driver runs the compiled executable program under their control of the problem text file which contains details of the text relating to each condition.

The problem text file contains information about the structure of the knowledge base. The rules file contains rules written or induced. ACL-TRAN converts the tree format into Fortran nested IF THEN ELSE statements.

The decision tree is made up of a series of attributes each forming a node in the tree. Each node has a set of links leading from it corresponding to the set of possible outcomes. The links from the attributes lead either to another attribute or a decision - as in Expert-Ease. The tree represents all possible combinations of attributes relevant to making a decision.

The inference logic used is propositional logic (section 3.1.2). The system searches forward through the decision tree following the links whose conditions match the current values of the attributes until a conclusion is reached.

6.3 Production Systems

Production systems as a method of representing knowledge have received a detailed discussion in Chapter 3. Here, we give an example of a specific system, as described by Winston (1984), called 'BAGGER'. A program of a similar type has been applied to the flare system design problem, and that is described in a later chapter.

6.3.1 Theory

Winston states that the most successful synthesis and analysis systems embody the rule-based problem solving paradigm. Such rule-based systems are built around rules of the 'if-then' format. In order to move from condition specifying
'if' parts, forward chaining is used and a forward chaining condition-action system results. When all the conditions in a rule are satisfied by the current situation, the rule is triggered. When the actions are performed, the rule is fired. If the conditions of several rules are satisfied simultaneously, conflict resolution is performed to determine which rule fires.

As a basis for developing a step-wise design program for solving the flare system problem, the method of Winston's was used, called 'BAGGER'.

6.3.2 BAGGER

This example involves a forward chaining synthesis-oriented rule-based toy system for bagging groceries. The system is required to bag groceries in the manner of a store checkout clerk. The system does not do optimal packing, but the system knows something of the fundamentals of bagging groceries, e.g. big bottles go at the bottom. BAGGER involves four steps:

1. Check what the customer has selected, looking to see if something is missing, with a view to suggesting additions to the customer.
2. Bag the large items, with special attention to putting big bottles in first.
3. Bag the medium items, taking care to put frozen things in insulated freezer bags.
4. Bag the small items, putting them wherever there is room.

Also, if necessary, a new bag is started. A knowledge base is now needed for the rules to look at. It must contain information about the items in each bag, the item yet to be bagged and the current step. The rules need access to information and other properties of various items, as shown in Table 6.1.
Table 6.1 'BAGGER' Knowledge

<table>
<thead>
<tr>
<th>ITEM</th>
<th>CONTAINER</th>
<th>SIZE</th>
<th>FROZEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bread</td>
<td>Plastic bag</td>
<td>Medium</td>
<td>No</td>
</tr>
<tr>
<td>Glop</td>
<td>Jar</td>
<td>Small</td>
<td>No</td>
</tr>
<tr>
<td>Granola</td>
<td>Cardboard box</td>
<td>Large</td>
<td>No</td>
</tr>
<tr>
<td>Ice Cream</td>
<td>Cardboard carton</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>Pepsi</td>
<td>Bottle</td>
<td>Large</td>
<td>No</td>
</tr>
<tr>
<td>Potato chips</td>
<td>Plastic bag</td>
<td>Medium</td>
<td>No</td>
</tr>
</tbody>
</table>

Each item has a step name, and each rule in BAGGER's rule base tests the step name. The rules are then partitioned into packets suited to each bagging step. Activation of the following rule is limited to the check order step.

B1 If the step is check order
there is a bag of potato chips
there is no soft drinks bottle
then add one bottle of pepsi to the order

The next rule gets us out of the order checking step and into the next step.

B2 If the step is check order
then discontinue the check order step
start the bag large items step

This, on the face of it appears to indicate that B2 would trigger at anytime in the check order step, preventing the first rule from acting. Conflict resolution is used in order to prevent this from happening. BAGGER uses context limiting strategy since, by convention, the first condition clause of
each rule limits the rule to a particular step. Assuming BAGGER also uses specificity ordering, B2 check-order rule can never fire as long as any other check-order rule triggers. Each step has a rule just like B2 to switch into the next step when nothing else can be done. Specificity ordering helps out in other ways too. Consider the first two rules for bagging large items.

B3 If the step is bag large items
    there is a large item to bag
    there is a large bottle to bag
    there is a bag with less than 6 large items
then put the bottle in the bag

B4 If the step is bag large item
    there is a large item to bag
    there is a bag with less than 6 large items
then put the item in the bag

Large items go into bags which do not have many items, but the bottles, being heavy go in first. The extra condition in B3 ensures this ordering. When there is a large bottle, both conditions will match, but B3 has more conditions than B4 so B3 takes precedence.

Next is a rule for handling large items if there is no room in any bag.

B5 If the step is bag large items
    there is a large item to bag
then start a fresh bag

Another step changing rule moves us on to the next step.

B6 If the step is bag large items
    then discontinue the bag large items step
    start the bag medium items step

The rest of the rules proceed in a similar manner :-
If the step is bag medium items
there is a medium item to bag
there is an empty bag or a bag with medium items
the bag is not yet full
the medium item is frozen
the medium item is not in an insulated
freezer bag
then put the item in an insulated freezer bag

If the step is bag medium items
there is a medium item to bag
there is an empty bag or a bag with medium items
the bag is not yet full
then put the item in the bag

Specificity ordering again works here, since B7 and B8 are matched, but B7 wins, ensuring that the frozen items are put in an insulated freezer bag.

Continuing :-

If the step is bag medium items
there is a medium item to bag
then start a fresh bag

If the step is bag medium items
then discontinue the bag medium items step
start the bag small items step

If the step is bag small items
there is a small item to bag
there is a bag that is not yet full
the bag does not contain bottles
then put the item in the bag

If the step is bag small items
there is a small item to bag
there is a bag not yet full
then put the item in the bag
B13 If the step is bag small items
there is a small item
then start a fresh bag

The final rule terminates action when done.

B14 If the step is bag small items
then discontinue the bag small items step
stop

A Prolog version of this system is shown in Appendix B.

6.4 Advice Giving Systems

6.4.1 Micro-Expert

Cox (1984) gives a useful account of the thinking and theory behind the expert system shell Micro-Expert. It is written in PASCAL and runs under the UCSD operating system. It is written as an advice-giving system, and is modelled on the PROSPECTOR system. This was chosen because of its method of inference - using Bayes's rule, which means that the rules can be represented as collections of trees diagrammatically. The Micro-Expert (1984) User Manual also gives a useful insight to the system.

Micro-Expert consists of two programs, a rules language compiler called EXPCOMP and a runtime system called RUNEXPT. To create a system, the user first writes his rules in the system rule language, and these are typed and edited under the UCSD system editor. The rules are stored as a source file, which is then compiled using EXPCOMP. Two output files are produced, the first being the object file which contains the structure of the model. This is in the form of a table. The second output file is the message file, which contains the ASCII text strings from the source. The compiler also produces a listing which can be directed to either the screen or the printer. The listing contains a copy of the source with line numbers followed by any error messages. A list of all the
variables and goals in the program is then given.

When the model is correctly compiled, the program RUNEXPT is executed. Initially, the program asks for the name of the model. The object file is then read for that model and then the system tries to establish the certainty of the model's goal hypothesis by asking the user questions.

The user can interrogate the system in many ways. He may ask why the system is asking a particular question, the current certainty of any hypothesis in the model, amplification of any question or the current state of the goal hypothesis.

Often, it is useful to represent the model in diagrammatic form, and an example of this is shown for the emergency isolation valve problem in Chapter 10. Here, we will describe some aspects of the advice language. Each model has a GOAL hypothesis which is the hypothesis the model is designed to prove or disprove. Each goal can have any number of hypotheses to help to establish the certainty of the goal. These come in various forms. A hypothesis of the type QUESTION with subtype CERT causes the system to ask for a certainty factor. For example:

"How certain are you that THE TEMPERATURE IS HIGH [-5..0..5]?

The user replies by typing in a number in the range -5 to 5. "-5" signifies "certainly not". "5" signifies "certainly" and "0" signifies "not sure". Other numbers in the range may also be used.

Questions can also have the subtype YES/NO which is replied to with one of the following answers:

Y signifying YES
N signifying NO
or I signifying NOT SURE
The program automatically converts these to certainty factors of 5, -5 and 0 respectively.

The other type of QUESTION subtype is NUMERIC. This causes Micro-Expert to ask a question of the form :

"Give a value for TEMPERATURE ?"

A range for the value is given and the value typed in is validated with either a High Value or a Low Value, these depending on the range allowed by the model. The system then converts the value into a probability using RULE subtypes MODULUS or RANGE which have a target value and a maximum deviation. If the numeric value of the antecedent hypothesis is equal to the target value then the probability of the hypothesis is 1. If the difference between the target value and the numeric value is greater than the maximum deviation, then the probability of the hypothesis is 0.

The other type of hypothesis is a RULE. Any hypothesis which is neither a GOAL or a QUESTION must be a RULE. GOALS and RULES have common subtypes, and these indicate the way in which the certainty of the hypothesis is inferred. The subtype AND signifies that its probability at any time is the lowest of the probabilities of any of its antecedents. The subtype BAYESIAN signifies that the current probability is a Bayesian function of the current probabilities of its antecedents. For such a hypothesis, an a priori probability must be assigned to it. This indicates the probability of the hypothesis being true before any of the questions have been answered.

As we indicated earlier the plausible reasoning scheme is based on Bayesian decision theory. Micro-Expert uses the so-called 'odds-likelihood' form of the rule. This form relates three quantities involving an evidence assertion E and a hypothesis assertion H, the prior odds O(H) on the hypothesis, the posterior odds O(H:E) on the hypothesis, given that E is observed to be present, and a measure of sufficiency LS. Then Bayes' rule can be stated as : -
O(H:E) = LS * O(H)

Odds and probabilities can be interchanged using the relation:

\[ O = \frac{P}{1 - P} \]

where \( P \) denotes the probability, and hence

\[ P = \frac{O}{1 + O}. \]

The sufficiency factor is a standard quantity in statistics called the likelihood ratio, and is defined by:

\[ LS = \frac{P(E:H)}{P(E:H')} \]

where \( H' \) means 'not \( H \)'.

A complimentary set of equations describes the case in which \( E \) is known to be absent, i.e., when \( E' \) is true. In this case, Bayes' rule can be written:

\[ O(H:E') = LN * O(H) \]

where

\[ LN = \frac{P(E':H)}{P(E':H')} \]

The quantity LN is called the necessity measure. The two factors LS and LN receive attention below.

Micro-Expert therefore takes into account uncertainty in data and tentativeness in rules. The probability of a hypothesis is calculated from the certainty factors. A certainty factor of -5 is equivalent to a probability of 0. A certainty factor of 5 is equivalent to a probability of 1, and 0 is equal to the a priori odds of the hypothesis. For intermediate
values linear interpolation is used. The current probability of a BAYESIAN hypothesis is calculated by first turning the a priori probability into odds. For each antecedent hypothesis the odds are multiplied by a factor. If the certainty of the antecedent hypothesis is -5, then this multiplying factor is equal to the LN value. If the certainty factor is 0, then the multiplying factor is equal to 1, and if the certainty factor is 5 then the multiplying factor is equal to the LS value.

When each of the antecedent hypotheses has been dealt with, the final odds are converted back to the probability of the hypothesis.

Other types of RULE subtypes include OR and NOT. OR takes as its probability the highest probability of its antecedent hypotheses, and NOT can only have one antecedent hypothesis, its probability being 1 - (the current probability of that hypothesis).

The hypotheses GOAL and RULE can have other subtypes, here, we have discussed the ones used most.

One useful feature of Micro-Expert is its 'blocking' ability. It may be necessary to have control over the order in which questions are asked by the system and to ignore irrelevant questions. This is done with the subtype BLOCKED. An example of blocking is shown in the emergency isolation valve problem.

In order to understand the workings of Micro-Expert more fully, it is worth quoting an example from the User Manual:

GOAL APPX 'The patient has appendicitis'
TRACE 'Considering appendicitis'

BAYESIAN SITE LS 10 LN 0.1
TEST LS 100 LN 0.01
YNG LS 5.0 LN 0.5

The hypothesis SITE means that the site of the pain at onset
was right-lower-quarter and is a certainty question, the hypothesis TEST is a rule meaning that the patient has pain in the right-lower-quarter for more than 24 hours and rebound pain, and the hypothesis YNG is a certainty question for the patient is young.

Let us look at the a priori probability of the goal first. The value given is 0.1 which means that the probability of a patient having appendicitis before any of the antecedent hypotheses have been resolved is estimated to be 0.1. The a priori probability is turned into a priori odds. A probability of 0.1 gives an odds of 1/9, i.e., 1 chance for and 9 against.

\[ \frac{1}{9} = 0.1111 \]

Let us assume that :-

SITE has a certainty factor of 5
TEST has a certainty factor of 2.5
YNG has a certainty factor of -1

The multiplication factors are therefore :-

\[
\begin{align*}
\text{SITE} & : 1 + (10-1) \times \frac{5}{5} = 10 \\
\text{TEST} & : 1 + (100-1) \times \frac{2.5}{5} = 50.5 \\
\text{YNG} & : 1 - ((1-0.5) \times \frac{-1}{-5}) = 0.9
\end{align*}
\]

The current odds of APPX will therefore be :-

\[ 0.1111 \times 10 \times 50.5 \times 0.9 = 50.5 \]

The current probability of the patient having appendicitis will therefore be :-

\[ \frac{50.5}{1 + 50.5} = 0.98 \]

The shell Micro-Expert is used in this work for the topic on emergency isolation valves, described in Chapter 10.
In this chapter we introduce the subject areas covered in this study, which we have termed "candidate topics". We are looking for characteristics of design problems together with the kinds of expertise inherent in chemical engineering design with the emphasis on plants handling hazardous materials. In the following sections, each of the topics are introduced, together with a section linking the problem of design to the techniques of artificial intelligence. The final section deals with the short studies carried out.

7.1 Introduction

The first section in this chapter follows on from section 1.1 and describes chemical engineering design and discusses the inherent expertise involved.

7.1.1 Design Expertise

Design expertise is a skill, and may be studied as such in its own right. Useful accounts of engineering design have been written by Simon (1984), Turner (1984), and specifically in chemical engineering by Rudd and Watson (1968).

Engineering design is a process of achieving desired goals. It is creative and must start with a clear statement of the objectives to be achieved. Specifications must be met by both the project and the products. The process starts with the identification of a need together with evaluation criteria. Designs also have a number of alternative solutions, and these must be evaluated and often reinforced by modelling or simulation. Optimization is another important technique, and is used to find the solution which best fits the requirements, also considering the constraints.

Simon (1975) speaks of design in great detail, and classifies design phases as :-
The next stage of design is the anatomy of design. This involves the detailed examination of the action of the designs in identifying and solving the problem. The design anatomy will then lead to the final plan before implementation is realised. Simon characterises the design steps as:

- Problem statement and needs formulation
- Information collection
- Modelling
- Value statement
- Synthesis of alternatives
- Analysis and testing
- Evaluation
- Decision
- Optimization
- Iteration
- Communication

The value statement is the criteria and constraints. The decision is the decision to adopt a general type of design. The optimization is then performed on this general design to obtain a more specific design of this type. Iteration is included at one point near the end of the list, but actually applies throughout.

Rudd and Watson (1968) deal with these aspects related to chemical engineering. In particular they deal with:

- Generation of alternatives
- Screening of alternatives
- Multi-level attack on very large problems
- Engineering for variability
They also emphasise the definition of the primitive problem, namely an ill-defined statement of a need.

Using the ideas of these authors as background some of the characteristics of the design skill include the following functions:

- Problem recognition
- Problem definition
- Generation of alternatives
- Selection of alternatives

The methodologies for performing the design skill are:

- Logical argument
- Rules of thumb
- Mental models
- Use of analogies

Awareness and information relate to:

- Objectives
- Constraints
- Hazards
- Costs
- Process materials
- Materials of construction
- Instrumentation
- Site characteristics
- Legislation
- Standards, codes of practice

Another interesting discussion on the theory of design is that of Simon (1981), which contains strong links with this work. Here, Simon states that design is a science of the artificial and is concerned with how things ought to be and with devising artefacts to attain goals. The discussion proceeds with descriptions including the logic of design, means ends analysis, design evaluation and generator and test
cycles. Many aspects are directly related to this work and it is worth noting some of his key points.

The theory centres around logic, including standard logic (such as propositional and predicate calculus), declarative logic, finding alternatives and the logic of search. Standard logic is to do with declarative statements and is well suited to assertions about the world and to inferences from those assertions. Generation of assertions will involve alternatives to the solution, and once a candidate has been found, we have to be satisfied that all the design criteria have been met. If an alternative does not meet the criteria, then a search must be made for another one. The characteristic of the search for alternatives is that the complete solution is built up from a sequence of component actions. The solution can be described as the complete set of actions which constitute a design. In carrying out a search, it is efficient to explore several tentative paths. Attached to the end of each path is a branch which has a number that expresses the value of a path and thus the gain in pursuing the search - a sort of heuristic search.

For a complex design, Simon talks of the power of decomposing the system into a series of semi-independent parts. Here, generate and test can play a part. Alternatives can be generated and tested against a whole array of constraints. The generators implicitly define the decomposition of the design problem and the tests guarantee that important indirect consequences will be noted and weighed.

7.1.2 Design Problem Characteristics

In deciding what topics are suitable for this study, it has been necessary to define characteristics, so that we know what to look for in the candidate topics.

Generally, candidate topics are problems that are somewhat ill-defined in nature and which cannot be solved by the application of a simple algorithm. As an example, consider pressure relief. The design of a relief valve appears to be a
relatively straightforward problem, with a logical sequence of steps to follow. However, the definition of relief requirements is a more suitable topic.

Topic characteristics are likely to include:

Not fully straightforward/algorithmic
Choice of solutions
Constraints
Conflicts

Solution for the candidate topics is likely to require:

Archetypes
Models
Rules
  Algorithms
  Heuristics
Data

These are the types of characteristics that we are looking for. Also, in these candidate topics, we must look for the types for expertise that are being deployed.

7.2 Topics Investigated

In section 7.1.2, a brief description of the characteristics of suitable candidate topics was outlined, together with possible design solutions. Here we give a list of suitable topics, covered by the characteristics mentioned previously. The list is by no means complete, but gives some idea as to the type of topics that we are tackling.

7.2.1 Candidate Topics

The area of Safety and Loss Prevention is a large one, and there are many subjects worthy of further study. However, in the scope of this project, a short list has been drawn up, and suitable topics taken further for extensive work. Our aim
is to find something out about each candidate topic, particu-
larly the different types of expertise in chemical engineering
design problems. One of the main characteristics of the candi-
date topics is that they all embody some form of expertise,
although this will be in different forms.

Specific Topics

Specific areas of Loss Prevention as suitable candidate
topics are listed below :-

Plant Layout
   General
   Electrical area classification
   Drains and sewers
Plants in buildings

Corrosive materials
Crystallising materials
Polymerising materials

High pressure
High temperature

Low pressure, vacuum
Low temperature

Furnaces
Centrifuges
Dust handling plant
Process heaters
Flares
Batch plant, especially multipurpose

Utilities

Terminals

Pressure relief and blowdown
Fire protection

General
Ignition sources, especially static electricity
Fire fighting agents

Chemical reaction runaway, decomposition
Chemical reactor protection

Explosion prevention and protection
Atmosphere control, inerting
Combustion venting of vessels
Combustion venting of ducts
Venting of dust explosions

Hydraulic problems, especially water hammer

Fugitive emissions
Alarm systems
Instrumentation
Microprocessor control

Availability, throughput

Inherently safer design

Construction defects leading to leaks
Pumps
Valves
Emergency isolation valves
Slip plate systems

Startup
Shutdown
Turndown

Trip systems

Hazards of air
Hazards of water

Topics picked out for further study are listed below.

Emergency isolation valves
In addition, certain other areas have not been the subject of an original study, but they have been reviewed. These are:

Hazard identification
  Coarse scale
  Fine scale
Modification chains
Valve system design and operation

7.2.2 Emergency Isolation Valves

Emergency isolation valves (EIVS) are used in process plant to prevent leaks of flammable or toxic fluids. Kletz (1975) has given a discussion of emergency isolation valves, particularly on the situations in which they should be used. Points on a plant which are particularly vulnerable are pumps, drain points and hose connections. Kletz gives instances of leaks in which an emergency isolation valve would have averted disasters. In one example, a pump leaked over a period of 20 minutes, and there was a 3 ton escape of ethylene. Another area in which emergency isolation valves might be an advantage are drain points, in which water from hydrocarbons is drained off, and if there is less water than expected, or the operator forgets to return to the tank, hydrocarbons are liable to leak. Another problem with drain points is blockage, caused by ice or hydrate formation. In such cases, the operator may be unable to shut the valve, and a leak may occur. Such a blockage is thought to have caused the disaster at Feyzin, France, in 1966, in which 18 people were killed (Lees, 1980). An appropriately situated emergency isolation valve may have averted the disaster by stopping the leak of propane gas.
Emergency isolation valves are usually installed between the inventory and the point of the expected leakage. However, emergency isolation valves are not installed on every piece of equipment which might leak, due not only to the cost, but because such a device itself introduces further chances of a leak. Thus, emergency isolation valves are only installed where the chances of a leak are significant, and the consequences serious. However, prevention of a leak in the first place is far better than isolating it after a leak has occurred. If however the probability of a leak is still high, then an emergency isolation valve may be fitted.

This problem is on the face of it more clear cut than others that we are describing in this chapter. However, the decision on whether or not to fit an emergency isolation valve or not is dependent on many factors, including pressure, temperature, inventory, etc. The decision involved is a straightforward yes/no type, but many factors are used in establishing the decision. The problem has been tackled in various ways, with particular emphasis on an olefin plant, and these are described in Chapter 10.

7.2.3 Fire Protection of Storage Tanks

Fires in process plant are a very serious hazard both to life and property. Petrochemical plants with large amounts of storage are particularly at risk and very stringent controls for prevention and protection are necessary.

Normally, fires in process plants occur due to leakage or spillage of a flammable material from the plant. Leaks can occur from a variety of sources including vessel, pipe or pump failure, or from glands and seals. Ignition of the leak can take a number of forms, as can combustion. Ignition may occur at the point of issue, or a vapour cloud may form which may take some time to ignite. Alternatively, a flash fire may occur. Prevention of a fire is thus a matter of eliminating leaks and ignition sources.
Here, we are concerned with the protection of process plant from fires, particularly in storage tanks. There are a number of codes of practice dealing with fire protection, together with extensive literature on the subject, notably the Fire Protection Manual (Vervalin, 1985).

There are two kinds of fire protection, which can be considered. The first is "passive" protection, which deals with fire elimination, including material transfer, fire-proofing, fire spread and plant layout. The second is "active" protection, which deals with fire warning systems, fire detection, fire-fighting agents and water supply. Obviously, not all the protection measures will be required in all installations, so distinctions are made, including such factors as type and size of tanks, layout of the tanks and the type of material being stored.

In this work, we have concerned ourselves with active fire protection chiefly for hydrocarbon installations. We have established the main criteria for designing a fire protection system, and derived rules with a view to building a knowledge-based system. The problem appears to be one in which a simple algorithmic approach is applicable and has been programmed in Prolog as such. However, the interpretation of certain phrases associated with the algorithm such as 'near' can involve expert heuristics.

An attempt has also been made to treat it as a classification problem using a version of ID3 written by the author. However, it was decided that this problem is not one of classification, partly because the expertise comes in the form of rules given in codes rather than examples and partly because the number of classes appears to be almost as large as the number of specific examples.

The design of a fire protection system is therefore essential at an early stage in the proposals for a storage facility. Further, good operation and maintenance are necessary during the working life of the plant. Lees (1980)
makes the following point:

"Safety in storage is as much a matter of operation and maintenance as of design."

This aspect of design in the safety of process plant is discussed in more detail in Chapter 9.

7.2.4 Flare Systems

In many industrial installations such as oil refineries, petroleum plants, storage and shipping facilities gaseous and liquid effluents which are a hazard to the environment must be disposed of. For gaseous effluents, a flare relief system is such a disposal facility, and its function is generally to deal with materials vented during normal and emergency conditions. Flare systems separate liquid for reprocessing and burn off the gases. They must cope with all situations, whatever the cause of the release and dispose of the effluents in a manner which is acceptable to the environment.

A typical flare system is made up of a flare stack and a series of pipes which gather the gases which are to be vented, along with a flare tip, often with a smoke suppression device, seals on the stack to prevent flashback, and a knockout drum at the base of the stack to remove the liquid from the gases. The main flare header should slope towards the liquid knockout drum in order to drain the liquid into it, and low points should be avoided as liquids and solids could collect.

Elevated flares, such as that described above can create a public nuisance due to noise, light and smoke. Often, a ground flare is used alongside the elevated flare, and will be used for burning the continuous excess gases that are generated in a process plant operation. The elevated flare will then be used in emergency conditions, for start-up or shutdown and any abnormal operation that may occur. They are basically large capacity open roof incinerators and are of the order of 10-20m in height.
Lees (1980) lists some of the hazards inherent with flaring operations:

1. Explosions in the flare system  
2. Obstructions in the flare system  
3. Low temperature embrittlement of pipework  
4. Heat radiation from the flare  
5. Liquid carryover from the flare  
6. Emission of toxic material from the flare

Even if not all these features are present to cause a hazard, the latter three may cause environmental problems. Other features likely to cause such a problem are smoke, noise and glare. One of the main causes for concern is explosion in a flare system. There is always a source of ignition present, so it is vital that air is kept out of the stack. This and other hazards will be discussed in more detail in the chapter on flare systems, Chapter 11.

7.2.5 Fugitive Emissions

Releases of chemicals from process plants to the atmosphere can be either controlled or fugitive in nature. Controlled emissions come from stacks and vents, and the details of their release are known such as the rate of release, their composition and their direction. It is essential that this kind of information is known about controlled emissions from an environmental point of view. Fugitive emissions, however, are quite different. There is no control of their rate, composition or direction, and they can occur at any time. Such emissions occur from a wide range of plant equipment, including pumps, flanges, compressors and valves, which are the largest contributor. The reason that such emissions are important is environmental, to increase the public credibility of the plant, and to reduce operating costs of the plant itself. The financial gains to be made by reducing fugitive emissions are quoted by Jones (1984). Potential losses can be as much as $1500 per day.
There are various factors which influence fugitive leak rate, which include high temperature, seal technology and maintenance. Jones (1984) gives the effect of these on leak rate. High stream temperature leads to a drying out of the seals, valves in "hot" service should have special status, and increased attention should be given to reduce the fugitive loss. Wetherold (1983) gives a cost benefit analysis of a monitoring and maintenance programme for valves. Valves are inspected every 6 months and those that are leaking replaced, is estimated to give a nett saving for a large olefin plant of $125,000 per year.

This problem appeared to be one which is worthy of further study, and the use of heuristics seem an appropriate way forward. Fugitive emissions have been subject to only a short study in this project, and this is described in section 7.3.1.

7.2.6 Pressure Relief and Blowdown

Every kind of pressure vessel, including reactors, distillation columns, storage tanks and heat exchangers needs a pressure protection system, with provision for protection against overpressure. Here, we are concerned with situations in which the pressure rise is gradual as opposed to sudden pressure increase such as that caused by explosions. A number of codes of practice exist, particularly the American Petroleum Institute codes, API RP 520 on the design and installation of pressure relieving systems, and API RP 521 which is a guide to pressure relieving and depressuring systems. In addition, individual companies have their own codes of practice. Legally, the Factories Act of 1961 states that steam boilers, receivers and air receivers should have a safety valve in order to prevent over pressure. The Chemical Works Regulations of 1922 require the fitting of a safety valve or similar to prevent overpressure on stills and closed vessels handling gases. Also, under the Health and Safety at Work Act of 1974, there is a general duty to provide safe equipment.

The threat of rupture may arise from a number of diffe-
rent sources including excessive pressure or vacuum, excessive temperature, overfilling, corrosion or explosion. The first and most difficult stage in relief blowdown design is the identification of potential hazards. Some, such as fire are obvious, but others may not be so. Also to consider is which of the causes are going to impose the largest relief load. The next stage in the design is to decide the kind of protection required. In some cases, protection may not be required, e.g. if the source of overpressure is a pump which has a maximum delivery pressure less than the design pressure of the vessel to which it is pumping, then protection will not be needed. The next question is whether pressure relief is the most practical way of dealing with the problem. Alternatives therefore exist, such as an inherently safer plant, high-trip systems or ignoring trivial hazards.

The design of relief blowdown systems is a complex task, and, as stated by Fitt (1974), it is not practical to apply quantitative hazard analysis techniques to all pressure relief problems, and the analysis must therefore be based on intuitive judgement.

There are many cases of overpressure causes and consequences, and in the past each problem has been dealt with on an individual basis. Examples include bursting of heat exchanger tubes, control failure, pressure letdown, power failure and cooling water failure. In this project we have tried to quantify the problem and obtain rules for relief blowdown design in a more general way. The intention has been, as in all the candidate topics, to try to understand the nature of the expertise involved in the design of a relief system and to indicate the possible application of expert systems to it. We have tried to indicate also, whether the problem is one of deep structure or whether it is a straightforward if long decision type problem. This work is described in Chapter 12.

7.2.7 Static Electricity

Static electricity is an important source of ignition in
process plants. There have been many mysterious explosions, the cause of which was eventually traced to static electricity. There is now much more information available on both the nature and on the prevention of static electricity, notably, British Standard 5958, parts 1 (1980) and 2 (1983). Despite this, it remains a phenomenon which is often not well understood or appreciated.

Static electricity is a potential source of ignition wherever there is a flammable mixture of gas or dust. Its generation is essentially a surface effect, which is associated with the contact and separation of dissimilar bodies. When the surfaces are separated, one body tends to be left with a positive charge, and the other with a negative charge. If the bodies are good conductors of electricity, the charge moves freely and both bodies are effectively restored to their original state through the last points of contact at separation. If one or both of the bodies are poor conductors, the charge will be retained.

Many industrial processes involve surface contact, and movement and separation of poorly conducting materials. These processes may be classified in terms of the phases involved, e.g. gas-solid, dust-powder, or by the nature of the process, e.g. pneumatic conveying. The hazard of static occurs in the process industries in fluid handling operations, dust and powder operations, truck filling and agitation. Also, dry refined oils, such as gasoline, kerosine, jet fuels and similar oils will become charged during the following operations:

- Flow through a pipe
- Agitation
- Overhead splash filling of tanks
- Settling of water from a volume of oil following a water wash
- Filtering of oils
- Rapid evaporation

123
For a fire on a plant, air, a flammable vapour and a source of ignition must be present. It is difficult to prevent air and flammable vapours from forming an explosive mixture, so the source of ignition must be eliminated. There are a number of measures that can be applied to prevent static from building up and these can be incorporated at the design stage by following rules. These will be discussed in the Short Studies section, described in section 7.3.2.

7.2.8 Expertise in Candidate Topics

The candidate topics that we have discussed above are characterised by the fact that they all embody some form of expertise. This expertise comes in different forms, and the aim of the current work is to find the characteristic features of the design of these topics in order to understand the expertise involved. This will be shown in subsequent chapters together with some example solutions.

7.3 Short Studies

7.3.1 General

Coarse Scale Hazard Study

The shell, Micro-Expert has been used to do a mockup of an expert system which determined how probable it was that a particular plant would give rise to serious hazard problems.

HAZOP Study

There has been a number of works concerning expert systems and HAZOP studies. Ferguson (1985) looks at an expert systems approach to HAZOP, and also work by Parmar (1986) has shown the application of Prolog to HAZOP. This, and other related work is described in Chapter 8.
Modification Chains

It may be possible to develop rules which allow the user to check the probable effects of a plant modification which he is proposing.

Valve System Design and Operation

There can be a problem in designing a complex system of valves so that they operate safely. Possible hazards are getting one fluid into another which is compatible, e.g. flammables and air. The problem of valve sequencing has been described by Rivas and Rudd (1974).

7.3.2 Fugitive Emissions

We introduced fugitive emissions in section 7.2.5. There, we explained the nature of the problem and the extent to which companies can benefit from having a fugitive leak control programme. Here, we go into more detail, and have tried to work out some rules and heuristics for their control, particularly at the design stage.

The main work concerning fugitive emissions comes from the B.O.H.S. Technical Committee (1984). There, all aspects of the problem are explained, together with various control measures that can be employed. Work in this area also comes from Jones (1984), Morgester et.al. (1979), Rosebrook (1977) and Wetherold (1983).

Measurement of Fugitive Emissions

Direct quantification of fugitive emissions is difficult because of their inherently diffuse nature. There are a variety of techniques available, all involving assumptions and approximations.

Fugitive emissions can be measured at a spot remote from the leak, such as at a boundary fence. However, if control
measures are to be implemented, then they have to be quanti-
fi ed at source. The measurement techniques used include soap
bubble, in which soap is put on the surface and the bubbles
are observed and bagging, in which the emission is contained
in a known flow of air, and the resultant concentration
measured. The use of data collected by such methods is depen-
dent upon mathematical models of containment and dispersion.
The B.O.H.S. state that the major deficiency lies in the
absence of a standard technique. Ideally, the techniques for
measuring fugitive loss should be quick and simple, provide
quantitative data on mass, and be reproducible and reliable.
None of the techniques mentioned fulfills these requirements.

Control of Fugitive Emissions

In order to prevent a leak from a joint, the joint must
be sealed effectively, and the hygienist must be aware of the
alternatives available to him. Also, the design engineer must
know when it is necessary to seal a potentially leaking joint.
There are two classes of seals, static seals, which have no
relative movement of the seals surface, such as pipe joints
and manhole covers, and dynamic seals in which a seal must be
made between surfaces that move relative to each other. The
motion may be rotary (centrifugal pump) or reciprocating (pum-
p/compressor). Seals used for dynamic applications can be
either contact seals in which the seal bears against the
mating surface under positive pressure, or clearance seals
which operate with a positive clearance between the two sealed
surfaces. There are many types of seals available for diffe-
rent applications, among them mechanical seals, packed seals
and labyrinth seals. Mechanical seals are the most effective,
particularly when used as a double mechanical seal. These are,
however, expensive and add to maintenance requirements. Jones
(1984) points out that for valves, use of 100% foliated carbon
packing considerably reduces fugitive losses, and it also has
the advantages of flexibility and self lubrication, which
means that drying out does not become a problem. The B.O.H.S
point out that use of 100% foliated carbon may prove to be a
major advance in the control of fugitive emissions.
Other workers in the field suggest alternative ways of dealing with fugitive leaks. Blackwood (1982) implies that the most effective control is to change the process or re-route materials to prevent air contamination. He goes on to point out that for now, fugitive emissions control is left to the imagination of the engineer. Goltz (1984) recommends an annual check of all gas or vapour valves using soap solution and carrying out selective maintenance. This has shown to be both effective and economic. Wetherold et. al.(1983) states that an effective valve inspection and repair program can result in substantial savings. They also say that for a given plant, there is an optimum monitoring frequency which will result in maximum savings.

**Heuristics for the Control of Fugitive Emissions**

The problem of fugitive emissions looks to be one that could be solved by the application of heuristics. It is a problem which is not fully defined, one that is covered by constraints, it has a choice of solutions and it is a problem of design. However, fugitive emissions are dependent on a large number of parameters, that are difficult to define for any one leak, and the measurement techniques are unreliable. There are no rules to say when an item will leak, or to what extent it will leak, although there are rules which state what type of seal to use in certain conditions, but this is not really tackling the problem at hand. The solutions outlined above are all clear cut solutions that appear to work well. As a result, it has been difficult to derive rules and heuristics that define and solve the problem. Another barrier in the definition of the problem is when does a fugitive emission become a leak ?. The Environmental Protection Agency in America merely define an arbitrary leak rate, above which preventative maintenance is required.

The problem of fugitive emissions was not carried further in this work largely because the view was forcefully put to us from one source that there is a simple engineering solution to most fugitive emission problems, namely 100% foliated carbon.
7.3.3 Static Electricity

Static electricity was introduced in section 7.2.7. Here we take the topic further by looking at some of the ways in which static can be controlled in various installations. The basis for this has come from British Standard BS:5958 Part 2 (1983). There are various other works concerned with the hazards of static electricity, notably Klinkenburg and van der Milne (1958), Gavis (1972) on the origins of static electricity, Eden (1972) on the hazards of static in powders, and Howard (1985) and Mahley (1972) on the hazards of static in the petroleum industry.

Causes of Static Electricity

When liquid flows through a pipe, charge separation occurs and the liquid emerging from the pipe is charged. The amount of charge is dependent upon flow conditions. Turbulent flow generates more charge than laminar flow for a single phase liquid. The flow of a liquid/liquid or a solid/solid mixture through pumps etc. also causes charge to generate. Such generation may be greater than with a single flow liquid due to the increased interfacial area. Fine particle filters can be prolific generators of static, and generally the more stringent the filtration requirement the greater the charge generated. Settling also causes problems, but such generation is complex, and little data is available. Other causes of static generation in liquids is splashing of liquid jets, ejection of liquid droplets from a nozzle, stirring and mixing, and discharges involving liquids, e.g. mists and sprays. High resistivity solids and particularly powders are also prone to charge generation.

Control of Static Electricity

BS 5958 part 2 (1983) classifies the problem according to the items of equipment that need protection, such as storage
tanks, either metallic or non-metallic, road and rail trucks, ships and barges, filters and water separators, pipelines and containers for high resistivity powders. The code considers operations such as tank filling, filters, gauging and sampling and cleaning for each of the items mentioned above. Control of static electricity falls into the following categories:

- Prevent formation of explosive mixture
- Prevent accumulation of static on equipment (earthing)
- Prevent accumulation of static in liquids (anti-static additives)
- Reducing the generation of static

If we consider the case of a fixed metal tank storing liquid, charge can be generated within the tank from splashing, from the pipeline feeding the tank, and also from personnel maintaining the tank. An ignition hazard is created when charge is retained in the liquid, on insulated conductors, or on personnel in the presence of a flammable vapour/air mixture. Such hazards can be avoided by observing a few simple precautions. The tank and the associated fittings must be in good electrical contact with each other, such that resistance at all points is less than 10 ohms. Personnel should not present a hazard (e.g. suitable footwear), and new or repaired tanks should be cleared of debris. In filling the tank, splashing must be avoided. For liquids up to 50 pS/m the inlet should be designed to minimize turbulence, linear flow velocity should be less than 1 m/s, and for liquids greater than 50 pS/m an anti-static agent should be used. These guidelines are only an outline of the precautions needed, but they demonstrate the kind of guidelines that are needed.

**Rules for the Control of Static Electricity**

From the above, a number of rules have been derived, for example:

If the item is new/has been out of service, then an inspection of the item should be made prior to use.
If a flammable atmosphere can form, then splash filling should be avoided.

If the liquid conductivity is up to and including 50 pS/m, then the inlet should be designed to minimize turbulence and agitation.

If the liquid conductivity is 50 pS/m then the linear flow velocity in the feed pipe should not exceed 1 m/s.

The item should be in good electrical contact with related items, and the resistance at all points is less than 10 ohms.

etc.

The problem of static electricity appears to be basically algorithmic with expert interpretation of some of the words used. The British Standard goes a long way to providing the expertise, e.g. it does not talk about 'high' conductivity, but gives a conductivity break point of 50 pS/m.

7.4 Summary

This chapter has highlighted the characteristics of some design problems, candidate topics and the expertise in such topics, and two short studies have been carried out. All the topics are quite different, although all embody expertise for which we are trying to identify information in an explicit form. We have seen that not all the topics are suitable for further study, not because of their lack of expertise, but their solutions are clear cut and well defined. If expert systems are to be useful, then it is important that relevant topics are studied, and correct problem-solving strategies employed. At this stage it is worth noting the comments of Shirai and Tsujii (1984) who state that for artificial intelligence work, problems which have a well defined method of solution should be excluded.
8 Hazard Identification

8.1 Introduction

In this work we have been dealing with the specific area of plants handling hazardous materials. It is relevant at this point to consider the general problem of hazard identification, the expertise involved and possible problem solving approaches to use. In particular, we review previous work on hazard identification based on fault propagation, and briefly look at coarse and fine scale hazard identification.

8.2 Hazard Identification

Hazard identification has two main goals. They are to reveal substances or processes which have hazard potential and to identify all conceivable threats to the process which might lead to loss of containment.

Hazard identification is a large and complex area and there is little in computer aided design to help.

8.2.1 General

In the context of this work, we should consider the nature and characteristics of hazard identification and establish the expertise and types of problem solver which might be appropriate.

As stated above, hazard identification is a large and general topic. It covers many areas including check lists, properties of the materials involved including reactivity, instability and explosibility and safety audits. Often pilot plants are built to identify unforeseen hazards, and use may be made of hazard indices such as the Dow Chemical Company's Fire and Explosion Index which assigns points for each hazard encountered. Hazard identification can be broken down into two main areas, that of coarse scale hazard identification and fine scale hazard identification.
Coarse scale is the determination of general hazards which may occur on a process plant, and draws on the expertise involved in check lists, safety audits and safety management. Fine scale is, by nature far more specific and involves hazard and operability studies (HAZOP) and fault trees etc. Lees and co-workers draw on the methods of fault propagation equations in order to trace a fault through a pipe, for example, and both methods have inherent expertise and some expert system characteristics. It is possible therefore to identify in hazard identification techniques some of the methods used in artificial intelligence, particularly in fault trees, which use top down and bottom up approaches. Work on a HAZOP style program involves a suite of programs partly written in Prolog, and having expert system features, and potential for enhancement of those features. This is described in section 8.4.1

In this chapter we describe the two methods of hazard identification and then concentrate on expert system features inherent in them. By studying this work we have been able to tentatively state the nature and characteristics of hazard identification and suggest possible problem solving strategies to enhance the methods, so making expert system development more promising. The next sections deal with the two methods of hazard identification, coarse and fine scale.

8.2.2 Coarse Scale

The core of this section is based on the work by the Chemical Industries Association (1977). In this section we will describe what is involved in coarse scale hazard identification.

When a major design is started, safety and loss prevention matters should be the concern of the whole design team. Thus, from the outset it is apparent that there are many experts involved who are helping to make the plant a safe one. This means that an appropriate system of work should be formalised for it to be effective. A hazard identification would
have little effect if the team were merely suggesting hazards 'ad hoc' as they went along. We can therefore identify the objects of a system of work. They are to fully exploit the experience of the team and to provide checks to keep errors down. A set of standing orders and instructions for particular activities is typically the way forward.

Thus a framework of knowledge is built up in these formalised stages, from research and development to the operation of the plant. It is also important that the items in the frame be executed correctly, both in phasing and matching. Another characteristic we can see developing is that of backtracking. An early task of the design team is identification of hazards. This first study will be coarse and will need revision at a later stage.

One of the most useful aids to coarse scale hazard identification is the safety check list, as outlined in detail by Wells (1980). The first aspect is the hazardous characteristics of the chemicals used in the process, including reaction synthesis and scale up. Next the basic process design must be carried out. Initially the process as a whole will be examined to determine whether it is inherently dangerous.

This early study involves the plant layout diagram as the working document, and is a very general study. It is intended to highlight major hazards, chemical data and plant location features. A useful checklist at this stage is (CIA, 1977):

<table>
<thead>
<tr>
<th>Fire</th>
<th>Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explosion</td>
<td>Vibration</td>
</tr>
<tr>
<td>Detonation</td>
<td>Noxious material</td>
</tr>
<tr>
<td>Toxicity</td>
<td>Electrocution</td>
</tr>
<tr>
<td>Corrosion</td>
<td>Asphyxiation</td>
</tr>
<tr>
<td>Radiation</td>
<td>Mechanical failure</td>
</tr>
</tbody>
</table>

8.2.3 Expertise in Coarse Scale Hazard Identification

The problem of coarse scale hazard identification is very
general, although specific to the design at hand. It is an ongoing problem, continuing throughout the design stage until the design is 'frozen'. The knowledge required is huge and includes aspects such as the process itself, together with that of specific installations such as reactions, storage and columns. Many different kinds of expertise are therefore involved, although there is unlikely to be algorithmic expertise involved. Classification will play a part in refining the hazards identified, but essentially the expertise of awareness of problems is the major component.

It is possible to see a framework of coarse scale hazard identification knowledge building up. Each aspect would have a general heading and particular cases would be specific instances of the heading. This could be represented in a semantic net type system, which was introduced in section 3.1.1. The nodes in such a system would be used for representing descriptors from the safety check list, and the links might represent the relation between them in either 'is-a' format or 'has-a' format. For example, the object vinyl chloride is-a very toxic substance, or LPG is-a fire hazard. These are obviously simple examples, but may possibly represent how the hazards assessor is thinking. Execution of the nets would have to have strict control, since phasing and matching are also important.

If a problem solving strategy were to be suggested, then a forward chaining approach seems the most likely. The reason for this is that throughout coarse scale hazard identification data and knowledge are collected in order to satisfy specific goals, and ultimately the common goal - that of a safe plant. The experts therefore work forwards given the plant information, and incorporate coarse scale hazard identification in the initial stages of design.

An early study carried out in the Department of Chemical Engineering, Loughborough University, considered coarse scale hazard identification. The presence of hazards was considered as a goal, to be set aside using a backward chaining method. In the view of the present author, this is probably not the
right approach.

8.2.4 Fine Scale

One of the methods of hazard identification on a fine scale is the HAZOP study. This is carried out by a multi-disciplinary team of specialists headed by an experienced chairman. Kletz (1983) gives a general account of the HAZOP process, and the HAZOP Study Guide (1977) by the Chemical Industry Saftey and Health Council of the Chemical Industries Association gives a detailed account.

HAZOPS are carried out once the design has been 'frozen' and the potential major hazards established. The purpose of a HAZOP is to discover deviations from intention and so discover possible hazards and operating difficulties.

The working document is the process flow diagram. It is examined pipe by pipe and parameters such as flow, temperature, pressure and concentration are reviewed using a check list of guidewords. The guidewords are NONE/NO, LESS, REVERSE, PART OF, MORE THAN and OTHER THAN. The causes and effects of each deviation are then considered and the need for action is evaluated. The process is normally carried out on pipes connecting the units, as the consequences will usually show up in the units themselves. The results of a HAZOP are given in a table, which has the following format:

<table>
<thead>
<tr>
<th>Guideword</th>
<th>Deviation</th>
<th>Possible causes</th>
<th>Consequences</th>
<th>Action required</th>
</tr>
</thead>
</table>

The HAZOP process is therefore a very skilled and time consuming process and is very costly in terms of man hours.

In order to gain a better understanding of the expertise in fine scale hazard identification, and in particular HAZOP, it is necessary to give an account of the HAZOP study procedure.
HAZOP is a critical examination procedure which can go to varying degrees of detail. Each part of the process design is questioned to discover how deviations from the intention of the design can occur.

The team leader will ask questions of the team based on the guidewords given above. The guidewords are combined with a set of physical variables which include flow, temperature and pressure. Each guideword and variable combination is considered to determine, how if at all, it could be caused and what if any, the consequences would be. Three cases are then to be considered. One is if the causes of a deviation are unrealistic or improbable, in which case the derived consequences are rejected. The second case is that the deviation may be trivial, or may have no consequences. Again the deviation is rejected. The final case is that of a deviation which has both valid and/or probable causes which lead on to a potentially hazardous situation. Here, the deviation together with the causes and consequences are noted down for remedial action.

8.2.5 The Expertise in HAZOP

The HAZOP process involves a team of highly qualified people and is therefore a process that has much expertise involved. It is difficult initially to identify any formalised methods of working or any specific expertise involved. However it is possible to break the process down and examine each step of it. Firstly in this section it is worth noting the kind of knowledge that a HAZOP team need to have.

Of course, the team need to be aware of the actual study procedure. For example, the order of examination, the meaning and use of guidewords and results presentation. Knowledge then required is in the form of specific facts, including the process under consideration the process equipment and materials, the hazards, costs and probabilities. This kind of knowledge is 'experience expertise', which is knowledge built up through design, implementation and operation of plant.
Perhaps the most expert knowledge involved is that of pruning or filtering. Consider the three cases we mentioned above - there is no point in listing all the actions for all the possible causes and consequences that are thought up. This is impractical and a waste of time. The team must be satisfied that they are justified in disregarding some of the causes and consequences. Inevitably this pruning will result in the use of heuristics.

Now let us go back and look at the HAZOP procedure. Roach and Lees (1981) have formulated a HAZOP algorithm as a result of recording an actual study in progress. This is interesting and shows that the HAZOP procedure taken as a whole is an algorithmic one and thus involves algorithmic expertise.

Let us now look at the HAZOP activities. Again, Roach and Lees were able to list a formal set of activities which shed some light onto the nature and expertise of the process. The activities are:

- Selection
- Identification
- Generation
- Explanation
- Estimation
- Checks
- Specification

The interesting activity in this list is the generation of causes and consequences. Roach and Lees report that the team work backwards to a cause and forwards to a consequence. This implies that hazards are noted and the possible causes generated and the consequences are generated and the actions noted. The authors also report that in fault diagnosis generally, the high probability paths are followed first and other paths only when the former are exhausted. This is opposed to following all paths in a fixed order regardless of probability. This is directly analogous to a best-first-search described in Chapter 4, section 4.3.4. In best first search,
the solutions found by a breadth first search are evaluated and the more highly probable ones are considered first, and least likely paths are explored later or discarded if an exhaustive search is not needed.

The other aspects of the HAZOP procedure are discussed in detail by Roach and Lees. In this section we have highlighted aspects that are relevant to expertise and problem solving.

A HAZOP then, involves much expert knowledge and commonsense. An expert system for the procedure has obvious advantages, and we shall examine later a computer aid which has expert system potential for hazard identification.

8.3 Fault Propagation

We will examine fault propagation by first looking at fault trees, followed by fault propagation applied to hazard identification.

8.3.1 Fault Trees

Fussel (1976) gives an early account of fault tree concepts and techniques. A fault tree starts with a top event which is usually a hazard. This top event must be chosen with care, since the probability of the hazard occurring must be appreciated. A fault tree is used to identify possible causes of the hazard, some of which may not have been envisaged. Fault tree analysis takes the form of system definition and construction, together with quantitative and qualitative evaluation.

Constructing fault trees can be a time consuming task, and therefore there is a need for a computer method for fault tree synthesis. The work by Kelly and Lees (1986) is along these lines, in which a computer method based on the flow of information in the plant has been developed. The basic methodology is to use the line diagram, convert that by decomposition into a block diagram of the plant and then derive a model
for each block, models being held in a model library. The models show how the output variables are affected by the input variables. The models are configured using rules, and rules are also used to synthesise the tree. The relationship between the inputs and outputs is given in the form of mini-fault trees. The main application of the computer processing is in the generation of the models in the form of mini-fault trees and the generation of the fault tree itself.

8.3.2 Fault Propagation

A fault propagation model for a process unit as described by Kelly and Lees (1986) is a representation of input process variable deviations into output variable deviations and also of the initiation and termination of these deviations. The behaviour of a unit is modelled in three forms:

1. Propagation equations
2. Event statements
3. Decision tables

Fault propagation modelling involves the representation of the initiation of a fault at one point, its propagation through the plant and its termination at another. Fault initiation involves the relation

Initial fault : Process variable deviation

fault propagation the relation

Process variable deviation : Process variable deviation

and fault termination the relation

Process variable deviation : Terminal fault

Functional equations are used to model the propagation of a fault through a unit. These are equations which describe the relation between an output variable of a unit and the input.
and other output variables of the unit. e.g. if $z$ is an output variable and if $x$ and $y$ are the two input variables which affect $y$ such that an increase in $z$ is caused by an increase in $x$ or a decrease in $y$ the functional equation is:

$$z = f(x, -y)$$

Figure 8.1 shows a mini-fault tree for this equation. Further branches can be derived from the event statements.

![FIGURE 8.1 Mini-Fault Tree](image.png)

Fault propagation has therefore played an important part in computer aided hazard identification, and some of the applications are discussed below.

8.3.3 Alarm Analysis

Many processes have a large number of alarms, and it is desirable to analyse them using a computer based method. The problem is particularly acute in the nuclear industry, and alarm analysis has been pioneered by this industry. Two objectives exist in alarm analysis, that of interpreting fresh alarms as they appear in real time and the other is to identify the original cause. Lees (1980) discusses various methods of alarm analysis. Most involve the flow of information through the plant, i.e. fault propagation. On the nuclear reactor at Oldbury, the propagation of a fault is followed up through successively higher levels of the tree. All the impor-
tant alarms are displayed together with the cause alarm at the head of the group and the effect alarms below it. The manual creation of alarm trees is time-consuming and attempts have been made to make it more systematic. In one approach (Andow and Lees, 1975), a network of process variables on which there are alarms is produced. The method is based on a list processing technique which produces the alarms automatically from the unit models. The diagrams produced indicate the search paths for prime cause alarms. In another approach, more highly structured alarm information has been produced in the form of fault trees by Martin-Sollis, Andow and Lees (1978) using an early form of the fault propagation technology as described above.

At this point it is worth quoting a point from Lees (1980):

"...the techniques of list processing are widely used in the implementation on computers of the methods of artificial intelligence, such as computer chess playing and are likely to find increasing applications in process control."

8.4 Computer Aided Hazard Identification

In the preceding sections to this chapter we have discussed general aspects of hazard identification. Here we describe one particular method developed for automatic hazard identification.

A manual HAZOP was described in section 8.2.4, and it was noted that the process is a very time consuming and costly one. The work by Parmar and Lees (1987) is described here showing its similarities to HAZOP and its expert system qualities. The deficiencies of the program are outlined which give some indication as to the nature of a full expert system for hazard identification.
8.4.1 Computer Aided Fine Scale Hazard Identification

The work of Parmar and Lees is a method of modelling fault propagation for hazard identification implemented on a computer based interactive facility. The information produced is similar to that of a HAZOP study, though less complete. The work is directly analogous to that of Kelly and Lees discussed earlier. The plant line diagram is decomposed into blocks. A model is written for each block, the set of models being held in a model library. In this case the models consist of a set of rules (as opposed to mini-fault trees in the Kelly and Lees method). Again, the model is generated automatically from functional equations and event statements. A cause and consequence search is then carried out.

Model Rules

Propagation equations were discussed in section 8.3.2. The equations, in terms of the initial and event statements may be expressed as rules. These are expressed in production system form :-

If A occurs, then B may/will occur

The propagation equation

\[ q_2 = f(q_1) \]

is equivalent to the rule

If \( q_1 \) is hi (lo), then \( q_2 \) may/will be hi (lo)

Another example is :-

The initial event statement

\[ \text{maldistribution} > x_2 \text{ hi, } x_4 \text{ lo} \]

Is equivalent to the rule
If maldistribution occurs, then $x_2 \text{ hi}$ and $x_4 \text{ lo}$ may/will occur.

The propagation equations and event statements make up the basic information for a unit model. They can be entered directly using the formats given above. The corresponding rules can then be generated automatically.

**Material Models**

The modelling of the process is handled in terms of materials used, both process materials and materials of construction. The model defines the characteristics of the material in terms of its properties which are characteristics which could lead to a consequence without being activated by a process variable deviation, and susceptibility which could lead to a consequence only if activated by a process variable deviation.

Searches are made for specific realizations of fault credibility and also in each line and vessel with respect to process material. The first search is for any material property resulting unconditionally in a specific realization, and the second is for any combination of susceptibility and process variable deviation which results in a specific realization. Realizations found are entered into the consequence list. Process materials also have two other types of property, one being leak realization in combination with materials of construction, and the other is a noxious property, relevant to escapes.

**Materials of Construction Models**

The materials of construction model describes the materials of construction. Characteristics are defined by its susceptibilities, as above, an example being if a material is susceptible to a leak by brittle fracture if the temperature is low.
Again, two searches are made of each line and vessel. They are for process material properties which activate a susceptibility and result in a specific realization, and for any variable deviations which activate a susceptibility which results in a specific realization. As before, these are then entered into a consequence list. Again, material of construction knowledge is entered as part of the configuration data.

Other Models

Other models that go to make up the configuration data are listed below.

Source model

Process variable deviations at the inlet of any line crossing the plant boundary are generated by the source model. It consists of a set of event statements and is also used to account for impurities entering the process stream.

Pipe model

This is a special model used to consolidate the many faults that can occur in pipes. In the cause list for every unit are the common faults leak and blockage, and application of the generation rules will lead to many such faults being shown up.

Escape model

A common fault on chemical plants is a leak from a process unit. These come in two forms – that of a major leak which affects flow and pressure variables and a minor leak, affecting only the environment. The minor leak aspect is treated using a separate model for the event escape. A separate escape heading is created at the end of each line search and all the leaks for the units in the line are summarized in the cause list. Using the process material model a search is made of each leak and any noxious materials are identified.
These properties are listed in the consequence list.

Other faults that are treated by the method are leak fault, blockage fault and impurities.

**Cause and Consequence Generation**

We stated above that in a manual HAZOP the process is started with a variable deviation in a line. Causes and consequences are subsequently generated backwards and forwards respectively for each unit. This is analogous to the computer method in that causes are generated by searching each line and unit for critical events which are causes of a deviation. The consequences are then generated by searching for terminal events which are consequences of the deviation.

A given cause normally results in most of the consequences, but not necessarily all. Similarly, a given consequence results from most of the causes, but not necessarily all. The link between causes and consequences is the variable deviation itself.

The hazard identification table generated lists all the causes and consequences of a variable deviation.

**The Program**

The main input for the program is the plant configuration and the models from the model library. Other data includes the material model, materials of construction model, source models, pipe model and escape model together with fault data such as leaks, blockage and impurities.

The program is actually a number of programs, in which the plant configuration data, as explained above, is input interactively by the user. The routine functions are carried out by the program MASTER, which contains CONFIGURATOR and CONSOLIDATOR. The former receives the data, unit models and materials models, and creates a file for the hazard identifi-
cation program. The latter receives from CONSOLIDATOR the hazard identification data and creates the hazard identification table.

The model generation program MODGEN receives unit model data from the user and automatically generates a set of rules. The unit models are saved in a library file MODLIB. Together with this there are the process material program PROCMAI and the materials of construction program MATCON. These programs are written in Fortran 77.

The main hazard identification program is called IDENTIFIER, and is written in Prolog. This receives the plant configuration data and generates as output the list of variable deviations associated with causes and consequences. The goals satisfied by IDENTIFIER are the individual causes and consequences of each deviation.

8.4.2 Deficiencies of the Computer Aid

We have shown that hazard identification involves many types of expertise and it is not supposed that fault propagation fully encompasses this.

Specific deficiencies of the program developed by Parmar and Lees compared to that of a manual HAZOP are outlined here. The largest and most difficult area to enhance is that of filtering/pruning the causes and consequences generated. This is probably the part of the HAZOP procedure that requires most expertise and by nature involves the use of hidden rules and heuristics. The team will think of causes and consequences and probably dismiss some of them without giving a verbal or written explanation. These cases will be 'cut and try' cases, but the ones that need most expertise are the cases that fall into the grey area between accepting a fault and dismissing a fault.

The key to the filtering process would appear to be the use of the heuristic that a fault is not pursued unless it has
a credible realization. This means that a specific realization of a fault must be identified. For example, a blockage in a pipe may be ignored unless there is a specific realization of it, a pipe freezing in winter say. Realizability is the key filter. Any program to fully simulate a HAZOP must contain in its knowledge base the heuristics used by the experts for such cases.

Other aspects not covered by the program are operability and maintenance considerations. The program is only suitable for continuous operations.

The above aspects are significantly the most expert parts of a HAZOP, and they are not easily represented using fault propagation equations.

It should be noted at this stage that this work is not intended as an automatic HAZOP. What it does demonstrate are the difficulties inherent in producing such a system together with the fact that expert systems are one way forward. It also demonstrates automatic fault propagation well, and with improvements could prove a valuable design tool.

8.4.3 Expert System Features

The work described on computer aided hazard identification has certain features of an expert system, although the method was not developed with this specifically in mind.

Firstly, the method has a knowledge base in the form of information in the plant configuration and in the unit and material model rules. Also, it has a control strategy in that there is a procedure for using the rules to obtain a hazard identification table. On the face of it there is no explanation facility. However, if we examine the results of a manual HAZOP it is evident that this is in fact an explanation of causes and consequences together with actions required. This is also the case with the program output, so no overt explanation facility is needed. There is no knowledge acquisi-
tion facility, but there is potential to write it into the system since the unit and material models provide a strong framework for improvements. Parmar and Lees state that the method has the potential to become the kernel of an expert system for hazard identification.

8.5 Expert Systems and Hazard Identification

As a result of this work, we have been able to establish the expertise of coarse and fine scale hazard identification, together with some problem solving strategies used by the experts. In this section we discuss the feasibility of expert systems for hazard identification, taking each in turn.

8.5.1 Expert Systems and Coarse Scale Study

It is difficult to foresee an expert system for coarse scale hazard identification. The subject area is vast, involving many aspects and any system developed would, to be cost effective have to be a general one covering all plants, both new and those being modified. It would be a difficult task to create a knowledge base to cover all eventualities. What is more practical and foreseeable is a number of mini-systems aimed at helping the design team in a few problem areas. Such systems would probably be forward chaining and would capture design expertise including awareness of hazards, their consequences, costs and probability. Possible application areas include a knowledge based safety check list which would ensure that all aspects of the plant have been covered, or systems for dealing with particular chemicals such as asphyxiants, or radiation hazards. At the present time however, a full expert system to encompass coarse scale hazard identification does not seem to be a feasible proposition, since the area is not restricted to one domain, it is very ill-defined and no formalised methods exist.

8.5.2 Expert Systems and Fine Scale

Problem aids to HAZOP have been discussed by Ferguson and
Andow (1986). This area holds more promise, but it still has limitations, as we shall outline below. Work already reported has shown this to be the case. We have discussed the specific area of computer aids for hazard identification, which produced a cause and consequence list using fault propagation equations. This, as we have pointed out, contains various expert system qualities.

If this work is to become a full blown expert system, development would have to take place in two directions. The first is to improve existing methodology in the performance of the tasks it is designed to do. The other is to add to it, or wrap around it, an expert aid to undertake the tasks such as filtering of the cause and consequence information and handling of aspects such as operability and maintenance. To create a complete knowledge based system for HAZOP presents a number of difficulties. The study by Ferguson and Andow (1986) has shed some light onto this concerning the knowledge itself. In our opening chapters to this work, it was pointed out that one of the main difficulties in building expert systems is the acquisition of expert knowledge. This seems to be a case in point.

One major difficulty in producing an expert system for HAZOP is that of the guideword 'other'. This guideword is intended as a 'clear-up' type of activity, i.e., any hazards that do not fit anywhere else in the table are put here. This could lead to a multitude of unforeseeable causes and actions.

Another difficulty is that a HAZOP is a very 'open-ended' study. By nature, a computer program is restricted by boundary limits, and often a HAZOP would need to go beyond these limits.

8.6 Summary

This study has investigated the problem of hazard identification, the expertise involved and the problem solving tools used. It is difficult at this time to envisage a complete
system for coarse scale hazard identification, but fine scale seems to show more promise, particularly after the work by Parmar and Lees.

The subject is one of great expertise including some classification and algorithmic expertise. For a HAZOP the expertise is in the form of knowledge of faults occurring, knowledge of fault propagation and knowledge of fault termination. This generates the deviations and there is then the expertise of assessing the hazards for realizability. One of the problems in identifying expertise is that a HAZOP is carried out by a multi-disciplinary team each with a responsibility for a different area.

Coarse scale hazard identification is concerned with the expertise of awareness of the hazards which may occur. The design team must be aware of all conceivable hazards, and one way of envisaging a system for this is in the form of a semantic net.
9 Fire Protection of Storage Tanks

This topic was the first to be studied in this work, and was used as a learning process for a relatively simple problem. However, work was carried out, and this is what is reported here. The following sections describe various aspects of storage tanks and fire protection measures available. The final section in the chapter describes a knowledge based Prolog program.

9.1 Overview

The largest amount of chemicals associated with a processing plant will be found at the storage facility. These may be at the plant site or in another remote area. Loss in storage through fires constitute a major loss sustained by the whole of the chemical industry, so loss containment is therefore very important. Much legislation exists for storage installations, together with many codes of practice and industry standards.

Siting and layout of storage is also an important feature. The fire at the Pemex installation, Mexico City in 1984 is a good case in point. Here, the storage facility was located in a highly populated area and over 350 people were reported to have died from the fire and the resulting BLEVES. The location of storage with respect to the process is also of concern since storage is more likely to be put at risk by a process. The two therefore are usually separated. The purpose of a storage facility is to smooth out fluctuations in the flows into and out of the process, and it is often held as an insurance. Storage is used to store reactants, intermediates and final products.

Flammable materials are stored as liquids or as liquefied gases. One objective of the design of a storage installation for such liquids is to minimize the risk of fire. This has several aspects which are to minimize the risk to personnel, to minimize loss due to the initial fire and to prevent the

1 BOILING LIQUID EXPANDING VAPOUR EXPLOSION
spread of the fire to other vessels and equipment. The implementation of these measures involves protecting the vessel and fire fighting.

In this work we are mainly concerned with the fire protection of flammable materials storage. The ICI/RoSPA LFG Code (1970) gives an account of fire protection of liquefied flammable gas (LFG) storage, and this code has been used extensively for the work developed here. Along with fire protection recommendations, the code goes into great detail and includes measures to minimize spillages, sizing and number of connections, and siting of the installation with respect to ground contours and prevailing winds.

Before going into detail on the aspects of storage covered here, it is worth describing some other aspects of storage tank fire protection.

9.2 Aspects of Storage

There are many different aspects to storage tanks, including design and construction, siting, safety protection measures, segregation and bunding. This section is an introduction to some of these aspects.

9.2.1 Segregation

The IP Refining Code (1974) gives a comprehensive account of the recommended distances for various types of storage. Factors accounted for include the type of tank and the material being stored. The main classification on which segregation of storage is made is on the flash point of the material. These are:

- Class A liquid with flashpoint below 22.8°C.
- Class B liquid with flashpoint between 22.8°C and 66°C.
- Class C liquid with flashpoint above 66°C.

The main distinction is between Class A/B and C liquids,
although some relaxation of fire protection measures is allowed to Class B liquids. Details of the recommendations of the code are considered later.

Also used for classification of storage areas, although not in this work, are electrical area classification zones. These are to be found in the ICI Electrical Installation Code (1972).

9.2.2 Bunds

Some storage tanks are surrounded by a bund wall or a pit. Bunds are recommended for most kinds of flammable storage, and they are generally made of concrete or earth. They are usually provided for atmospheric storage tanks and for refrigerated storage of liquefied gases, but not for pressure or semi-refrigerated storage.

They are used to contain liquid if the tank fails in any way so that the leak can be dealt with relatively easily. Thus, they are provided for relatively weak atmospheric storage tanks, but not for stronger pressure vessels. Often, low walls are used where full bunds are not necessary and this also provides protection against flammables reaching the tank from an external source, or the tank being damaged by site vehicles.

Class A/B liquids in atmospheric storage should have a full bund surrounding them and where there is a group of tanks in one bund, the capacity should be that of the largest tank, allowing for displacement.

9.2.3 Separation Distances

Minimum recommended separation distances have come from the ICI LFG code and the IP Refining Safety Code for this work. An alternative which may be allowed is the use of engineering estimates to set separation distances. These estimates may be based on the heat from burning liquid or the ignition
of a vapour leak.

9.2.4 Types of Storage

Hughes (1970) describes the various types of storage that are in common usage. Figure 9.1 shows some typical storage tanks and vessels. These are:

1. Atmospheric storage, which contains liquid at atmospheric pressure and temperature. They can be either fixed or floating roof type, and they are designed to withstand an internal pressure or a vacuum of 1 psig or below. Floating roof tanks can have various types of roof. In all cases the roof floats on the surface of the liquid and the roof can be a pan, an annular pontoon or a double deck type. A floating deck may also be used inside a fixed roof tank for keeping vapour loss down. This type of tank is shown in Figure 9.2.

The mechanical design of such tanks is well governed by standards, notably BS 2594:1955 for horizontal tanks and BS2654:1973 for vertical tanks.

2. Pressure storage, which contains liquefied gas under pressure at atmospheric temperature. Typical shapes for this kind of storage are horizontal cylindrical vessels and spherical vessels. They are more suitable for volatile materials such as gasoline.

3. Refrigerated pressure storage or semi-refrigerated storage, which contain liquefied gas under pressure at low temperature. Typical shapes for this kind of vessel include horizontal cylindrical pressure vessels and spherical pressure vessels. The later has certain advantages in that the surface to volume ratio is very low and about 88% of that of the former. This reduces heat leak. Also, the foundation structure is less complicated and the low temperature stresses are easily determined.

Such vessels are usually standard pressure vessels desig-
a) Horizontal Cylindrical Atmospheric Tank
b) Vertical Fixed Roof Atmospheric Tank (Coned Roof)
c) Vertical Fixed Roof Atmospheric Tank (Domed Roof)
d) Vertical Fixed Roof Atmospheric Tank (Large Dosed Roof)
e) Vertical Floating Roof Atmospheric Tank
f) Horizontal Low Pressure Tank
g) Vertical Cylindrical Hemispheroidal Tank
h) Spherical Pressure Vessel
i) Horizontal Cylindrical Pressure Vessel
j) Vertical Fixed Roof Refrigerated Atmospheric Tank (Domed Roof)
k) Vapour Dome Atmospheric Tank

FIGURE 9.1 Storage tanks and vessels
FIGURE 9.2 Fixed-roof tank with floating deck

- Manhole
- Gauge hatch
- Vent
- Guide pole
- Open eave
- Seal
- Floating roof
- Fixed roof supports
ned to withstand a higher pressure and their mechanical design is governed by BS5500: 1976 and the API Std 620.

4. Fully refrigerated storage which contain liquefied gas at atmospheric pressure and low temperature. Typically they are doomed roof flat bottom tanks and are essentially atmospheric vessels designed to a pressure below 1 psig.

5. Gas under pressure. These are again designed to the pressure vessel standard BS5500.

The hazards associated with these different kinds of storage are different. If a volatile material under atmospheric storage leaks, a slow evaporation will result. If a gas under fully refrigerated storage escapes, an initial flash off will occur followed by a slightly higher evaporation rate. Escape of a liquefied gas under pressure results in immediate flashing off of a large proportion of the gas followed by a slower rate of evaporation. This is usually the most serious case. It is therefore evident that the hazards involved vary widely, and so different fire protection measures are needed in each case.

9.3 Fires in Storage Tanks

Fire in storage facilities can be particularly hazardous because of the large quantities of material involved. There is therefore a need to design an adequate protection system, and there are various ways of achieving this goal. The subject as a whole is dealt with in detail by Vervalin (1985).

Fires in such installations can start in different ways, and examples includes a fire or explosion in the flammable atmosphere above the liquid in the tank, ignition of vapour clouds outside the tank, ignition of liquid overfill or spillage, lightning strikes or ancillary equipment failure. It is accepted that the most common cause is that of tank overfill, which can be due either to human error or instrument error. In such cases, flash back may occur thus spreading the fire.
Fires starting in tanks, which results in an explosion, may blow off the roof, but can usually be contained. Where a fire starts outside the tank the consequences can be very serious, since subsequent equipment failure may feed the fire. For example, a fire in a bund may cause a pump or pipework to fail resulting in the spillage of the tank contents. Lees (1980) quotes pipework as failing typically in just 10 to 15 minutes when exposed to a strong heat source. Heat radiation then becomes a problem causing neighbouring tanks to buckle.

There are many precautions that can be taken to prevent or contain potential fires. Specific measures include water sprays and foam, adequate pressure relief and to have mobile units available, but these measures will differ depending on the type of tank in use. Measures which are more passive, but equally important include good layout, separate bunds, and a minimum of pipework, flanges and valves in bunds. Burying pipes and putting pipes outside bunds also helps. Overfilling can be prevented by high level alarms, trips and good management generally. If a fire does occur in one tank, sprays should be activated to keep other tanks cool, and pumpdown of these tanks to a receiver is also recommended. The water and foam supplies and the pumps providing the fire fighting material should also be adequate with an active maintenance programme. Water is used for cooling and for fighting fires.

Water sprays and foam are often installed for fire protection and this is sometimes supplemented by fireproof thermal insulation and mobile water and foam units. Water is used to fight fire and to cool exposed surfaces, and foam is used to extinguish fires.

The next few sections deal with specific types of storage and the specific measures needed in each case. Figures 9.3 to 9.5 are some typical protection measures, showing a perimeter sprinkler, a deluge system and a sprinkler system for a storage sphere.
FIGURE 9.5 A sprinkler system for an LPG storage sphere
9.3.1 Atmospheric Storage

The effects of a fire on an atmospheric storage tank are that those parts of the tank not cooled by the liquid contents get hot and weaken, and the liquid in the tank is heated and and its vaporization is increased. An atmospheric vent is used in the case of fixed roof storage.

9.3.2 Pressure Storage

Fire effects on pressure storage are similar to that of atmospheric storage. Unwetted parts of the vessel weaken due to heat rise. The vessel may burst even if it is below its design pressure. As the heat rises in the liquid the relief valve will lift. Further temperature rise will cause the liquid to vaporise without further rise in pressure so long as the valve has enough capacity. If not, or if the valve fails the vessel may burst. Kletz (1974) describes this in detail.

Fixed water sprays are again effective in cooling surfaces. For cylinders and spheres water comes off the vessel at the centre so sprays are provided above and below. The water spray systems are usually activated by an automatic deluge valve triggered by fire sensors. Fireproof thermal insulation is sometimes provided. This reduces the rate of heat uptake from the fire, which gives more time for mobile fire fighting units to be brought in. It is usual for insulation to provide 2 hours protection before the vessel becomes seriously overheated. Depressurization is also used, in which the tank contents are dumped on the start of a fire to a sink elsewhere.

9.3.3 Refrigerated Storage

Again, the effects of fire in this case are similar to that of atmospheric storage, but here vaporization of the tank contents will occur much more rapidly. Protection is by fireproof thermal insulation and fixed and mobile water sprays. The insulation is usually cork with vapour sealant which burns
slowly. An alternative is steel-jacketed polyurethane or perlite.

It should be noted that although we are concerned mainly with flammable storage facilities, adequate protection for materials such as hydrogen, chlorine and ammonia should also be considered, particularly if they are stored in the vicinity of a flammable storage compound.

9.4 Flammable Storage - Codes of Practice

The work on fire protection of storage tanks has largely been based around the codes previously mentioned, that is ICI/RoSPA (1970) and the IP Refining Safety Code (1965). This section describes their recommendations in addition to the protection measures discussed above.

9.4.1 ICI/RoSPA 1970

The ICI/RoSPA code gives recommendations for the safe storage of LFG. Topics covered in the code include area classification, pressure relief design, piping, instrumentation, transportation and operations and training. Here we are mainly concerned with the section on fire protection, and this information, together with that given above will be used for deriving rules for fire safety in storage installations.

ICI/RoSPA give the following three goals of fire protection equipment:

1. To prevent explosion or sudden spread of fire that would cause a hazard to personnel

2. To minimize the loss of equipment and process materials involved in the initial fire

3. To prevent the fire spreading to adjacent storage vessels or process equipment
The section on fire protection in the code deals with all aspects of protection of LFG installations.

**Vessel Supports**

Vessels should be supported on concrete plinths or steel supports with fireproof thermal insulation. The supports themselves should be capable of withstanding at least four hours of exposure to an external heat source.

**Pipework**

The code suggests that discharge and drain lines should pass from the storage vessel through a remotely operated isolation valve. It is also suggested that fireproof insulation be applied to pipework which is in close contact with the vessel. Another problem is that liquefied gas can become trapped between two closed valves which may subsequently rupture the pipe. To alleviate this a pressure relief device should be fitted to all lines which can be isolated between closed valves.

**Hydrants**

Two fire hydrants are recommended at close proximity to the storage facility. They should be about 15 m from the vessels and close to roads. The goal in hydrant design is to be able to provide cooling to tank surfaces which could become engulfed in fire.

**Access**

Access to roads should be such that it is possible to reach two sides of the facility and for 50 m³ or more of storage, roads should have passing places available.

The final sections on fire protection in the code deal with tank off-loading facilities, bund drainage, fire control, water supply, electric cable protection, LFG transfer and
water spray and foams.

9.4.2 IP Refining Safety Code, Part 3 1965

This code deals with aspects of petroleum plant. Relevant to this work is the section concerned with bulk petroleum storage. Topics covered include tank spacing and bund walls, gas-freeing and cleaning, construction of tanks and ladders and hand rails.

As stated earlier, classification is done on the basis of material flashpoint. The code provides a comprehensive table of distances for bunds and tonnage per bund. This has been used to formulate rules in this work.

9.5 Problem Solutions

The solution to the fire protection of storage facilities problem has been tackled by writing a Prolog program. A classification is used where the type of tank and material being stored are considered first.

This type of problem is different from the other problems studied in that it is a kind of classification problem, but not the kind where we ask an expert to generate examples. It has been heavily worked on by various codes etc., and is a fairly cut and dried algorithmic problem.

The first part of this chapter was concerned with the literature available on storage tank fire protection. This has provided a basis of the type of knowledge needed in order to design a safe facility. The information is taken from expert accounts including literature and codes of practice. The next step in building an automated design program is to organise the knowledge into a form suitable for rule type representation. The exercise has also proved useful for the process of knowledge acquisition. In the section below the suggestions made by an expert are outlined. The next section discusses the formation of the knowledge for a Prolog program.
By reading the expert accounts on safe storage facilities it is evident that the nature of the storage needs to be established, including aspects such as the type of tank, the material being stored and the proposed layout of the tanks. The designer therefore has to be familiar with the kind of storage before proposing the fire protection measures necessary. For this, it was decided that an algorithmic diagram would be most suitable, in which a series of questions regarding the material and type of storage are asked indicating which kind of protection was needed. This also has the advantage of organising the knowledge from the codes of practice and literature into a readable form. The diagram is shown in Figure 9.6. The diagram was constructed from distinctions that were made in the codes and also by discussions with an expert. As can be seen from Figure 9.6, the first classification made is on material. Initially, non-hydrocarbons are identified, and then classifications made on the basis of flashpoint for hydrocarbons or tank type. Next to be considered are the amount being stored and tank layout. As a result of making these distinctions, measures for fire protection are suggested.

The knowledge used can be split up into three areas, that of knowledge from codes, engineering knowledge or expert knowledge. For example a 50 feet separation requirement is obviously that of a code, whereas '110% bunds' is knowledge that was acquired from an expert. The expert was also able to clarify certain points concerning the diagram. These are listed below:

Clarify use of words such as 'near'.

Questions such as 'Is there a another tank in the vicinity?' would be enhanced by stating 'Is there another tank in the vicinity near enough to cause a knock-on effect?'.
Figure 9.6 Algorithm for Storage Tank Fire Protection
1. fixed roof? yes
   yes: total tank capacity is 60,000 m³ per bund
   no: is the material LPG, LFG or kept above its boiling point?

2. 20% bunds required
   yes: floating roof
   no: total tank capacity is 120,000 m³ per bund

3. water cooling insulation? yes: fit vapour seal
   no: ground slope to pit (if insulated drench is O.K.)
      drain down spray is 0.2 U.S. gallons/ft²/min

4. is diameter of tank < 9 m?
   yes: 15 m minimum to a total of 8000 tons water
   no: are the tanks in a group?
      yes: > 15 m from inside top of bund of any adjacent group
      no: is diameter of larger tank/
           diameter of smaller tank/15 m, whichever is least

   6 m minimum separation

   15 m minimum separation
It can be taken for granted that an ignition source is present.

Bunds should be able to contain 110% of the largest tank. Nitrogen blanketing should be installed for class A hydrocarbons stored in quantities of greater than 100 m³.

The algorithmic logic diagram has proved useful in classifying the knowledge available and has shown the kinds of decisions that an expert and subsequently any knowledge-based system needs to make. The diagram has also shown that the problem is fairly well defined with clear-cut solutions available and that the basic structure of the problem is algorithmic. The decisions and interpretation of the algorithm may require the use of heuristics. However, it is thought that a knowledge-based system is appropriate because of the extensive amount of literature and expert accounts together with a wide variety of protection measures for different storage installations.

9.5.2 Rules and Heuristics

The rules used for the Prolog knowledge based program will be discussed later. Here, we detail the rules and heuristics that were acquired from an expert and from reading expert accounts. These rules, as opposed to the knowledge used in the program can be termed 'refining rules' and would be used to adjust the detail of any fire protection measures established. The knowledge was not used in the program, but it is foreseen that any improvements to the program would probably incorporate the rules. They were acquired through discussions with an expert, and from an expert account by Kletz (1985). The rules are listed below and have been stated in production system format:

Storage

If the rate of filling is high and batch sizes are large, then fit a high level alarm.
If bunds are to be installed, then fit drain valves or bunds and inspect weekly.

If a spillage occurs once in 5 years, then two protective devices must be installed.

If the level indicator measures weight, then fit a high level alarm that measures volume.

Liquid Overpressure

If the tank vent is to pass liquid, then fit the vent near to the edge of the roof and its top not more than 8" above the top of the walls.

If the vent is not large enough to pass the liquid inlet rate, then fit the tank with a hinged manhole cover and fit it to the roof near the tank wall.

Maintenance

If the tank is to be steamed out, then leave several hours to cool, with the manhole open.

If a floating roof tank has a rim fire, then personnel should not enter the tank to fight it.

If welding on a floating roof tank then, the legs should be flushed with water from the top.

Static Electricity

If the tank is floating roof, then it should be grounded using shunts every few metres around the rim.

If static electricity proves to be a problem then one or more of the following precautions can be taken :-

Nitrogen blanketing
Antistatic additives
Minimize pumping rates
Filters and other restrictions should be followed by a long length of straight pipe

In this case, expertise is needed in the selection of the appropriate method.

The problem is therefore capable of being represented in one decision tree, even if it is somewhat large. Therefore, a simple expert system type of program is valid. This is described below.

9.5.3 Prolog Rules

The rules listed below have been derived from two codes, that of the IP Refining Safety Code and the ICI/RoSPA Code. The rules have followed on from the algorithmic logic diagram, but with a greater level of detail. The rules listed are only a sample of the rules in the program. The remainder are given with the program listing in Appendix C. The first category is atmospheric storage with either fixed or floating roof tanks. Common conditions for this set of rules are 110% bunds, 60,000 m³ per bund (fixed roof) and 120,000 m³ per bund (floating roof) and water/foam sprays and mobile units.

If atm storage, fixed roof, tanks in a group
Then >15 m from inside top of bund of any adjacent group

If atm storage, fixed roof, dia <9 m
Then grouped up to 8,000 ton water
15m minimum separation

If atm storage, fixed roof, dia >9 m
Then separation 1/2 dia largest tank/15 m whichever is smaller

If atm storage, floating roof, dia >9 m
Then >15 m from inside top of bund to any adjacent group

172
If atm storage, floating roof, dia > 9 m
Then separation 1/2 dia largest tank/15 m whichever is smaller.

The next category is pressure storage. Common conditions here are 15 m to overhead power lines, pipe bridges, low pressure refrigerated storage bunds, 7.5 m to above ground power cables and pipelines, water spray, fire proof thermal insulation. Other details are given in the program listing given in Appendix C.

If pressure storage, ethylene
Then > 60 m from tanks to boundary/ignition sources

If pressure storage, c_three hydrocarbon
Then > 45 m from tanks to boundary/ignition sources

If pressure storage, c_four hydrocarbon
Then > 30 m from tanks to boundary/ignition sources

If pressure storage, other hydrocarbon
Then > 15 m from tanks to boundary/ignition sources

Other categories covered in the program include refrigerated storage and non-hydrocarbon storage.

9.5.4 Prolog Program

As stated earlier, the program is written in Prolog and runs on a Honeywell Multics DPS8. It is written in two parts, one consisting of the knowledge base and the other the control mechanism. On loading the main program, the knowledge base is consulted automatically. The system is not supposed to be an expert system, but it is a small knowledge based system that demonstrates a way forward for future work. Also, the program is not designed to encompass the whole area of fire protection of storage tanks. Both expert system development and a program to cover all aspects of storage would be time-consuming and are beyond the scope of the current work.
Some example consultations are shown below. These illustrate the kind of classifications made by the program. Classification is on the basis of the type of tank, including the type of roof, the size of the tank and layout of a group of tanks together with the material being stored. On the basis of this information, the program searches the rule base for rules which match the conditions. The system is forward chaining data driven and uses Prolog's matching and backtracking to satisfy the goals. For example, the first consultation listed below concerns that of pressure storage. The user has a pressurized tank containing a C_3 hydrocarbon. Prolog thus matches pressure and C_3 with the following rule that is contained in the rule base:

```
rule('pressure','c_three','at least 45m from tanks to boundary and ignition sources').
```

The remainder of the rules are concerned with all aspects of pressurized storage, including separation distances and water or foam sprays.
York Portable Prolog Release 2.1.

Please pass on any comments on this Prolog to RSM Kirkwood via Multics Mail.

?- KNOWLEDGE BASED SYSTEM FOR THE FIRE PROTECTION OF A STORAGE TANK

PRE-CONSULTATION OPTIONS
------------------------
start a consultation (s)
list help (h)
exit from the program (e)

Option...? s.

rules consulted.

Is the material stored a hydrocarbon? Answer yes(y) or no(n)...
y.

What are the conditions of storage?

Type atm, pressure, or refrigerated... pressure.

Is the material ethylene, C_3 or C_4? Answer by typing ethylene, c_three, c_four.... c_three.

If pressurized C_3 storage, then :-
----------------------------------
at least 45m from tanks to boundary and ignition sources

Other separation requirements are :-
------------------------------------
15m from tanks to buildings containing flammable materials
15m from tanks to road/rail tank filling points
15m to overhead powerlines and pipe bridges
7.5m to above ground power cables and pipelines
1/4 sum of diams of adjacent tanks to other pressure storage tanks
15m from low pressure refrigerated tank bunds and 30m from tank shells
The two fire protection options are:

- evenly applied water spray
- fire proof thermal insulation - insulation to finish 200mm above base

State which option you prefer...
Type a for water spray and b for fire-proof thermal insulation...

Option...?

a.
Is the ground beneath the vessel sloped away to a catchment area?
Answer yes(y) or no(n)...

n.
water application rate of 0.2gal/min/sq ft

Do you want another consultation?
Answer yes (y), or no (n)

y.

KNOWLEDGE BASED SYSTEM FOR THE FIRE PROTECTION OF A STORAGE TANK

PRE-CONSULTATION OPTIONS
------------------------
start a consultation (s)
list help (h)
exit from the program (e)

Option...?
s.

rules consulted.

Is the material stored a hydrocarbon?
Answer yes(y) or no(n)...

y.

What are the conditions of storage?
Type atm. pressure, or refrigerated...

atm.

Is the tank diameter 9m or less?
Answer yes or no
Is the tank roof fixed or floating?
Type fixed or floating...
fixed.

110 per cent bunds

15m minimum separation for tanks in one group from inside top of bund of any adjacent group

15m minimum separation for tanks to boundary or ignition source

fit drain valves to bunds

normally closed position

check weekly

no more than 80,000 cubic metres per bund

separation 1/2 diam larger tank,
or diam smaller tank/15m whichever is least

bunds

fit drain valves to bunds

normally closed position

check weekly

Do you want another consultation?
Answer yes (y), or no (n)
y.

?-

KNOWLEDGE BASED SYSTEM FOR THE FIRE PROTECTION OF A STORAGE TANK

PRE-CONSULTATION OPTIONS
-----------------------------
start a consultation (s)
list help (h)
exit from the program (e)

Option...?
s.

rules consulted.
Is the material stored a hydrocarbon? 
Answer yes(y) or no(n)... 
n.

Is the material stored as LFG? 
Answer yes(y) or no(n). 
n.

Ammonia, chlorine and hydrogen are covered in this program. 
Type in one of the above for information about its storage.

Is there a tank containing flammable liquid near enough to cause a knock-on effect? 
Answer yes(y) or no(n)... 
y.

For ammonia storage in a hazardous area, then:--
--------------------------------------------
15m minimum separation and cooling water spray

Do you want another consultation? 
Answer yes (y), or no (n) 
y.

?-
KNOWLEDGE BASED SYSTEM FOR THE FIRE PROTECTION 
OF A STORAGE TANK

PRE-CONSULTATION OPTIONS

start a consultation (s) 
list help (h) 
extit from the program (e)

Option...?
s.

rules consulted.

Is the material stored a hydrocarbon? 
Answer yes(y) or no(n)... 
n.

Is the material stored as LFG? 
Answer yes(y) or no(n). 
y.

Is the nonhydrocarbon soluble in water? 
Answer yes(y) or no(n) 
n.

Does the substance come under the category of methylamines?
Answer yes (y) or no (n)
y.

For nonhydrocarbons, then:

- at least 15m from tanks to boundary and ignition sources
- at least 15m from tanks to buildings containing flammable materials
- at least 15m from tanks to road/rail car filling points
- at least 15m from tanks to overhead power lines and pipe bridges
- at least 7.5m to above ground power cables and important pipelines
- one quarter the sum of the diameters of adjacent tanks between tanks and other pressure storage tanks
- at least 15m from tanks to bund wall of other low pressure tanks

The two fire protection options are:

- evenly applied water spray
- fire proof thermal insulation - insulation to finish 200mm above base

State which option you prefer...
Type a for water spray and b for fire-proof thermal insulation...

Option...?
a.

- Is the ground beneath the vessel sloped away to a catchment area?
  Answer yes (y) or no (n)... y.

- Is the pipework carried beyond the vertical projection of the vessel, with no pipe fittings beneath the vessel?
  Answer yes (y) or no (n)... n.

  water application rate of 0.2gal/min/sq ft

Do you want another consultation?
Answer yes (y), or no (n)

[Execution aborted]

[Leaving Prolog]
We stated earlier that the program was not intended to be a full expert system. The study was done to show how an expert system might go about tackling such a problem. Included in the features of an expert system are an explanation facility and a knowledge acquisition mode. Neither of these are present in the current system, but it is envisaged that it would be possible to enhance the program by adding these features. Also, the knowledge contained in the rule base is from codes of practice and takes no account of expert heuristics such as those listed above. Again, these could be added without too much difficulty.

9.5.6 Possible Application of ID3

The theory of ID3 has been discussed in Chapter 6. A decision was made to see if this type of problem could be handled by a classifier, and also to get an idea of the application areas of a classification system of the rule induction type.

For this, the author has written his own version of ID3 in FORTRAN77, on a Honeywell 'Multics' DPS8.

The system relies on a number of attributes being defined with suitable values together with a number of examples to represent the domain of storage facilities. Attributes chosen were conditions of storage, material being stored and the type of tank. These three attributes are the main ones considered when designing a fire protection facility for storage installations. Values for these attributes are listed below:

Table 9.1 Attributes and Values for Storage Installations

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conditions</td>
<td>Pressure</td>
</tr>
<tr>
<td></td>
<td>Refrigerated</td>
</tr>
<tr>
<td></td>
<td>Atmospheric</td>
</tr>
<tr>
<td>Material</td>
<td>Class A or B hydrocarbon</td>
</tr>
</tbody>
</table>

180
At this stage it is necessary to obtain some examples of storage tanks so that we can use ID3 to work out which of the above attributes contains the most information, and to do a tree representation. However, on investigation, it was found that storage tanks tend to have unique, individual solutions, resulting in no classification of the traditional kind. It was therefore decided not to carry the work further.

9.6 Summary

We have in this chapter shown a method of solving the problem of fire protection of storage tanks. The method involved the use of an algorithmic logic diagram to develop a forward chaining rule based program in Prolog. The Prolog program is not designed to be comprehensive system, but is intended to show application areas and improvements that can be made to it.

Also tried as a possible problem-solving approach was a FORTRAN77 based ID3 method.

Such systems could find a use for the process plant designer, if the designer has no indication of previous storage applications.

There is a lot of expertise in fire protection of storage facilities, but this expertise is well established, and an expert is not really needed. It was originally chosen as a simple problem for the first topic in the work. We have shown however that a rule-based approach is worthwhile, that there is some expertise involved and that there is a large amount of knowledge, including literature and codes which encompass both
rules and heuristics. In this topic the main source of rules has been codes of practice.
10 Emergency Isolation Valves

10.1 Introduction

Emergency isolation valve installation was introduced in section 7.2.2 as one of the candidate topics selected for further study. Three expert system tools have been used. Two of them, Expert-Ease and Ex-Tran7 are both classifiers, while Micro-Expert is an advice giving system. The basis for the work comes from a paper written by Kletz in 1975.

The first part of this chapter describes some background to emergency isolation valves, including some case histories, criteria for installing emergency isolation valves and some typical installations. The second part describes the methods by which the solution to the problem has been tackled.

10.2 Case Histories

There are a number of case histories where the installation of an emergency isolation valve would have averted a disaster. These are given by Kletz (1975), and a few are described below.

10.2.1 Pump Glands and Seals

Leaks from pumps in cold duty are not uncommon, and there have been many lucky escapes from fires which potentially were disastrous. In one such case, the plant above the pump in question was protected by a concrete wall. In another, the resulting fire could have caused disaster on a road, 35 feet away from the leak. Various companies report fires from leaks in pump glands and seals. One company reported 3 major leaks from cold ethylene pumps in one year. Causes of these leaks are typically failure due to stress corrosion of bolts holding the seals in position. In one of the cases, 90,000 cubic feet of vapour escaped in 20 minutes. In the other two, the leaks ignited near the pumps, probably due to static electricity.
In another example, the glands of a high pressure injector failed due to stud failure. An extensive fire resulted causing serious injury to personnel and plant damage. The leak was ignited by the plant furnaces 200 feet away. As a result of this disaster, the injectors were fitted with remotely operated emergency isolation valves, and they were relocated inside a steam curtain.

In 1972 a fire occurred as a result of a leak from a large hydrocarbon pump when the bearings gland failed. The fire burned for 6 hours, but was successfully contained. In this case a remotely operated emergency isolation valve was fitted, but for process use. The operating buttons for the valve were near to the pump, and therefore could not be reached by the operator.

10.2.2 Fittings and Valves

A leak of propylene occurred when the gasket of a level indicator failed. Measures to contain the leak, such as blow-down and increased cooling, failed and 20 minutes after failure the leak ignited causing considerable damage to the plant.

10.2.3 Drain Valves

Leaks from drain valves occur usually as a result of ice or hydrate formation. The most notable incident is that of Feyzin, France in 1966 in which 18 people were killed. It started with a blockage in the drain line from a propane storage sphere. The drain line open end pointed vertically to the ground below the sphere, and contained two valves. The procedure to isolate the tank was to open the valve next to the tank fully and control the draining operation with the second valve. The first valve could then be used to isolate the tank if the second valve gave trouble.

At Feyzin the operator drained on the first valve with the second wide open. The first valve blocked, so he opened it...
wider until the blockage gave way, resulting in a jet of propane being released onto the ground, rebounding in the operators face and knocking the handle off the valve. Attempts to replace the handle and shut the second valve failed and the resulting vapour cloud ignited 500 feet away, flashing back to the original sphere.

The disaster was made doubly worse by the inexperience of the fire crew, who assumed the relief valves would prevent overpressure, so did not provide any water cooling. The tank eventually ruptured, spreading the fire to nearby storage tanks.

10.3 When to Install An Emergency Isolation Valve

The case histories described above indicate the need for emergency isolation valves for isolating leaks before they ignite and cause injury and damage. Due to the cost of an emergency isolation valve they cannot be practically installed on every item of plant equipment which might leak. Also the installation of such a valve may in itself cause increased chances of a leak. Far better is to design the plant initially for leak protection rather than isolating after a leak has occurred. Emergency isolation valves should be installed when the chances of a leak are high and/or the consequences are serious. Kletz describes some situations to be considered, including the size and consequences of a leak:

1. The equipment is particularly likely to leak; for example, very hot or cold pumps.

2. The equipment is less likely to leak, but if it does a very large quantity of material will run out and there is no way of stopping it; for example, the bottom pump on a still containing more than, say, 50 tons of flammable liquid.

3. The equipment is less likely to leak, but if it does so the leak will be very large; for example, a very large pump.
Apart from using the above rules to decide whether or not to fit an emergency isolation valve, the particular conditions must be considered in order to decide if a leak will occur. These are considered in detail later. The following argument can also be used to help the decision.

If the cost of installing an emergency isolation valve is $5000 (1972 prices) and the loss resulting from a leak is $5 million and if the chance of a leak is greater than 1 in 1000 in the life of the plant, then it is worth installing an emergency isolation valve. However, data on the frequency of gland failure is not usually accurate or available, and Kletz argues that it is easier to estimate a piece of missing data then to estimate the answer to a whole problem. In this work, we have considered data on the conditions in a large plant rather than rely on data on the probability of a leak.

Figure 10.1 shows some possible approaches to emergency isolation valve installation on an existing plant.

10.3.1 LPG Storage Installation

On LPG storage vessels there should only be one outlet below the liquid level, protected by a remotely operated isolation valve. All other connections, such as drain points, sample points, instrument connections etc. should come after this valve. The figure (Figure 10.2) shows the arrangement recommended by ICI/RoSPA (1970).

10.3.2 Olefin Plant

Data from an olefin plant was used in this work, and is shown later on in the chapter, as given by Kletz (1975). The data takes the form of likely sources of leaks, material, its temperature and pressure, the inventory which will escape if there is a leak. Also considered is if an item is particularly likely to leak, based on industry wide experience. Kletz (1975) gives similar data for an aromatics plant.
FIGURE 101 Possible approaches to EIV installation on an existing plant

possible site for single emergency valve

alternatively, these valves could be motorised
FIGURE 10.2 General arrangement of piping to a spherical vessel
Generally speaking, emergency isolation valves are only fitted on the suction lines of pumps and other items. Delivery lines are usually protected by non-return valves.

10.4 Solutions to the problem

As described above, the data for solving the problem was taken from that given by Kletz for an olefin plant. The work indicates that it is better to decide on whether to fit an emergency isolation valve from examples already existing and from experience learnt from case histories. The data used is shown in Table 10.1. The entry 'A' indicates that the item is particularly likely to leak. Entry 'B' indicates that the equipment is less likely to leak, but if it does, a very large quantity of material will run out, and unless there is an emergency isolation valve there will be no way of stopping it. Various tools from artificial intelligence were used in solving this problem. These are described below.

10.4.1 Expert-Ease

The theory and background to Expert-Ease has been described in Chapter 6. It will be recalled that it is an example driven system, in which the examples are in the form of attributes with values and a class value.

Table 10.1 EIV Data for an Olefin Plant

<table>
<thead>
<tr>
<th>Item of Material</th>
<th>Temp</th>
<th>Press</th>
<th>Inventory</th>
<th>AorB</th>
<th>EIV</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equipment</td>
<td>°C</td>
<td>lb/sq.in.</td>
<td>tons</td>
<td>fitted?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Furnaces Naphtha</td>
<td>100</td>
<td>120</td>
<td>Small</td>
<td>-</td>
<td>Yes</td>
<td>Control valves used as EIV</td>
</tr>
<tr>
<td>Feed pump Naphtha (gassy)</td>
<td>15</td>
<td>140</td>
<td>70</td>
<td>B</td>
<td>No</td>
<td>No leak history cold</td>
</tr>
<tr>
<td>Item of Material</td>
<td>Temp</td>
<td>Press</td>
<td>Inventory</td>
<td>AorB</td>
<td>EIV</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>-------</td>
<td>-----------</td>
<td>------</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>bottoms pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fuel oil</td>
<td>210</td>
<td>30</td>
<td>100</td>
<td>A</td>
<td>Yes</td>
<td>close to auto ignition temp.</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sidestream distillate pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fuel oil</td>
<td>160</td>
<td>25</td>
<td>20</td>
<td>B</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bottoms pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fuel oil</td>
<td>210</td>
<td>30</td>
<td>60</td>
<td>A</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bottoms pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gasoline</td>
<td>85</td>
<td>20</td>
<td>25</td>
<td>-</td>
<td>No</td>
<td>Invent can be pumped</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bottoms pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fuel oil</td>
<td>220</td>
<td>30</td>
<td>1</td>
<td>A</td>
<td>No</td>
<td>Invent small feed isolable</td>
</tr>
<tr>
<td>distillate</td>
<td>180</td>
<td>25</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fuel oil</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>reflux pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ethylene</td>
<td>-30</td>
<td>270</td>
<td>30</td>
<td>A</td>
<td>Yes</td>
<td>Pump has leaked</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sidestream ethylene pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ethylene</td>
<td>-30</td>
<td>270</td>
<td>10</td>
<td>A</td>
<td>Yes</td>
<td>&quot; and ignited</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>reflux pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>propylene</td>
<td>40</td>
<td>250</td>
<td>50</td>
<td>B</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cold al methane exchanger</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>methane</td>
<td>-100</td>
<td>370</td>
<td>30</td>
<td>A</td>
<td>Yes</td>
<td>Isolate with hand + control valves, +EIV</td>
</tr>
</tbody>
</table>

190
<table>
<thead>
<tr>
<th>Item of Equipment</th>
<th>Material</th>
<th>Temp °C</th>
<th>Press lb/sq.in.</th>
<th>Inventory tons</th>
<th>A or B</th>
<th>EIV</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Methane</td>
<td>40</td>
<td>30</td>
<td>50</td>
<td>-</td>
<td>No</td>
<td>No</td>
<td>history</td>
</tr>
<tr>
<td>gas Ethylene</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>leak</td>
</tr>
<tr>
<td>compressor Propylene</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>rate</td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>low</td>
</tr>
<tr>
<td>Compressor Ethylene</td>
<td>-100</td>
<td>65</td>
<td>40</td>
<td>-</td>
<td>Yes</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>for</td>
<td></td>
<td>process</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>convenience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compressor Propylene</td>
<td>-40</td>
<td>40</td>
<td>100</td>
<td>-</td>
<td>No</td>
<td></td>
<td>as for</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>process</td>
<td></td>
<td>gascomp</td>
</tr>
<tr>
<td>Bottoms Light</td>
<td>90</td>
<td>150</td>
<td>5</td>
<td>A</td>
<td>No</td>
<td></td>
<td>Manual</td>
</tr>
<tr>
<td>pump gasoline</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflux Methane</td>
<td>-100</td>
<td>370</td>
<td>5</td>
<td>A</td>
<td>Valve</td>
<td></td>
<td>fitted</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>remote</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflux Ethane/</td>
<td>-14</td>
<td>350</td>
<td>5</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ethylene</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Converter Ethane/</td>
<td>0/150</td>
<td>350</td>
<td>-</td>
<td>A</td>
<td>Yes</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>ethylene</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflux Propylene</td>
<td>5</td>
<td>90</td>
<td>1</td>
<td>-</td>
<td>No</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflux Propylene</td>
<td>30</td>
<td>175</td>
<td>5</td>
<td>-</td>
<td>No</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflux Butylene</td>
<td>120</td>
<td>50</td>
<td>15</td>
<td>-</td>
<td>No</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflux Butylene</td>
<td>35</td>
<td>50</td>
<td>5</td>
<td>-</td>
<td>No</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Notes

The entry 'A' indicates that the equipment is particularly likely to leak.

The entry 'B' indicates that the equipment is less likely to leak, but if it does a very large quantity will escape.

It is immediately apparent from Table 10.1, reproduced from Kletz (1975), that the data for the olefin plant is very similar to that required by Expert-Ease. The attributes are equipment item, material, temperature, pressure, inventory, 'A' or 'B' and the classes are emergency isolation valve fitted and emergency isolation valve not fitted. This is a straight binary split. Typical values would be furnaces (equipment), naphtha (material), 100°C (temperature), 120 lb/sq.in. (pressure), small (inventory), don't care (A or B) and the class value is fit emergency isolation valve. This example, and others similar to it were fed into Expert-Ease.

The system was written on a version of Expert-Ease in the 'Alvey Starter Pack' running on an IBM PC/AT.

Results

The following 41 node rule was induced by Expert-Ease from the examples as given in Table 10.2:

Table 10.2  Expert-Ease EIV Rule

<table>
<thead>
<tr>
<th>Material</th>
<th>Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>naphtha</td>
<td>Equipment</td>
</tr>
<tr>
<td>furnace</td>
<td>fit,eiv</td>
</tr>
<tr>
<td>feedpump</td>
<td>null</td>
</tr>
<tr>
<td>bottomspump</td>
<td>null</td>
</tr>
<tr>
<td>sidestreampump</td>
<td>null</td>
</tr>
<tr>
<td>refluxpump</td>
<td>null</td>
</tr>
<tr>
<td>compressor</td>
<td>null</td>
</tr>
</tbody>
</table>
In this case, Expert-Ease has established that the attribute 'material' is the most discriminatory, i.e. the one
that yields most information. The next most discriminatory are the attributes 'equipment', and 'temperature'. The value 'null' means that the system does not have enough information in order to reach a decision.

The rule as stated can be translated into 'if-then' format. The rule would then look like (ignoring the null values):

IF the material is naptha AND the equipment is a furnace THEN fit an eiv

IF the material is fueloil AND the temperature is less than 215°C THEN fit an eiv ELSEIF the temperature is greater than or equal to 215°C THEN no eiv

IF the material is distillate fuel oil AND the equipment is a bottomspump THEN noeiv ELSEIF the equipment is a sidestream pump THEN fiteiv

IF the material is gasoline THEN noeiv ELSEIF the material is ethylene THEN fiteiv ELSEIF the material is propylene AND the temperature is less than 35°C THEN noeiv ELSEIF the temperature is greater than or equal to 35°C THEN fiteiv ELSEIF the material is butylene THEN noeiv ELSEIF the material is ethane
THEN  fiteiv

IF  the material is methane
AND  the equipment is a refluxpump
THEN  fiteiv
ELSEIF equipment is an exchanger
THEN  fiteiv
ELSEIF the equipment is a gascompressor
THEN  noeiv

On querying the system, the value of the most discriminatory attribute is asked for first. If the user types 'naptha', the value of 'Equipment' is then asked. Supposing the user typed 'furnace', then the system responds with the class value 'fiteiv'. If the response to the value of 'material' had been 'fueloil', then the system would have requested the value of the attribute 'temperature' before reaching a conclusion.

The answers given by Expert-Ease are correct when queried about an olefin plant. However, this is not much use when talking about chemical plants in general. Also a designer is not going to have much confidence in a system if it produces a result on the basis of material and equipment type alone. On paper, the designer would take into consideration many other factors, such as leak history, and type of hazard. It is therefore evident that the attributes need revision if the system produced by Expert-Ease is going to be of use to general plant design.

Expert-Ease has been able to fully represent the problem with the use of three of the original six attributes. This is fine from a rule inducing point of view, since Expert-Ease will use as few attributes as possible in order to represent a problem. The designer in the emergency isolation valve problem will use more attributes, relevant to his particular installation.

The emergency isolation valve problem tackled by Expert-Ease needs revision because :-
1. The examples do not fully represent the domain

2. Greater care is needed in choosing the attributes and examples

3. A more general rule needs to be induced

Revision of the Emergency Isolation Valve Problem

As a result of subsequent discussions with T.A. Kletz, the definition of the problem has been changed.

It transpired from these discussions that the table was in fact drawn up by two experts working together, and that in fact, some of the entries were the outcome of compromises.

The comment was also made that the problem was to retrofit rather than fit from new, and that modern practice is to fit in some cases where the decision at the time was not to fit.

The experts apparently started from rules rather than examples, and they have stated some rules explicitly. However, the creation of the examples does, nevertheless, advance the matter. The rules stated by the experts were mentioned above, and these rules have been reformulated by the author as follows:-

1. Equipment is particularly likely to leak. It is so (a) if it has a history of leaking or (b) it is pumping at extremes of temperature or (c) it is otherwise likely to leak.

2. Equipment is less likely to leak but if it does could release a large quantity with no means of stopping it.

3. Equipment is less likely to leak but if it does could release a very large quantity.

Rule 1 is considered to be a 'strong' rule.
There also appears to have been some 'hidden' rules which came to light after discussions with the expert. These are:

4. The material is more hazardous if it is above its autoignition temperature.

5. Fit an emergency isolation valve if it is convenient for process reasons.

6. Control valves may be treated as a kind of emergency isolation valve.

7. Do not fit an emergency isolation valve if there is another means of stopping the leak.

8. Do not fit an emergency isolation valve if the fluid is gas in a compressor.

A particular rule also seems to have some influence as a counter-rule. Thus a small inventory weights the decision against fitting an emergency isolation valve. This is referred to as a counter-rule. The rules above have been applied to the original table of data, and this is shown in Table 10.3. The attribute 'leak history' has been included as a separate column in the table, since it has not been included in the rules list above. The table shows that the criterion of history of leaks tends to be fairly dominant in that a history of leaks is associated with the decision to fit an emergency isolation valve, and no history of leaks with the decision not to fit one. This is also the case with the criterion of rule 2, i.e. that the equipment is less likely to leak, but if it does could release a large quantity with no means of stopping it. This attribute has been termed 'large hazard'.
<table>
<thead>
<tr>
<th>Item</th>
<th>History of leaks</th>
<th>Fit EIV</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Furnaces</td>
<td>-</td>
<td>Y</td>
<td>Rule 1(c) apparently, rule 6</td>
</tr>
<tr>
<td>Feed pump</td>
<td>N</td>
<td>N</td>
<td>Influence of counter-rule 1(a), rule 4</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>Y</td>
<td>Rule 1(a) + (b), rules 2 and 4</td>
</tr>
<tr>
<td>Sidestream pump</td>
<td>N</td>
<td></td>
<td>Rule 1(a) + (b) not satisfied, but influence of rules 2 + 4 perhaps</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>Y</td>
<td>Rule 1(a) + (b), rule 2</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>-</td>
<td>N</td>
<td>Rule 7</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>N</td>
<td>Rule 1(a) + (b) but overridden by rule 7</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>-</td>
<td>N</td>
<td>Rule 1(a) + (b) not satisfied</td>
</tr>
<tr>
<td>Reflux pump</td>
<td>Y</td>
<td>Y</td>
<td>Rule 1(a) + (b)</td>
</tr>
<tr>
<td>Sidestream pump</td>
<td>Y</td>
<td>Y</td>
<td>Rule 1(a) + (b)</td>
</tr>
<tr>
<td>Item</td>
<td>History of leaks</td>
<td>Fit EIV</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------</td>
<td>--------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Reflux pump</td>
<td>N</td>
<td>Y</td>
<td>Rule 2 apparently</td>
</tr>
<tr>
<td>Cold Al exchanger</td>
<td>Y</td>
<td>Y</td>
<td>Rule 1(a) + (b), partly rule 6</td>
</tr>
<tr>
<td>Process gas</td>
<td></td>
<td>N</td>
<td>Rule 8</td>
</tr>
<tr>
<td>compressor</td>
<td>-</td>
<td>Y</td>
<td>Rule 8, but overridden by rule 5</td>
</tr>
<tr>
<td>Compressor</td>
<td>-</td>
<td>N</td>
<td>Rule 8</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>N</td>
<td>Rule 1(a), but overridden by rule 7. Influence of counter-rules 2 and 4</td>
</tr>
<tr>
<td>Reflux pump</td>
<td>Y</td>
<td>?</td>
<td>Meaning of notes not clear. Does valve fitted refer to existing valve?</td>
</tr>
<tr>
<td>Convertor</td>
<td>Y</td>
<td>Y</td>
<td>Special case</td>
</tr>
<tr>
<td>Reflux pump</td>
<td>-</td>
<td>N</td>
<td>Rule 1(a) + (b) and 2 not satisfied</td>
</tr>
<tr>
<td>Reflux pump</td>
<td>-</td>
<td>N</td>
<td>&quot;</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>-</td>
<td>N</td>
<td>&quot;</td>
</tr>
<tr>
<td>Reflux pump</td>
<td></td>
<td>N</td>
<td>&quot;</td>
</tr>
</tbody>
</table>

Notes
* Refers to borderline case
Extremes of temperature taken as \(-30\) degC and \(>180\) deg C
<table>
<thead>
<tr>
<th>Plant Item</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>EIV fitted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Furnace</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Feed pump</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Sidestream pump</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Bottoms pump</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Reflux pump</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Sidestream pump</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Attribute No.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>EIV fitted?</td>
</tr>
<tr>
<td>--------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>------------</td>
</tr>
<tr>
<td>Reflux</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Cold Al</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>exchanger</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Process gas</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>*</td>
</tr>
<tr>
<td>compressor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>Compressor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y Y</td>
</tr>
<tr>
<td>Compressor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y * N</td>
</tr>
<tr>
<td>Bottoms</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>pump</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
<tr>
<td>Convertor</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>* Y</td>
</tr>
<tr>
<td>Reflux</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>pump</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Bottoms</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>pump</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>*</td>
</tr>
</tbody>
</table>

Key

<table>
<thead>
<tr>
<th>Attribute No.</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Is there a history of leaks?</td>
</tr>
<tr>
<td>2</td>
<td>Is it otherwise likely to leak?</td>
</tr>
<tr>
<td>3</td>
<td>Is it above the auto-ignition temperature?</td>
</tr>
<tr>
<td>4</td>
<td>Is it pumping at high temperature?</td>
</tr>
</tbody>
</table>
5  Is it less likely to leak, but if so a large quantity will escape with no way of stopping it? ('large hazard')
6  Is there an alternative way of stopping a leak?
7  Is the gas fluid in a compressor?
8  Is the fitting of an EIV convenient for process reasons?

Attribute values

Y  yes
N  no
*  don't care
As a result of establishing the above rules, the data for Expert-Ease was revised, and this is shown in Table 10.4. The attributes have been changed such that they are questions, with values being 'yes', 'no' or 'don't care'. It is believed that this new set of examples is representative of factors which are considered for any plant when the decision is made as to whether to fit an emergency isolation valve or not.

The data was input to Expert-Ease as before. The equipment item was not included as this is not an attribute that is particularly important when considering the choice. More important are temperature, alternatives and leak history etc. These and other attributes have been included in the data.

**Result**

Expert-Ease induced the following rule from the data given in table 10.5 :-

<table>
<thead>
<tr>
<th>Table 10.5 Revised Expert-Ease Induced Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>large hazard</td>
</tr>
<tr>
<td>yes : fiteiv</td>
</tr>
<tr>
<td>no : history</td>
</tr>
<tr>
<td>yes : fiteiv</td>
</tr>
<tr>
<td>no : otherwise likely to leak</td>
</tr>
<tr>
<td>yes : fiteiv</td>
</tr>
<tr>
<td>no : pumping at high temperature</td>
</tr>
<tr>
<td>yes : alternative</td>
</tr>
<tr>
<td>yes : noeiv</td>
</tr>
<tr>
<td>no : fiteiv</td>
</tr>
<tr>
<td>no : noeiv</td>
</tr>
</tbody>
</table>

This is an 11 node rule, and the example which includes the first compressor (14) contradicts the rule. Translated into 'if-then' format, the rule looks like :-

**IF**  the hazard is large
THEN fit an eiv
ELSEIF there is a history of leaks
  THEN fit an eiv
ELSEIF it is otherwise likely to leak
  THEN fit an eiv
ELSEIF it is pumping at extremes of temperature
  AND there is an alternative way of stopping the leak
  THEN no eiv
ELSE fit an eiv
ELSEIF it is not pumping at extremes of temperature
  THEN no eiv

The attribute 'large hazard' is defined as 'it is less likely to leak, but if it does so, could release a large quantity with no way of stopping it'. Expert-Ease has established that this is the most discriminatory attribute, and on querying the system the user is asked whether there is a 'large hazard'. If the answer is 'yes', then a conclusion is reached, i.e. fit an eiv. If the answer had been 'no' then the user is asked the value of the next most valuable attribute, that of leak history, and so on down the tree.

This version of the rule appears to be much more useful than that previously induced. It is also more general and is applicable to many cases. On testing the system, the rule worked well in the face of the data available.

10.4.2 Ex-Tran7

The theory of Ex-Tran7 was explained in section 6.2.5. It works in a similar way to Expert-Ease, but uses an enhanced version of ID3. Also, the facilities available to the user are more comprehensive than Expert-Ease. The system starts off in a similar vein to Expert-Ease, with the user defining attributes, values and classes, then typing in the example set. In this case, the same set of examples was used, i.e. the data in Table 10.4. Once a rule has been induced, the user can enhance the explanation facility for user consultations.
The system was written on Ex-Tran7 running on a DEC Micro-VaxII under the VMS operating system.

Ex-Tran7 induced the following rule :-

Table 10.6 Ex-Tran7 Induced Rule


In 'if-then' format, the rule looks like :-

IF the hazard is large
THEN fit an eiv
ELSEIF it is otherwise likely to leak
THEN fit an eiv
ELSEIF there is a history of leaks
AND an alternative way of stopping the leak
THEN no eiv
ELSE fit an eiv
ELSEIF there is no history of leaks
THEN no eiv

As can be seen, the rule is slightly different to that induced by Expert-Ease. The reason for this is the conflicting example, i.e. the example that contradicts the Expert-Ease rule. In Ex-Tran7, conflicting examples are put into a secondary dormant store, and are taken into account when a rule is induced.

Again, the rule works well in the face of the data available, and has again shown that induction works well when the
examples used specifically cover the problem domain.

Listings from Ex-Tran7 for the emergency isolation valve problem are shown in Appendix D.

10.4.3 Emergency Isolation Valve Advice System

One of the main advantages of rule induction systems such as Expert-Ease and Ex-Tran7 is their ability to induce concise and reliable rules in the face of a large amount of seemingly random data. Although rule inducers can be written in an advice-giving mode, it would seem better to use a regular advice system for routine advice work. For this reason, we have now turned to Micro-Expert, the theory of which was explained in section 6.4.1.

The rules developed by the above two systems have been converted into Micro-Expert advice language to produce an efficient system, which allows the use of probabilities and has been built with the best information available.

This aspect of the work, therefore, is an example of the transfer of rules from a rule induction system to an advice system. The use of an advice system then makes it possible to introduce probabilities which allow both for the strength of the rules and uncertainty in the values of the attributes.

It is worth indicating at this stage how the classification rules were coded into advice-giving rules. The rule automatically induced from Ex-Tran7 was used. The Micro-Expert user manual suggests that in order to write the advice language code, a diagram is needed. This is shown in Figure 10.3. The model is drawn in diagrammatic form. Each block in the diagram represents a hypothesis which is given a name shown in the top left hand box of the block. For example the block at the top of the diagram is called EIV and represents the hypothesis "An EIV is needed". This hypothesis is a goal hypothesis because it is the hypothesis which the model is designed to prove or disprove. The hypotheses HISTORY, HAZARD, TEMP and
FIGURE 10.3 EIV Block Diagram
ALTER are of the type QUESTION.

Also associated with the hypotheses are the logical sufficiency and logical necessity factors. In the diagram, HAZARD is assigned an LS factor of 100 and an LN factor of 0.01. This means that the definite presence of 'large hazard' makes the fitting of an emergency isolation valve 100 times more likely. The definite absence of 'large hazard' makes the fitting of an emergency isolation valve 100 times less likely (i.e. \(1/0.01\)).

The code for the system, together with the explanation of how it works, appears in Appendix E. Here, we show the results of some sample consultations.

The system is running on a 'Torch' micro-computer running under the UCSD PASCAL operating system. After compiling the code successfully using 'EXPCOMP', the system is run using the program 'RUNEXPT'.

The following is an example of the output, user input is shown in bold type :-
Start of consultation 1.

"The current goal is whether or not
An eiv is needed
Certainty factor is 0.00 Certainty range is -5.00 to 5.00

How certain are you that the hazard is large [-5..0..5]? 5

The current goal is whether or not
An eiv is needed
Certainty factor is 4.90 Certainty range is 0.00 to 5.00

Is it true that there is an alternative way to stop a leak [Y..!..N]? N

The current goal is whether or not
An eiv is needed
Certainty factor is 4.90 certainty range is 0.00 to 5.00

Is it true that the item is pumping at extremes of temperature [Y..!..N]? Y

The current goal is whether or not
An eiv is needed
Certainty factor is 4.99 Certainty range is 4.90 to 5.00

How certain are you that the item has a history of leaks [-5..0..5]? 3

This goal was whether or not
An eiv is needed
Certainty factor is 5.00

End of consultation 1.

Here, the user is certain that the hazard is large, i.e.,
the item is not particularly likely to leak, but if it does, a large quantity will be released, the item is pumping at extremes of temperature, and he is reasonably certain that the item has a leak history. Micro-Expert has therefore concluded (i.e. the certainty factor is 5) that it is certain that an emergency isolation valve is needed.

Start of consultation 2.

"The current goal is whether or not :-
An eiv is needed
Certainty factor is 0.00 Certainty range is -5.00 to 5.00

How certain are you that the hazard is large [-5..0..5]? -5

The current goal is whether or not :-
An eiv is needed
Certainty factor is -5.00 Certainty range is -5.00 to 4.80

Is it true that the item is pumping at extremes of temperature [Y..!..N]? N

This goal was whether or not :-
An eiv is needed
Certainty factor is -5.00"

End of consultation 2

In this case, an emergency isolation valve is definitely not needed, since hazard is certainly not large, and the item is not pumping at extremes of temperature - as indicated by the certainty factor of -5.

Start of consultation 3.

"The current goal is whether or not :-
An eiv is needed
Certainty factor is 0.00 Certainty range is -5.00 to 5.00
How certain are you that the hazard is large [-5..0..5]? 3

The current goal is whether or not :-
An eiv is needed
Certainty factor is 4.84 certainty range is -1.23 to 5.00

Is it true that the item is pumping at extremes of temperature [Y..1..N]? Y

The current goal is whether or not :-
An eiv is needed
Certainty factor is 4.98 Certainty range is 4.84 to 5.00

How certain are you that the item has a history of leaks [-5..0..5]? -2

This goal was whether or not :-
An eiv is needed
Certainty factor is 4.97

End of consultation 3.

In this case it is certain that an emergency isolation valve be fitted.

Start of consultation 4.

"The current goal is whether or not :-
An eiv is needed
Certainty factor is 0.00 certainty range is -5.00 to 5.00

How certain are you that the hazard is large [-5..0..5]? 0

The current goal is whether or not :-
An eiv is needed
Certainty factor is 0.00 Certainty range is -4.90 to 4.52

Is it true that there is an alternative way to stop a leak [Y..1..N]? Y
The current goal is whether or not: -
An eiv is needed
Certainty factor is -4.90 Certainty range is -5.00 to 4.52

Is it true that the item is pumping at extremes of temperature [Y..1..N]? N

The current goal is whether or not: -
An eiv is needed
Certainty factor is -4.99 Certainty range is -5.00 to 3.53

How certain are you that the item has a history of leaks [-5..0..5]? 0

The current goal is whether or not: -
An eiv is needed
Certainty factor is -5.00

End of consultation 4.

In this case it is certain that an emergency isolation valve is not needed.

Start of consultation 5.

The current goal is whether or not: -
An eiv is needed
Certainty factor is 0.00 Certainty range is -5.00 to 5.00

How certain are you that the hazard is large [-5..0..5] 0

The current goal is whether or not: -
An eiv is needed
Certainty factor is 0.00 Certainty range is -4.90 to 4.52

Is it true that there is an alternative way to stop a leak [Y..1..N]? N

The current goal is whether or not
An eiv is needed
Certainty factor is 1.67 Certainty range is -4.95 to 4.80

Is it true that the item is pumping at extremes of temperature 

The current goal is whether or not
An eiv is needed
Certainty factor is 3.52 Certainty range is -1.56 to 4.80

How certain are you that the item has a history of leaks [-5. .0. .5]? 1

This goal was whether or not :-
An eiv is needed
Certainty factor is 4.33"

End of consultation 5.

In this case an emergency isolation valve is almost certainly needed.

Start of consultation 6.

The current goal is whether or not :-
An eiv is needed
Certainty factor is 0.00 Certainty range is -5.00 to 5.00

How certain are you that the hazard is large [-5. .0. .5]? -1

The current goal is whether or not :-
An eiv is needed
Certainty factor is 4.76 certainty range is -2.11 to 5.00

Is it true that there is an alternative way to stop a leak 

The current goal is whether or not :-
An eiv is needed
Certainty factor is 4.76 Certainty range is -4.96 to 5.00

Is it true that the item is pumping at extremes of temperature [Y..I..N] N

The current goal is whether or not :-
An eiv is needed
Certainty factor is -4.61 Certainty range is -4.96 to 3.02

How certain are you that the item has a history of leaks [-5..0..5]? 2

This goal was whether or not :-
An eiv is needed
Certainty factor is 1.22"

End of consultation 6.

In this case it is likely that an emergency isolation valve is needed.

10.5 Discussion

This chapter has shown one way of building an advice giving expert system. The original data for deciding whether or not an emergency isolation valve is needed was used as an example set for a rule induction method. The data needed subsequent revision, but eventually a rule was induced that appeared to completely cover the domain. Two aids to building an expert system were used for this. The first, Expert-Ease induced a similar but longer rule than the enhanced version, Ex-Tran7. Both methods rely on an example set with a certain classification, in this case a binary split between fitting an emergency isolation valve and not fitting one. The expertise required therefore is of a classification type. The induction methods were able to decide which of the attributes representing the examples provided the most information in order to reach a decision. With this in mind, we can infer that the rule induced is the best possible in respect of the data
The method has shown a certain disadvantage of induction methods. Great time and care is needed in choosing the attributes for the examples to fully represent the domain in question. Our original example set did not work well under an induction method, and reasons for this are given in the text above. However, when time had been spent with an expert, new 'hidden' rules were found to be useful in re-representing the problem. The method also highlighted which rules were strong, or most influential.

Although it appears that induction methods induce only one rule, this is not quite the case. The rule can be 'split' into a number of production style rules, although the order in which they are fired must be kept the same. The induction method has provided us with a set of rules, in which we have confidence and which we then used in an advice giving system.

The shell Micro-Expert was used in order to produce a true advice giving system. The rules that had been induced by induction were used for this purpose. The shell requires that the rules be written in Micro-Expert advice language. This is a straightforward procedure which allows probabilities together with sufficiency and necessity factors to be assigned. The rules induced are thus useful in deciding what these factors should be, since an indication of the strength of the rules is given.

The attribute 'large hazard' is the most discriminatory of the attributes that the rule induction system dealt with. This therefore is a strong rule with respect to fitting an emergency isolation valve, and Micro-Expert allows us to exploit this with the use of a logical sufficiency factor. Hence in the advice giving system 'large hazard' has the highest sufficiency factor, i.e. 100. Micro-Expert also allows us to deal with counter rules, i.e. those rules that come out strongly against fitting an emergency isolation valve. In this case the attribute 'alternative' is a counter rule, and for this we
assign a low logical sufficiency factor and a high logical necessity factor, i.e. 50. The other factors for the advice system have been assigned on the basis of their 'placing' in the induced rule. This has indicated a useful property of an induced rule. The rule indicates the value of the information gained by using a particular attribute, which is essential when the rule is used in a practical way, such as in an advice giving system.

We have shown in this study an example of the development of a successful advice giving system derived from classification rules.
11 Flare System Design

11.1 Introduction

In Chapter 7 the problem of flare system design was introduced as one of the candidate topics thought to be worthy of further study. Here the problem is expanded and a design strategy is suggested for its solution. The opening sections of this chapter give accounts of the general aspect of flaring together with a review of the literature. Rules have been derived in production system format together with a more precise outline of the design process. A production system program, written in Prolog, for some aspects of the design is then described.

11.1.1 The Need for a Flare

When plant equipment malfunctions, a rise in pressure is not uncommon. These pressures are relieved to an adequate relief disposal system before the pressure levels reach unacceptable limits. If the material being used is flammable, then it is generally sent to the flare relief system to be burned off, and the safety of the plant is then dependent on a reliable relief disposal system. Such a system must therefore be able to cope with a whole variety of conditions and materials.

A flare system therefore is needed due to overpressure caused by plant upset conditions. What are the causes of overpressure? This will be discussed in more detail in Chapter 12, but it is worth noting some causes here:

- Loss of cooling water
- Power failure
- Failure of heat exchanger tubes
- Control system failure
- Chemical reaction runaway
11.2 General Aspects to the Design of a Flare System

Reference was made in Chapter 7 to some of the components of a flare system. These included a knock-out drum, seals and flaring hazards. In this section more detail is outlined on these and other aspects.

A typical flare system will consist of a relief header, a stack, a knock-out drum, a flare tip and usually some sort of seal or other device for preventing flashback down the flare. The main hazards encountered in a flare system are the ingress of air leading to explosion, a blockage, heat radiation, liquid carryover, low temperature embrittlement and the problem of toxic gases. Related problems include smoke, noise, glare, land sterilization and positive ignition.

11.2.1 Physical Design

The physical design of a flare stack is well documented, such as the API RP 520 and 521. Here a brief description of the sizing of a flare stack is given. The process starts with basic physical data and includes expected gas flowrate through the flare, average molecular weight, gas temperature, gas pressure at the tip and the expected wind velocity at the tip. The diameter of the flare stack is calculated on the basis of designing for a velocity Mach Number of 0.2. The flame length is calculated using the flame models, which take account of flame distortion caused by wind velocity. The flare stack height required is based on the heat liberated, the fraction of heat radiated and the maximum allowable radiation 150 feet from the stack. The API suggest a 25% increase on the design height from that calculated. The final procedure is to calculate the maximum concentration of gas at grade. This will obviously depend on the type of gas being flared.

In this work we are concerned with the procedure for designing a flare stack together with the choices and conflicts for the designer. For this, each aspect of a flare system is described below. This is termed an 'awareness aid'.
11.2.2 Flare System Awareness Aid

Boeiji (1979) discusses flare relief systems in general and he gives some typical flare installations together with commercially available ground flares, flare system control, burners and flashback protection. Another paper concerned generally with flaring comes from Bluhm (1985) whose discussion includes hazards, knock out drum sizing, low temperature flares and maintenance.

Elevated Flares

Elevated flares are usually used in conjunction with a ground flare. They are used for infrequent emergency conditions and are typically designed to handle 1,200,000 lb/hr in the molecular weight range 25-40. Figure 11.1 shows a typical elevated flare installation.

Ground Flares

Ground flares are used for burning normal excess emissions from a plant during the day to day running. This ensures that the elevated flare is ready in the case of an emergency condition. Their main advantage is that they burn quietly and only harmless invisible combustion products are discharged to the atmosphere. The burning occurs inside a special refractory lined enclosure. Generally, excess gases which are not recoverable go to the ground flare. Relieved gases come from the plant and hit a pair of liquid seals. When an emergency occurs the gases pass through the upper liquid seal and burn at the elevated flare. A typical flowrate of a ground flare is 70,000 lb/hr and gases of molecular weights in the range 20-40 are normally burned. Their installation is dependent on local conditions, including environmental. Figure 11.2 shows a typical ground and elevated flare installation.

Knockout Drum

A knockout drum is used to prevent hazards associated
FIGURE 11.2 Typical flow scheme of a ground/elevated flare installation
with burning liquid droplets escaping from the flare stack. The drum must be of sufficient diameter to effect the desired vapour-liquid separation. The capacity of the drum should provide a liquid hold up time of 10 to 30 minutes. Generally the diameter of the drum is 1/2 to 1/3 the length of the drum and is 3 to 4.5 times the diameter of the flare stack. A high and low level alarm should be fitted together with an automatic pump out to the slops system. Seals are required to stop air entering the header since after a release the header cools, and air could be drawn in.

Sizing is a complex task, and in doing so the type of process equipment connected to the flare, the maximum rates at which liquid can enter the system and the time operators need to correct a system upset must be considered.

**Water Seals**

Water seals are used to prevent flashback into the flare headers. The water provides a seal between the header and the outside atmosphere. Their design is based on the maximum quantity to be released. They should offer minimum resistance to flow during emergency conditions. Where freezing temperature conditions are a problem, water seals are not recommended.

**Quench Drums**

Quench drums are used to cool liquid streams that may be released at high temperature. Their sizing depends on the design of the drum internals as well as the amount of heat that must be removed. Generally, the exit stream temperature must be reduced to a range 150 - 200°F, assuming that the inlet stream will be 40 - 50% vaporized.

**Molecular Seals**

Molecular seals are installed below the flare at the top of the riser and will stop the entry of air into the stack.
They are very prone to choking, particularly from carbon from incompletely burned gases. For this reason, companies are generally not using such seals.

Flame Arresters

Flame arresters will protect against flashback as long as the flame arresting elements can withstand the heat released from excessive combustion. They are, however, subject to fouling that cause large pressure drops during periods of high flow and may cause dangerous overpressure on connected equipment. One view is that there are only two instances were their use is justified:

1. If the gases being vented can decompose without the addition of air.

2. In vented pipes of storage tanks containing a flammable mixture of vapour and air.

Purging

Purging of inert gases is used as an effective way to stop air entering the relief system when it is not in use. A purge ensures safe operation and protects against explosion and detonation. It can, however, be expensive. A purge of flue gas is estimated to cost about $7,500 per year (1986). Generally, a purge rate adequate to produce a flame visible from the ground in daylight is safe. One company recommends that purge should be continuous with a minimum flowrate of 236*flare diameter m^3/hr. The purge gas is usually either nitrogen or flue gas. Not all refiners use such a system, but it seems to be one of the most effective ways of keeping air out of a stack.

Igniters and Pilots

Continuous pilot burners on the flare tip ensure ignition. The pilot burners themselves are commonly ignited by a
flame front generator system. Ignition is then guaranteed regardless of wind direction by placing three continuous pilots around the flare tip. Thermocouples on the pilot activate an alarm system to warn of pilot flame failure.

**Noise Levels**

The roar of the combustion process is the most serious problem in an elevated flare stack. Sources of noise include steam injection, moisture condensation shock, seal drum sloshing and low flow instability. It is difficult to deal with combustion roar, but one solution is to burn the most moderate and frequent releases at a ground flare, keeping the elevated flare for emergencies. Steam injection will add to the combustion roar as well as providing its own higher frequency roar. The noise can be controlled by mixing the steam with the inspirating air more quickly.

**Smoke**

Flares can be made smokeless by providing an adequate quantity of oxygen in the combustion zone. Factors to consider in designing a smokeless flare are:

1. The quantity of oxygen in the combustion zone.
2. The temperature of the combustion zone.
3. The type of hydrocarbon being burned

For smokeless burning of a paraffin approximately 20% of the stiochiometric quantity of air must be evenly distributed in the primary mixing zone. For an olefin, this quantity is 30%.

It is an offence under the Clean Air Acts 1956 and 1958 to emit 'dark smoke' or 'black smoke' except at certain times.

**Flare Tips**

There are three generic designs of flare tips available. They are the utility flare tip, the centre steam flare tip and
the steam ring flare tip. The choice of which to use is dependent on the materials being burned and the type of burning required. They are susceptible to failure due to cracking or the burning out of internals.

**Heat Radiation**

The API RP 520 states that the following heat radiation limits should not be exceeded in the area around the flare stack:

- 1.7 kW/m² for exposure of the general public or for continuous working.
- 5 kW/m² for short time exposure of personnel
- 13 kW/m² for control room externals

In calculating the distance required between the nominal flare centre and a point of exposure, the fraction of heat radiated, the allowable radiation, the flare rate and the net calorific value must be considered.

**Controlled Blowdown**

Controlled blowdown is a method whereby the relief loads are smoothed by staggering the operation of pressure relief valves. It has the advantages of saving money, reducing noise and smoke levels and space. The effect of blowdown is to reduce the peak flow during flaring operations (although the total amount flared is the same with or without controlled blowdown) thus reducing the size of the stack. However, it does require a much more thorough design. The system is particularly well suited to offshore platforms where space is short. Figure 11.3 shows graphically the advantages of controlled blowdown.

**Atmospheric Relief**

Many relief valves are routed directly to the atmosphere.
Graphical representation of controlled blowdown showing how blowdown can be controlled to avoid initial peak flow.
Often they are used in conjunction with a flare system. Only light hydrocarbons and non-toxics are disposed of in this way, but it is a good way of reducing the header and relief requirements.

11.2.3 Hazards in Flaring

Flaring is by nature a hazardous operation because there is an open line to the atmosphere with a continuous ignition source present. The presence of a flare system implies that there are hazardous unwanted materials to be disposed of. Therefore any such system must be inoffensive to the surroundings with respect to noise, light and pollution. In this section the main hazards will be detailed together with methods of eliminating them.

Oxygen Intrusion

For an explosion to occur, a flammable material, a source of ignition and oxygen must be present. All three are present in a flare system if precautions are not taken to prevent this. It is possible for air to enter the stack through a variety of ways including open vents or drains, flare lines, start-up or rapid condensation of hot vapours which can suck air back down a line. Precautions against this happening have been outlined in the awareness aid, and include purging, flame arresters, or molecular seals. These are only precautions and are not a fool-proof technique. It is better to prevent air entering the system by stopping it in the first place by eliminating air in-leak into the pipework, and purging equipment to atmosphere rather than the stack during start-up.

Liquid Carryover

This is usually prevented by using liquid knock-out drums as described above. Their purpose is to separate large quantities of liquid that accompany vapours going to the stack. If liquid does get to the flare, then the flame becomes more smoky and burning droplets of liquid disperse.
Blockages

Flare system obstructions can occur as a result of seals blocking or freezing. To prevent this, flare lines are usually sloped to the knock out drum and are free of pockets. Knock-out drums themselves are often protected against freezing. Oil may also plug the system by congealing at the temperatures of the flare stack. The only cure for this is elimination.

Low Temperature Embrittlement

When liquids of low boiling point are being flared, then complete flare line failure is a possibility resulting from low temperature brittle fracture. It can occur from low transition operating temperature, notches, cracks and stress. If the system is operated at a temperature below the transition temperature of the material from which the flare is made, then brittle failure may occur. Special materials of construction are used to prevent this, and notches and cracks are alleviated by stress relieving.

Maintenance

Whilst maintenance is in progress, flammable or toxic material may be released. The risk is increased because it is often necessary to open flare lines without gas freeing. The most common problem seems to be asphyxiation of personnel whilst opening lines for maintenance.

11.3 An Account of Flare Literature

There is an extensive literature concerning flare systems. This is largely in the form of technical papers, but some codes of practice are available. Some 100 papers have been collected for the purpose of acquiring knowledge, rules and heuristics together with some ideas as to how a flare system is designed. In this section some of the more notable papers are described.
11.3.1 General

The American Petroleum Institute (1986) give an extensive account of the safe operation of refinery flares and the precautions necessary. The following quote, applicable to this work comes from the API:

"A flare system does not lend itself to a standard design or to uniformly applicable procedures."

This comment supports the idea that flare system design is an expert matter.

Reed (1968) talks of the importance of flare stacks being able to cope with upsets, keeping air out of the system and purge rates.

Kneale (1984) is concerned with the engineering design of relief systems. Aspects include system requirements, total system hardware, data and other factors including dispersion and emission, noise, and scrubbing. Kneale also describes the design process as:

1. Defining the system
2. Sizing the device
3. Designing the hardware
4. Treating the products of relief

Klooster et al. (1975) discuss flare system design optimization and factors in design which can lead to a safe disposal system.

Brzustowski (1977) is concerned with heat radiation, flame shape and radiation.

11.3.2 Specific Aspects

Many of the works available describe more specific aspects of a relief system design. These include explosions, smoke, noise and radiation control.

229
Explosions

The paper by Kilby (1968) is specifically to do with explosions in flare stacks. He describes four incidents showing how different conditions can lead to an explosion. These include air contamination during maintenance, entry of combustible material prior to air freeing and contamination by reactive materials. Peterson (1967) describes experiments designed to control the hazard of flare stack explosions. In particular, he discusses an explosion suppression system.

Environmental

Seebold (1971 and 1972) deals with the specific aspect of flare noise. The noise from flares comes mainly from combustion and steam injection. Steam noise seems to be dependent upon the type of injection nozzle used. Miller et al. (1958) and Agar (1978) discuss the problem of smoke from flares. Steam and water injection help to alleviate the problem, and smoke can also be reduced by using a multi-jet burner. Steam does, however, cause noise, so there will inevitably be a trade-off between the two, and is therefore an expert matter.

Heat radiation is also the topic of many papers, and works by McMurray (1982), Kent (1968) and DeFaveri et al. (1985) deal with the problem. McMurray argues that better estimates of heat radiation from flares can be obtained by using data from controlled field tests. Kent and DeFaveri discuss estimating flare radiation from physical data. The emissivity of the flare is important and this is estimated from flame temperature, reaction coefficients for complete combustion and steam quantities.

Flare Tips

Brzustowski (1976) gives a good account of the different kinds of flare tips available. The choice is largely dependent on the needs of the designer. For example, if the flare needs smoke suppression and only small quantities of heavy saturates
and unsaturates are being flared, then a centre steam flare tip is suitable.

Sizing a Flare Stack

Tan (1967) offers nomograms for speeding up the process of stack height and knock-out drum sizing, together with purge gas rates and steam requirements. Oenbring and Sifferman (1980) ask "..are current methods too conservative?". They compare radiation, noise and flame length and deflection from actual plant flares with the methods given in API RP 521. They suggest that before designing for the worst case the designer considers the likelihood of worst cases occurring simultaneously and the consequences of this occurring. Finally, Seebold (1984) discusses general calculation procedures including height, flame stability, purge and ignition.

Other papers in this field deal exclusively with specific plants, e.g. Feldman and Grossel (1968) talk about an ethylene plant and Cindric (1984) discusses ammonia plant front end vents. Kletz (1974 and 1985a) gives details of case histories involving flare stacks and the subsequent lessons that designers can learn from them.

Vapour Recovery

McGill and McGill (1978) report on plant vent and flare hydrocarbon recovery and reprocessing systems. Refineries and petrochemical plants have traditionally suffered continuous losses of valuable hydrocarbons to plant flares and free vents. They discuss various aspects of designing such a system, including problem identification and selecting the best option. Packages commercially available include clean gas recovery, in which a clean gas vent needing no processing to be used as fuel is recompressed in a simple compression-cooling system. Also available is a system for recovery and polishing gas for fuel use, which uses a compression-absorption system and a system for recovery from flares containing solids. Generally, a vapour recovery system is
implemented by using a compressor which takes suction off the flare line. This has the advantage of reducing pollution and keeping air out of the stack.

Various papers have been discussed in this section - there are many more. However, we have been able to show an important aspect of acquiring knowledge and some sources of rules and heuristics. Points to note are the author and date of the work, any contradictions that occur from one worker to another and words such as "normally", "generally" etc. These words imply that there are exceptions to the statements made.

11.4 Discussion with an Expert

Whilst collecting together the knowledge required for designing a flare system, a number of points of confusion arose. It is therefore relevant at this stage to include details of discussions with an expert to fill in some finer points concerning the design. The actual points creating confusion become apparent from the transcript below which is taken from a discussion with an expert. It has been quoted verbatim so that nothing is lost through re-writing it. The discussion took the form of a question and answer session and is reproduced in Table 11.1

<table>
<thead>
<tr>
<th>Question</th>
<th>Expert Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why are ground level flares used?</td>
<td>They are used to minimize light and noise. Their use is considered a luxury and the question of using them is based on environmental reasons.</td>
</tr>
<tr>
<td>Are ground flares always used in conjunction with</td>
<td>Yes.</td>
</tr>
</tbody>
</table>

232
elevated flares?

Are elevated flares only used in emergencies?

The use of an elevated flare can be classified into two:
1. In refineries they are used on a continuous basis and a purge of nitrogen may not be needed.
2. In chemical plants they are not in continuous use, and usage may be rare. A nitrogen/flue gas purge is needed, and they are strictly for emergencies.

How does a flare designer go about designing a flare?

The design of a flare stack is based on the maximum rate of flow and on the composition of the gas. Typically, for an olefin plant the worst flow occurs, that is emergency, when the cooling water fails. This is said to occur once in 10 years. The designer will then consider the radiation levels of the stack. He will have to optimize between the height of the flare and the radiation levels. The higher the stack, the lower the heat radiation at ground level, for sterilization.
<table>
<thead>
<tr>
<th>Question</th>
<th>Expert Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>What are the requirements of any flare system?</td>
<td>A pilot light, an ignitor, steam for smokeless flaring, a purge, unless flaring is continuous, a knock-out pot, flow measurement, a check for oxygen once per shift, and a tip, which will be of specialist design.</td>
</tr>
<tr>
<td>Which seals are used and when?</td>
<td>Seals are optional extras. It is strongly recommended that there are no flame arresters, no molecular seals and no lutes, except in warm weather.</td>
</tr>
<tr>
<td>How are large quantities of hydrogen dealt with?</td>
<td>Hydrogen is best vented in a completely separate system, and it need not be lit. The important property with hydrogen, and indeed $\text{CH}_4$ and below, is its density. In order to dispose of hydrogen one would need vast quantities of nitrogen because its buoyancy would keep it down. The amount of nitrogen needed could not be supplied. So, hydrogen should be directed to atmosphere, along with high velocity gases, restricted to $\text{C}_3$ and below, in infrequent discharges if quantities are trivial.</td>
</tr>
</tbody>
</table>
11.5 Rules for Flare System Design

So far in this chapter we have described many aspects of flare system design together with an account of the literature available. In this section we draw out some rules, some in 'if-then' format. They have been drawn mainly from the literature, and below some rule derivations are described by quoting the sentence from the work then writing the rule. This gives an insight into how rules might be formed. Not all of the rules are noted in this way, the rest appear in the summary table which gives the source, date and topic of the rule. Finally, contradictions are noted.

11.5.1 Rules Derivation

Here the rule source is quoted together with the author and date of the source.

Quote :-

"Ground flares are particularly useful to the petrochemical industry where start-ups or shut-downs of units, e.g. ethylene units, can give rise to large volumes of gases that have to be burned over an extended period."

Rule :-

If there are large volumes of gases to be burned over an extended period, then a ground flare should be used.

Boeije, 1979

Quote :-

"To prevent air pollution at low level, gas containing relatively high concentrations of sulphur compounds or other toxic components may have to be burned at the elevated flare or in incinerators specifically designed for burning the gases."
Rule :-

If the gas contains toxics, then they should not be sent to the ground flare, but the elevated flare or special incinerators.

Boeije, 1979

Quote (on utility flare tips) :-

"...can be used for flaring hydrogen, methane, hydrogen sulphide and carbon monoxide..."

Rule :-

If either hydrogen, methane, hydrogen sulphide or carbon monoxide are to be flared, then a utility flare tip may be used.

Brzustowski, 1976

Quote (on centre steam flare tips) :-

"...cheapest provision for smoke suppression for small installations in which only small quantities of heavy saturates and unsaturates are flared."

Rule :-

If only small quantities of heavy saturates and unsaturates are to be flared and smoke suppression is required, then a centre steam flare tip should be used.

Brzustowski, 1976

Quote :-

"The quicker the steam becomes mixed with inspirated air, the less noise produced by the steam system itself."
Rule:

If the steam is mixed quickly with inspirated air, then less noise will be produced.

Seebold, 1972

11.5.2 Rules Summary

The following is a table listing all the rules derived concerning flare system design.

Table 11.2 Rules for Flare System Design

<table>
<thead>
<tr>
<th>Source</th>
<th>Date</th>
<th>Flare Topic</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Kletz</td>
<td>1985b</td>
<td>Atmospheric venting</td>
<td>If the gas velocity can be made high, frequency of discharge is low, quantities are small, and gas contains C₃ and below, then vent to atmosphere.</td>
</tr>
<tr>
<td>2. Bluhm</td>
<td>1985</td>
<td>&quot;</td>
<td>If start-up conditions apply, then vessels are usually vented to atmosphere.</td>
</tr>
<tr>
<td>3. Paruit and Kimmel</td>
<td>1979</td>
<td>Controlled blowdown</td>
<td>Conditions favourable to controlled blowdown are large gas inventories, shortage of space and off-shore.</td>
</tr>
<tr>
<td>4. Kletz</td>
<td>1984</td>
<td>Elevated</td>
<td>Elevated flares flare in the mol.wt. range 25-46</td>
</tr>
<tr>
<td>5. Klooster</td>
<td>1975</td>
<td>&quot;</td>
<td>Typical flowrate through an elevated flare is 1,200,000 lb/hr.</td>
</tr>
<tr>
<td>Source</td>
<td>Date</td>
<td>Flare Topic</td>
<td>Topic</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>---------------</td>
<td>-------</td>
</tr>
<tr>
<td>Kletz</td>
<td>1985a</td>
<td>Ground</td>
<td>Excess continuous gases which are not recoverable should go to the ground flare.</td>
</tr>
<tr>
<td>Klooster</td>
<td>1975</td>
<td>&quot;</td>
<td>If there are light ends in the molecular weight range 20 -40 they are usually sent to ground flare.</td>
</tr>
<tr>
<td>Brzustowski</td>
<td>1976</td>
<td>Ground</td>
<td>Design heat release rates for ground flares are in the range 1 to 100MW.</td>
</tr>
<tr>
<td>Boeije</td>
<td>1979</td>
<td>&quot;</td>
<td>If there are large quantities of gas to be burned over an extended period then a ground flare should be used.</td>
</tr>
<tr>
<td>Boeije</td>
<td>1979</td>
<td>&quot;</td>
<td>If the gas contains toxics, then they should not be sent to ground flare but to the elevated flare or special incinerators.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1984</td>
<td>General</td>
<td>Stacks should not contain any bolted joints between unmachined surfaces.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1984</td>
<td>&quot;</td>
<td>A boot should be fitted at the bottom of the stack for the collection of debris (Fig 11.4).</td>
</tr>
<tr>
<td>Boeije</td>
<td>1979</td>
<td>&quot;</td>
<td>If the gas discharged comes under one of the following : acts corrosively</td>
</tr>
</tbody>
</table>

238
<table>
<thead>
<tr>
<th>Source</th>
<th>Date</th>
<th>Flare Topic</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>blocks passage for effluents reacts/decomposes explosively produces toxic combustion gases then the gas should not be discharged to the flare system.</td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>Kletz 1984</td>
<td>&quot;</td>
<td>Large stacks should be fitted with an oxygen analyser, small stacks should have a portable analyser.</td>
</tr>
<tr>
<td>15.</td>
<td>Brzustowski 1976</td>
<td>Flare tip</td>
<td>If either hydrogen, methane, hydrogen sulphide or carbon monoxide are to be flared, then a utility flare tip should be used.</td>
</tr>
<tr>
<td>16.</td>
<td>Brzustowski 1976</td>
<td>&quot;</td>
<td>If the system requires smoke suppression, then a utility flare tip is not recommended.</td>
</tr>
<tr>
<td>17.</td>
<td>Brzustowski 1979</td>
<td>&quot;</td>
<td>If only small quantities of heavy saturates and unsaturates are to be flared, then a centre steam tip is recommended.</td>
</tr>
<tr>
<td>18.</td>
<td>Brzustowski 1979</td>
<td>&quot;</td>
<td>If large flows of heavy saturates and unsaturates are to be flared, then a steam ring flare tip is recommended.</td>
</tr>
<tr>
<td>19.</td>
<td>Seebold 1984</td>
<td>&quot;</td>
<td>Three continuous pilots should be placed around the flare tip.</td>
</tr>
<tr>
<td>Source</td>
<td>Date</td>
<td>Flare Topic</td>
<td>Rule</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>Kletz</td>
<td>1985b</td>
<td>Hydrogen</td>
<td>If hydrogen is present then it should be sent through a separate vent at least 10 feet from buildings.</td>
</tr>
<tr>
<td>API</td>
<td>1957</td>
<td>Layout</td>
<td>If the following are present oil water separators floating roof tanks then there should be at least 200 feet horizontally from the flare to these units.</td>
</tr>
<tr>
<td>API</td>
<td>1957</td>
<td></td>
<td>A refinery flare should not extend less than 50 feet above the top of the tallest unit within a radius of 100 feet.</td>
</tr>
<tr>
<td>API</td>
<td>1957</td>
<td></td>
<td>If ground flares are present, then they should be at least 300 feet from process units.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1985a</td>
<td>Purge</td>
<td>If not flaring refinery gases, then a purge of 0.03-0.06 m/s is recommended at all times.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1984</td>
<td></td>
<td>If hot gases are being flared then there should be a 10 times increase in the purge gas flow rate.</td>
</tr>
<tr>
<td>Seebold</td>
<td>1972</td>
<td>Steam</td>
<td>If the steam becomes thoroughly mixed with inspired air then, less noise will be produced by the steam.</td>
</tr>
<tr>
<td>Source</td>
<td>Date</td>
<td>Flare Topic</td>
<td>Rule</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>-------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Seebold</td>
<td>1972</td>
<td>&quot;</td>
<td>If moisture condensation shock is a problem, then dry or superheated steam is recommended.</td>
</tr>
<tr>
<td>Seebold</td>
<td>1972</td>
<td>&quot;</td>
<td>A steam injection rate of 1/2 lb of steam per lb of hydrocarbon is recommended.</td>
</tr>
<tr>
<td>Seebold</td>
<td>1972</td>
<td>&quot;</td>
<td>Steam is supplied to the injectors at 100-150 psig.</td>
</tr>
<tr>
<td>Seebold</td>
<td>&quot;</td>
<td>&quot;</td>
<td>Steam should not be used in stacks which might freeze.</td>
</tr>
<tr>
<td>Reed</td>
<td>1968</td>
<td>Pollution</td>
<td>If sulphur dioxide is present, then on emission it should not produce a grade level concentration of greater than 0.1ppm.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1985</td>
<td>&quot;</td>
<td>If hydrogen sulphide is present, then it can be burned off at the stack.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1985</td>
<td>&quot;</td>
<td>If sulphur dioxide and/or sulphur trioxide are present, then they should be removed by scrubbing.</td>
</tr>
<tr>
<td>API</td>
<td>1957</td>
<td>&quot;</td>
<td>If ammonia, or trace quantities of phosgene, or hydrogen sulphide or hydrogen cyanide are present, then complete burning is required.</td>
</tr>
<tr>
<td>Reed</td>
<td>1968</td>
<td>&quot;</td>
<td>If conditions are such that</td>
</tr>
<tr>
<td>Source</td>
<td>Date</td>
<td>Flare Topic</td>
<td>Rule</td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
<td>-------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Andrew +</td>
<td>1974</td>
<td>&quot;</td>
<td>If blockage is a problem, then a steam trace can be used.</td>
</tr>
<tr>
<td>Williams</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seebold</td>
<td>1984</td>
<td>Seals</td>
<td>If the gas velocity is 1-3 feet per second, then a gas seal need not be fitted.</td>
</tr>
<tr>
<td>Company</td>
<td>1966</td>
<td>&quot;</td>
<td>If the stack is handling hot gases or there is a risk of blockage, then a molecular seal not suitable.</td>
</tr>
<tr>
<td>Reed</td>
<td>1972</td>
<td>&quot;</td>
<td>If the gas content in the vertical part of the flare can be buoyant, then a molecular seal is recommended.</td>
</tr>
<tr>
<td>Kletz</td>
<td>1985a</td>
<td>Flame arrester</td>
<td>If the gas can decompose with air, then fit a flame arrester.</td>
</tr>
<tr>
<td>Andrew +</td>
<td>1974</td>
<td>Knockout</td>
<td>If there is likely to be formation of condensibles, then a knockout drum is needed.</td>
</tr>
<tr>
<td>Williams</td>
<td></td>
<td>Drum</td>
<td></td>
</tr>
<tr>
<td>Andrew +</td>
<td>1974</td>
<td>&quot;</td>
<td>If a knockout drum is used, liquid then a hold up time of between 10 and 30 minutes is recommended.</td>
</tr>
<tr>
<td>Williams</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
If a knockout drum is used, then an additional seal is needed at the base of the stack to trap any liquid formed downstream of the drum.

If a knockout drum is used, then demister mats are not recommended due to liability of fouling.

Most of the rules are therefore of the 'if-then' type. Those which are not are in cases where 'if' cases do not occur, i.e. where it is certain that the item under consideration will be used on a flare stack.

It is evident from the rules table that there are various types of rule. Rule types that have been identified are prohibition rules, suggestions of alternatives, suggestions of additions, selection of one alternative, suggestions to a solution, and typical numerical design values. The following table categorizes the rules under these headings:

<table>
<thead>
<tr>
<th>Rule Type</th>
<th>Rules in these Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prohibition</td>
<td>10,11,13,30,31,34,38,44</td>
</tr>
<tr>
<td>Alternatives</td>
<td>1,2,3,4,6,7,9,10,36</td>
</tr>
<tr>
<td>Additions</td>
<td>12,14,26,33</td>
</tr>
<tr>
<td>Solutions</td>
<td>15,16,17,18,27,32,35,39,41,43</td>
</tr>
</tbody>
</table>
There are likely to be exceptions to the rule types prohibitions and selections of alternatives.

The following table shows the nature of expertise in these rule types:

Table 11.4 Expertise in Flare Rules

<table>
<thead>
<tr>
<th>Expertise often Explicit</th>
<th>Expertise often Elusive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prohibition rules</td>
<td>Suggestions of alternatives</td>
</tr>
<tr>
<td>Typical design values</td>
<td>Suggestions of additions</td>
</tr>
<tr>
<td></td>
<td>Suggestions to a solution</td>
</tr>
</tbody>
</table>

11.5.3 Conflicts

The rules table shows rules for many aspects of flare design. They have been derived from the literature, as shown in the previous section and refined after discussions with an expert. Many of the rules come from literature written some 20 years ago, and this has been the main cause of the conflicts discovered. Apart from this the literature is consistent in its opinions on the design of flare systems. It is worth noting some of the conflicts that have been discovered.

Many earlier workers recommend the use of flame arresters for flashback protection. Today it is widely accepted that flame arresters can cause blockages, so they are now only recommended in severe cases where the flared gas is likely to decompose with air.

The same is true of molecular seals. These are 'inverted cans' put in the stack to stop air coming down it. They are susceptible to blockage by carbon deposits and ice, and their use is not recommended today. Water seals may be subject to water loss or freezing in extreme weather conditions and also
fall into this category. It is accepted today that the only effective and safe way to seal a flare system is by the use of a continuous gas purge.

Other sources of conflict arise from changes in legislation regarding grade level concentrations of toxics, noise, smoke and light. In these cases the date of the source is therefore very important.

11.6 Flare Design Expertise and Strategy

Having described aspects of flare design through the use of an awareness aid and a set of rules, the overall shape of the design problem is becoming apparent. In this section we propose a strategy for flare system design together with the method that a designer might use. The first section is concerned with the expertise in flare system design.

11.6.1 Flare System Design Expertise

The expertise in the design of a flare system comes in the form of knowledge of the components of a flare system and the awareness of conventional solutions, alternatives available, costs, consequences and probabilities. In the actual process of design, distinctions and decisions must be made by being aware of the criteria and consequences. In order to show these aspects some illustrative examples are given below.

Basic System

The designer must be aware of all the possible components of a flare system together with which of them is needed on the particular installation under consideration. He must also decide whether a ground flare is needed alongside an elevated flare, and whether using multiple flares will maintain capacity during a partial shutdown of one of them.

In the design of an elevated flare there are certain minimum components, such as a flare tip, a stack and pilot burners. The
expertise is then in the form of decision-making, deciding what other components are needed, such as molecular seals, knockout drums, smoke suppression and noise suppression. The rules developed above will help in these areas.

For a ground flare, there is also a minimum requirement of all such flares, that is that the air inlet shall be screened off to cut out direct routes for noise and light. An alternative is grouped burners for consecutive operation so giving a greater turn-down.

These aspects of flare design are characterized by the fact that each has minimum requirements and the expertise is in the form of deciding which 'add-on' components are needed. Some of these are discussed below.

Combustion Devices

This aspect is a good example of alternatives available and criteria for deciding. The nature, frequency and quantity of relief are factors which aid the decision-making, together with the effects on the environment.

Steam assisted burners are one such device, but only the external type is to be used if freezing is a problem. Air-assisted burners are to be used if smokeless burning is required, high pressure gas assisted burners are to be used if heat radiation is a problem and water assisted gas burners for ground flare installations. Here, though, an awareness of the criteria for freezing is required.

An awareness of operational flaring (periods exceeding 30 minutes) is needed for deciding if smoke suppression is needed, together with a knowledge of legislation and local conditions.

Siting

There are no specific siting rules so the decisions must be made on the basis of the characteristics of the particular flare
and local conditions. Knowledge of the proximity of serving units, the likely route of the flare line, prevailing winds, thermal radiation, and the probability of burning droplets are needed. In the case of multiple flares, it must be possible to maintain one whilst the others are operational (unless shutdown of all flares is an operational requirement).

**Flashback Protection**

There are various ways of dealing with flashback protection including purging, liquid seals and gas seals. These may be used singly or in combination. Thus, knowledge of each of these is required together with an understanding of the particular flare system.

Purging is probably the most complex aspect here. A choice must be made for the gas used with attention given to the consequences of releasing unburned toxic materials. The purge gas rate can be determined by a conventional method, although attention is needed if the purge gas has a density lighter than that of air. If a flammable gas is used, a rate which adequately keeps the flare alight is needed, although internal burning of the flare tip may occur. If this is a problem, it must be economically evaluated against alternatives such as an increase in the purge rate, an upgrade of the material specification of the tip, replacing the tip more often or providing tip cooling.

Seal designs are conventional solutions the only distinction being which seal to use if at all. Criteria include not using flame arresters if blocking is a problem, and no water seals if the temperature of the vapour can fall below 0°C. General criteria to satisfy in the design of a seal system are:–

- Prevention of hydrocarbon build up.
- Prevention of displacement of seal liquid.
- Maintaining the correct seal liquid level over the operating pressure range.

As we can see, the design of a complete system encompasses
many types of knowledge and expertise. Conventional solutions do exist for some aspects for elevated flares, flare tips, ground flares and knockout drums. Additional requirements for a system involve awareness of alternative solutions together with the criteria for deciding which alternative will satisfy the design constraints.

The following table is a summary of some of the elements of the expertise required for the design of a flare system:

Table 11.5 Expertise for the Design of a Flare System

<table>
<thead>
<tr>
<th>Knowledge - codes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>- literature</td>
<td></td>
</tr>
<tr>
<td>- expertise</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rules - if-then type</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>- sources and dates</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Expertise - distinctions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>types of plant</td>
<td></td>
</tr>
<tr>
<td>refiners</td>
<td></td>
</tr>
<tr>
<td>petrochemical plants</td>
<td></td>
</tr>
<tr>
<td>- alternative solutions</td>
<td></td>
</tr>
<tr>
<td>atmospheric venting</td>
<td></td>
</tr>
<tr>
<td>controlled blowdown</td>
<td></td>
</tr>
<tr>
<td>trip systems</td>
<td></td>
</tr>
<tr>
<td>- conventional solutions (archetypes)</td>
<td></td>
</tr>
<tr>
<td>elevated flares</td>
<td></td>
</tr>
<tr>
<td>ground flares</td>
<td></td>
</tr>
<tr>
<td>flare tips</td>
<td></td>
</tr>
<tr>
<td>knockout drums</td>
<td></td>
</tr>
<tr>
<td>seals</td>
<td></td>
</tr>
<tr>
<td>- alternatives for specific flare items</td>
<td></td>
</tr>
<tr>
<td>flashback protection</td>
<td></td>
</tr>
<tr>
<td>flare tips</td>
<td></td>
</tr>
<tr>
<td>- awareness</td>
<td></td>
</tr>
<tr>
<td>local conditions</td>
<td></td>
</tr>
<tr>
<td>legislation</td>
<td></td>
</tr>
<tr>
<td>environment</td>
<td></td>
</tr>
<tr>
<td>costs</td>
<td></td>
</tr>
</tbody>
</table>

249
11.6.2 Flare Design Strategy

The following table is a design strategy for the overall design of a flare relief system. The list is in the form of aspects of the design that we have discussed in this chapter and is in an order that a designer might use.

**Table 11.6 A Strategy for Flare Design**

<table>
<thead>
<tr>
<th>System definition</th>
<th>Required capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Alternatives</td>
</tr>
<tr>
<td></td>
<td>atmospheric venting</td>
</tr>
<tr>
<td></td>
<td>controlled blowdown</td>
</tr>
<tr>
<td></td>
<td>trip systems</td>
</tr>
<tr>
<td>Design options</td>
<td>multiple flare systems</td>
</tr>
<tr>
<td></td>
<td>dry gas vs wet gas</td>
</tr>
<tr>
<td></td>
<td>regular vs emergency flow</td>
</tr>
<tr>
<td></td>
<td>high vs low level flares</td>
</tr>
<tr>
<td>Hazards</td>
<td>Blockage</td>
</tr>
<tr>
<td></td>
<td>Explosion</td>
</tr>
<tr>
<td></td>
<td>Heat radiation</td>
</tr>
<tr>
<td></td>
<td>Toxic gases</td>
</tr>
<tr>
<td></td>
<td>Liquid carryover</td>
</tr>
<tr>
<td>Evaluation criteria</td>
<td>Heat radiation levels</td>
</tr>
<tr>
<td></td>
<td>Smoke levels</td>
</tr>
<tr>
<td></td>
<td>Noise levels</td>
</tr>
<tr>
<td></td>
<td>Toxicity limits</td>
</tr>
<tr>
<td></td>
<td>Space considerations</td>
</tr>
</tbody>
</table>

**Economics**
Relief header costs
Purge gas costs
Land sterilisation
Equipment costs

Hardware design
Process design
  relief header
  flare stack
  knockout drum
  flare tip
  pilot light
Mechanical design
  materials of construction
  corrosion
  brittle fracture

Explosion Protection
  Purge gas
  Seals
  Oxygen monitoring
  Flame arresters

Process control
  Manual vs automatic control
  Controlled blowdown

Startup
  Initial purge with inerts
  High gas flows

Shutdown
  Air diffusion

Plant operation
  Operating instructions
  Emergency action
  Plant Maintenance
By studying in detail the design of a flare system, it has become apparent that the strategy is very much a linear forward moving activity. There seems to be little if any iteration involved. However, there may be some in the actual sizing of the stack and other system components. Apart from that, the design appears to move from one section to the next, as outlined above. The problem is probably amenable to solution by an algorithmic approach or by a production system using forward chaining. The number of rules would, however, probably be greater than those identified here. Of the two methods, the production system approach appears attractive with its usual advantage of allowing the rules to be written down in a transparent form. It is the production system approach that is being pursued here.

11.7 A Prolog Program for Flare System Design

In this section a brief description of a program for flare system design is given. The program is written in Prolog and runs on a Honeywell Multics DPS8. It is intended to show how Prolog might be used for a more extensive application of the design of a flare system. It is based on the methodology of Winston, given in Chapter 6, who designed a toy system for packaging groceries called BAGGER. He used a step design technique in production system format. Those aspects have been incorporated in this work, including the following design steps:-

check flare items
determine flare type
determine flare layout
determine flare tip
establish alternatives
establish safety requirements
general aspects

A listing of the program is given in Appendix F, together with a list of production rules that the program uses. Here we will show the output of the program.
Like BAGGER this program uses forward chaining strategy, the control mechanism then matching the data with the rules in the rule base. It is non-interactive, and the output consists of a list of recommendations for various flare applications. The program contains a random list of rules about flare system design and provides a design according to an ordered sequence. The rules used have been derived from the rules listed above.

The program, like BAGGER, uses context limiting conflict resolution strategy because by convention the first condition clause of each rule limits the rule to a particular step. Specificity ordering is also used in that the rule with more stringent requirements is triggered after a rule with less stringent requirements.

11.7.1 Program Output

The output from the flare program is shown below :-
York Portable Prolog Release 2.1.

Please pass on any comments on this Prolog to RSMKirkwood via Multics Mail.

?- flare consulted.

?-
ground_flares present !

stack present !

*** step is flare_type ***

There is an item to flare !

hydrocarbon

Conditions are lo_flow and continuous
Conditions are lo_flow and excess

*** therefore ground_flare ***

Conditions are hi_flow and non_recoverable
Conditions are lo_flow and non_continuous

*** therefore elevated_flare ***

*** step is plant_layout ***

There is a ground flare !

plant_layout

Conditions are any_flow and process_units

*** therefore three_hundred_feet_separation ***
There is a stack!

plant_layout

Conditions are any_flow and oil_water_separator
Conditions are any_flow and floating_roof_tank
Conditions are process_units and stack
*** therefore two_hundred_feet ***

*** step is flare_tip ***
There is an item for flare tip!

Conditions are hi_flow and smokeless_flame
Conditions are unsaturated_hydro(o) and hi_flow and smokeless_flame
*** therefore steam_ring_flare_tip ***

Conditions are hydrogen_sulphide and any_flow and smokey_flame
Conditions are methane and any_flow and smokey_flame
Conditions are carbon_monoxide and any_flow and smokey_flame
*** therefore utility_flare_tip ***

Conditions are saturated_hydro(o) and lo_flow and smokeless_flame
Conditions are unsaturated_hydro(o) and lo_flow and smokeless_flame
Conditions are saturated_hydro(o) and hi_flow and smokey_flame
Conditions are unsaturated_hydro(o) and hi_flow and smokey_flame
*** therefore centre_steam_tip ***

*** step_is alternatives ***
There is an item for alternatives! 
gas

Conditions are lo_flow and toxic
*** therefore do_not_flare ***

Conditions are hi_flow and hydrogen
Conditions are discharge_freq_lo and hydrogen
*** therefore vent_to_atm ***

Conditions are general_discharge and hydrogen
Conditions are general_discharge and methane
*** therefore use_separate_vent ***

*** step is safety ***
There are safety items!
gas + hydrogen

Conditions are hi_flow and large_stack
*** therefore oxygen_alarm_five_per_cent ***

gas

Conditions are hi_flow and large_stack
*** therefore oxygen_alarm_two_per_cent ***

Conditions are any_flow and maintenance
*** therefore purge_with_nitrogen ***

Conditions are any_flow and decompose_in_air
*** therefore fit_flame_arrestor ***

***step is general flaring rules ***

There are general flare items 1.
hydrogen

Conditions are any_flow and venting
*** therefore vent_at_least_ten_feet_above_buildings ***

Conditions are ammonia and any_flow and burning
Conditions are gas + phosgene and any_flow and burning
Conditions are gas + hydrogen_sulphide and any_flow and burning
*** therefore complete_burning_required ***

no
?- [Leaving Prolog]

257
11.8 Summary

In this chapter aspects of the design of a flare system have been described with a view to obtaining a design strategy and an understanding of the expertise of expert flare designers. This has been done by working from literature available and by talking with an expert. As a result, a flare system 'awareness aid' has been established together with a reasonably comprehensive set of rules for flare system design. The rules were mostly in 'if-then' format and could be characterised by being prohibition rules, rules for alternatives, rules for additions, rules for conventional solutions, and typical design values. This has been followed by a list of the expertise involved and the design strategy that an expert might employ.

The expertise comes in many forms including knowledge of the components of a system, rules to apply in designing the system, awareness of alternative solutions, awareness of conventional solutions including archetypes, awareness of local conditions, costs and probabilities, etc. and criteria needed for making decisions.

The design strategy included aspects of flare system design in an order that the designer might adopt. They include system definition, hazards, evaluation criteria, economics, hardware design and explosion protection, etc. It has been noted that this list seems to be a forward-moving one, and that a forward chaining problem solver or an algorithm might be appropriate.

Finally, a Prolog program has been written for various aspects of flare system design. It uses forward chaining and employs conflict resolution strategies.
Pressure Relief and Blowdown

12.1 Introduction

The subject of pressure relief and blowdown was introduced as a candidate topic in Chapter 7. It is one of the topics thought worthy of further study in order to understand the design process and the expertise involved. In contrast to the previous sections of this work, no programs have been written for the design of a pressure relief system. The topic has been chosen since it is one that appears to have deeper structure than the others studied. In this chapter, various aspects of the design of a relief system are described, an account of the literature is given, some rules are proposed and a design strategy and design expertise are established from an example problem.

Systems which operate under pressure need to be protected against overpressure. Various standards and codes are available to help with the design, most notably BS5500 and API RP 520 and API RP 521. These API codes have been the prime sources of information in this work. The basic requirement of any relief system is the protection from excessive pressure or vacuum, excessive temperature, corrosion or explosion or similar. Also, there is a legal requirement for overpressure protection, and this, together with some of the code requirements is discussed next.

12.1.1 Legal and Code Requirements

Apart from the obvious safety requirements that need to be met, the law in the U.K. states certain conditions regarding pressure vessels. Fitt (1974) regards them as having limited scope, and the only items covered are steam boilers, steam and air receivers and certain stills and closed vessels. They are covered by the Chemical Works Regulations of 1922 and the Factories Act of 1961. Apart from this, pressure relief is dictated by the common law obligation to safeguard employees and members of the public and insurance requirements.
API RP 520 and 521 apply to relieving devices and their discharge systems on refinery pressure vessels and equipment designed for a maximum allowable working pressure of more than 15 psi. They are recognized standards for the safe design and operation of pressure relieving systems. API RP 520 is divided into two parts, the first on design and the second on the installation of such systems. Part 1 starts with an introduction to the different types of relieving device available, relief requirements and fires outside vessels. Requirements for the relief of vessels exposed to open fires are then discussed followed by protection of vessels from fire. Part 2, on installation, is concerned with inlet piping, pressure drop limitations and piping configurations together with discharge piping and valve location. API RP 521 is a guide for all aspects of pressure relieving systems and starts with some causes of overpressure and determination of individual relieving rates. Next is a section on the criteria for selecting of a suitable disposal method, with the final section concerned with the physical design of one aspect, a flare relief system.

BS 5500:1976 'Unfired Fusion Welded Pressure Vessels' covers a wide range of topics to do with the physical design of such vessels. The standard applies to the design, construction, inspection, testing and certification of pressure vessels. With respect to materials, the standard covers the selection of materials, the nominal design strength and carbon, carbon manganese and alloy steels. The section on design includes corrosion, erosion and protection, design stresses, vessels under internal and external pressure, supports, attachments and internal structures and the design of welds. Cutting, forming and tolerances, welded joints and heat treatment are also covered in manufacture and workmanship, with a final section on inspection and testing. An Appendix of BS 5500 covers pressure relief devices.

12.1.2 Some Definitions

Before proceeding with the discussion it is worth noting
some definitions concerned with pressure relief and blowdown. Basic definitions appear in API RP 520 and 521, and Emerson (1985) has given an enhanced explanation of these.

Safety Valve

This is similar to a relief valve but opens rapidly and is hence sometimes called a 'pop' valve. It is primarily used for gas or vapour service.

Safety Relief Valve

This is an automatic pressure actuated relieving device suitable for use as either a safety valve or a relieving device. It can therefore possess all the features of both types of valves, and is thus suitable for both liquid and vapour service.

Relief Valve

This is an automatic pressure relieving device activated by the pressure at the inlet of the valve. It is used mainly for liquid service, and characteristics include valve opening in proportion to any increase in pressure over opening pressure.

Maximum Allowable Working Pressure

The maximum allowable working pressure is the maximum pressure at which a single pressure relieving device may be set for a system to prevent exceeding the limit.

Operating Pressure

This is the pressure to which the vessel or pipe is subjected in normal operations.
Set Pressure

The set pressure is the inlet pressure at which the pressure relieving device is adjusted to open under service conditions. Usually the operating pressure does not exceed 90% of the pressure relief valve set pressure.

Overpressure

The overpressure is the pressure over the set pressure of the relieving device.

Blowdown

Blowdown is the difference between the set pressure and the reseating pressure of a pressure relief valve, expressed as a percent of the set pressure.

Back Pressure

The pressure that exists on the outlet side of the valve is called the back pressure.

12.2 Aspects of Pressure Relief Design

In this section, various aspects of the need for and design of a pressure relief system are described. This includes sections dealing with the causes of overpressure, considerations for the determination of individual relieving rates, a discussion on the kinds of relieving device available and disposal systems.

12.2.1 Causes of Overpressure

Why have a pressure relief system? As we have outlined above, it is because vessels may become subject to a rise in pressure. There are various reasons for this and some of these were listed in the chapter on flare relief systems. It is worth reminding ourselves of the causes listed there: -
Loss of cooling water
Power failure
Failure of heat exchanger tubes
Control system failure
Chemical reaction runaway

In this section these and other qualitative causes of overpressure will be outlined in more detail. Overpressure itself is by an inbalance of flows in the plant, and this may have avariety of causes.

Closed Outlets on Vessels

If a block valve on the outlet of a vessel is closed while the plant is on stream, then a pressure exceeding the maximum allowable working pressure may result. If the block valve is not sealed or locked in the open position then a pressure relieving device is needed. To avoid such a hazard occurring it is recommended that block valves are not used interposed between vessels in series.

Water/Steam Hammer

Water hammer, steam hammer and process changes must also be considered when designing for overpressure. Water hammer cannot be dealt with by a safety relief valve due to its relatively slow response time. It causes pressure wave damage in vessels and piping, and the API recommend using pulsation dampers. Steam hammer occurs in pipes that contain condensate. Use of a slow closing valve may help to alleviate the problem.

Utility Failures

The following table, as outlined by the API RP 521, gives a list of some possible utility failures which will lead to overpressure together with the equipment that may be affected. Utility failure is an indirect cause of one of the other causes of failure.
### Table 12.1 Utility Failures

<table>
<thead>
<tr>
<th>Utility Failure</th>
<th>Equipment Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electric</td>
<td>Pumps, fans, compressors, instrumentation, including valves</td>
</tr>
<tr>
<td>Cooling water</td>
<td>Condensers, coolers, jackets</td>
</tr>
<tr>
<td>Instrument air</td>
<td>Instrumentation, including valves</td>
</tr>
<tr>
<td>Steam</td>
<td>Turbine drivers, reboilers reciprocating pumps</td>
</tr>
<tr>
<td>Fuel (oil, gas)</td>
<td>Boilers, reboilers, engine drives, compressors, gas turbines</td>
</tr>
<tr>
<td>Inert gas</td>
<td>Seals, catalytic reactors purge</td>
</tr>
</tbody>
</table>

**Fire**

Fires can occur from leakage of hydrocarbons from refinery equipment. The source of liquid or gas leakage feeding the fire may be either the vessel exposed to the flame or an adjacent operating or storage vessel. The leakage may result from leaking joints in a pipeline, from equipment or from operational mishaps. If there are ground slopes, then the fires may spread some distance from the source in the case of liquids. Gases can be spread by air-currents.

If the fire is open and free-burning, then the vessels and other adjacent equipment exposed to the flame will absorb the heat by radiation or by direct contact. The vessel con-
tents, if exposed for long enough, will then become heated and vapour will be generated, the pressure will rise, and the safety relief valve will open. The valve will then limit the maximum pressure in the vessel. In some cases, the rate of vapour generation will be greater than the rated capacity of the valve and the pressure then rises beyond the permissible overpressure and may reach a pressure unsafe for the vessel. To alleviate this problem, it is necessary, in sizing the safety relief valve, to consider the possibility of fire exposure. There are standard methods for dealing with general fire exposure. The method given by the API code is the principal such method. Not all cases are covered by the standard method. If the hydrocarbon leakage caused by a pipeline failure or a gasket blowing, for example, is then ignited, a jet fire will result. There is no standard approach for the impingement of a jet flame on a vessel.

12.2.2 Determination of Individual Relieving Rates

In this section we discuss the determination of individual relieving rates quantitatively.

The net energy input, in the form of heat and direct energy, is used to determine the liquid and vapour rates used to establish relief requirements. The maximum rate which must be relieved is then the peak individual relieving rate. The possibility of two unrelated failures occurring simultaneously is, according to the API, remote and does not usually need to be considered.

The discussion proceeds by taking each cause of overpressure in turn.

Cooling Water Failure to Condenser

On a distillation column the relieving rate is determined by a heat and material balance on the system at the relieving pressure. The required relief capacity is based on the total incoming steam and vapour plus that generated under normal
operation, less the vapour condensed by sidestream reflux.

**Reflux Failure**

In the case of a tower-top reflux failure, the required relief capacity is based on the total vapour to the condenser at relieving conditions, whilst for sidestream reflux failure, the difference between the vapour entering and leaving the section is considered.

**Power Failure**

This has to be considered for each individual situation, but generally the effect of power failure should be considered and the relief valve sized for the worst condition that can occur. In the case of power failure to a fractionator, the pumps would go down and hence reflux and cooling water would fail. The valves are then sized in a similar way to that of cooling water failure to a condenser. In the case of reactors, the agitation or stirring would stop, and the valve should then be sized for product generation from a runaway reaction. This is done by considering vapour generation from both normal and uncontrolled conditions.

**Entrance of Volatile Material**

This is a source of potential overpressure, but according to the API, no means for calculating the relieving requirements are available and no relieving device is provided for this contingency, although there may be exceptions to this.

**Abnormal Heat or Vapour Input**

In the case of fired heaters or steam reboilers, the required relief capacity is based on the estimated maximum vapour generation including non-condensables from overheating. In the case of a split reboiler tube the steam entering from twice the cross-sectional area of one tube is that rate required for relief capacity.
Protection of Vessels from Fire

Provision for fire relief does not give full protection against fire. Unwetted surfaces which are exposed to fire may experience overtemperature and may even rupture, even though overpressure of the vessel has not occurred. In addition to pressure relief, the vessel may be protected against fire by depressurization, fire insulation or water sprays. The depressurization of a vessel by removing part or all of the contents to a relief header or dump tanks not only relieves the pressure, but also decreases the amount of material able to feed the fire. The valve for depressurization should be capable of remote operation.

12.2.3 Pressure Relief Valves

As we have seen above there are various types of relief valve available, and there are rules which state which valve is to be used when. Anderson (1976) gives a good review of such devices. In this section, the various types of valve are described together with criteria for valve selection and aspects in sizing and capacity.

Safety Valve

These valves are suitable only for steam service in such applications as power boilers. They are available as either full nozzle or semi-nozzle design (the nozzle is the area in which the relieved pressure enters) and they reseat after relieving.

Safety Relief Valves

There are two types here, classified again in terms of the type of nozzle they have. The first, the full nozzle type, is used for liquid or vapour service and has the advantage that the body of the valve is isolated from the process fluid when not relieving. The second, the semi-nozzle type, can
again be used for liquid or vapour service, and is cheaper than the full nozzle type. Neither of them is suitable for polymer service, but both are suitable for applications such as unfired pressure vessels, pumps, compressors and water boilers.

**Pilot-Operated Safety Relief Valves**

This type of relieving device consists of two valves, a main valve and a controlling pilot valve. This has the advantage that it can withstand a high inlet pressure, it can be set to relieve near to the operating pressure, it can be remotely operated for manual depressurization and its high outlet velocity ensures good dispersion. However, it cannot be used in high temperature service or for services containing dirt, slurries or polymers. It is particularly suitable for clean high pressure gas service, gas pipelines and reciprocating compressors.

**Relief Valves**

The main type here is the base nozzle relief valve where the nozzle is formed from the base of the valve. Again, they are not good for polymer service, but they can handle toxics. Typical applications include pump discharge, thermal relief valves, heat exchangers and water heaters.

**Rupture Discs**

These consist of a breakable disc held between flanges. The two main types are pre-bulged and reverse buckling. They are good for slurries and polymers, they can handle large capacities and high pressures, and give fast response time and can therefore relieve explosions. They are, however, subject to fatigue, and once blown, the whole inventory is lost down to atmospheric pressure and the unit must be shutdown. For high pressure service, it is recommended that they be used in series with relief valves.
Setting and Capacity

The API RP 520 states that if the set pressure is equal to the maximum allowable working pressure (effectively the design pressure), then for abnormal operation the maximum pressure relief should not exceed 110% of the maximum allowable working pressure. For fire relief, it should not exceed 120% of the maximum allowable working pressure.

The valve setting and capacity requirements for adequate pressure relief differ between different codes and standards. The ICI LFG code summarises the requirements, a distinction being made between abnormal operation and that required for fire. The code states that under abnormal operation the maximum pressure attained during relief should not exceed 100% and 110% of the design pressure respectively. Under the combined situation of abnormal operation and fire relief the set pressure should not exceed 110% of the design pressure, and the capacity should be at least equal to the greater of the two capacities calculated for abnormal conditions and fire relief.

As given by BS5500, the size of a relief valve is estimated in terms of the rated capacity (in kg/s) and is a function of the actual discharge area, the coefficient of discharge, the molecular weight, the accumulation pressure and the absolute inlet pressure. The coefficient of discharge may vary, and BS5500 quotes 0.25 for a parallel inlet guided wing type of high lift value and 0.97 for a nozzle inlet type of flat disc value. More detail can be found in the standard, where the calculations for bursting discs are also stated.

12.2.4 Relief Disposal Systems

There are various ways of disposing of the products of relief, including using a flare system. This topic therefore connects with the work carried out on flare system design.

Disposal methods include the use of a scrubber for condensable mixtures, the products then going to atmosphere via a
vent stack, use of a closed flare system or a burning pit. Use of such pits depends on space, local conditions and economy. Another alternative is to send the relieved material to a low pressure system, providing an economical means of disposal. This is dependent on the receiving system being able to handle the additional load. Liquid from relief systems may also be disposed of in a number of ways. For example, water from coolers may be discharged to surface drainage. Hydrocarbons relieved from lines outside the process area should be discharged to a tank, closed vessel or sewer.

Atmospheric disposal of noncondensable vapour is possible provided that the equipment location is such that it is safe. Such discharge should be limited to those vapours which will not condense appreciably at low atmospheric temperature. Direct atmospheric relieving of valves is safe only if ignition can be tolerated with respect to hazards to personnel, structures and equipment.

12.2.5 Location and Position of Relief Valves

So that the valve can be maintained easily, pressure relief valves should be located for easy access and removal. Sufficient working space should be provided around the valve for this purpose.

The valve should also be close to the pressure source so that the valve will be 'fed' properly under flowing conditions, e.g. where a vessel is involved it is recommended that the valve be installed on top of the vessel. In cases where there are large pressure fluctuations at the source, the valve is usually located at a point where the pressure region is more stable. Where items such as reducing stations, orifice plates, flow nozzles and other valves and fittings are involved, it is recommended that the pressure relief valve be fitted some distance downstream of the device. Figures 12.1 and 12.2 show two typical pressure relief valve locations.
FIGURE 12.1 Typical pressure relief valve installation
12.3 Account of the Literature

Like the design of a flare disposal system, pressure relief system design is also the topic of many technical papers together with the codes that have already been described. In this section an account is given of some principal papers.

12.3.1 General

Fitt (1974) gives a good introduction on the process engineering of pressure relief and blowdown. He discusses the problem of excess pressure and ways of dealing with it. He states that a quantitative hazard analysis for each relief problem is impractical and a qualitative approach is more applicable. Alternatives for dealing with the problems of possible overpressure are given as:

- an inherently safer plant, i.e. stronger vessels
- instructing/equipping the operator to avoid or neutralize the hazard
- a high integrity trip system to shut off energy sources
- better emergency services
- limiting the consequences by excluding personnel from the danger area
- ignoring trivial or highly improbable hazards

Fitt then goes on to classify individual hazards into those which can be ignored, those where protection is needed, but normal relieving devices will suffice, and those which need a special design. Particular cases are then described including the shut-in pump or compressor, pressure letdown, a continuous still and low pressure storage tanks. Relief headers and the relief header design basis also receive attention. Fitt states that in designing a relief system, all the causes must be considered, and the single worst case is designed for.

Kauders (1981 and 1985) has written extensively on process engineering design in general and in particular
designing for plant upset conditions. His discussions are biased to the mechanical design of vessels, but include aspects such as pressure relief system design, vessel thickness, piping considerations and vessel depressuring.

Moore (1984) is also concerned with general aspects of relief. He states that the four main stages in the design process are:

1. Specification of the relief requirement
2. Selection of the relief device
3. Detailed design of the pipework/hardware
4. Designing for dispersion or treatment of the products of relief

Other aspects discussed include specifying the relief requirements, which can be done by a relevant safety code, taking into consideration the particular vessel or group of vessels, relief devices, pipework design, dispersion and a list of conventionally available safety valves.

Swift (1984) talks of the developments in the design of an emergency relief system for chemical reactors. He states how critical this is. Emergency relief systems are not called upon to function continually and the amount and type of data needed will depend on the particular design strategy adopted and data generated during the normal process design are not usually suitable, since they seldom cover extreme conditions. He proposes an algorithm for the elements of an emergency relief system design strategy.

12.3.2 Particular Cases

There are various accounts concerning pressure relief and blowdown applied to particular instances such as heat exchangers, distillation columns and storage tanks. Some of these aspects are described below.

Bradford and Durrett (1984) discuss aspects on the sizing
of safety relief valves for distillation columns. Apparently these are often under-sized, because the designer does not consider the actual maximum tower load. Factors considered in the sizing of such valves include tower flooding, reboiler capacity and condenser capacity. The question of alternative locations for relief valves is also discussed.

Crozier (1980) talks about the pressure relief to prevent heat exchanger failure. Such failures may be caused by water and steam hammer, vibration, erosion, overpressure and process upsets. Crozier then goes on to discuss various types of relief valves and discharge piping. Finally, he gives an example of the sizing of a relief valve for an ammonia vaporizer.

Air-cooled heat exchangers are the subject of a separate section in API RP 521 and are discussed by Brown and France (1975). Air-cooled heat exchangers can fail due to blocked outlets, louver failure or electrical failure. Methods for calculating the relief loads from blocked outlets are well established for air-cooled heat exchangers, but under electrical failure, when there is natural convection heat loss, methods are few and inaccurate.

12.3.3 Relief Devices

Most of the work covered in the general area of pressure relief and blowdown tends to be concerned with the different kinds of relief device available. Hodnick (1985) talks about four devices for unfired pressure vessels. These are rupture discs, safety relief valves, safety valves and combinations of safety valves and rupture discs. He states that the simplest relief device is the rupture disc. Further, the materials of construction are discussed which include stainless steel, inconel and nickel etc. An account is then given of the different types of relief devices available, and the use of a combination of rupture discs and safety valves in primary and secondary relief.
Puleo (1985) gives an account of the choice between relief valves or rupture discs. Factors considered are the loss of valuable fluid if a rupture disc is used, gradual pressure rises and rapid pressure rises. For rapid rises in pressure, a relief valve may not be adequate since it may not act quickly enough. In such explosive situations, rupture discs are far better.

Emerson (1985) categorizes valves into direct-acting or pilot-operated. Direct-acting types use a weight or spring to maintain valve closure to the set point, while pilot-operated valves use an unbalanced piston assembly in the main valve, with their position and type of operation controlled by a pilot.

12.3.4 Alternatives to Pressure Relief

One alternative to relief valves for pressure relief is the use of protective systems that isolate sources of pressure. Lawley and Kletz (1975) discuss this aspect. The reason for considering such an alternative is that as plants become larger so do relief systems and therefore costs. The use of stronger vessels may also avoid the use of a relief valve. In other cases, relief valves can be replaced by instrumented protective systems, or 'trip' systems. These detect a rise in pressure and shut off the source of pressure. Typically on a distillation column, the rise in pressure can be used to isolate the heat input to the base, and where runaway chemical reactions are a possibility a trip can isolate the supply of one of the reactants, avoiding the possibility of having to install a relief valve. Lawley and Kletz then go on to describe the design of a typical trip system for a fractionator. They state that an optimum trip system is one that yields adequate protection at minimum cost. The design is based on fail-danger fault rates of relief valves to obtain a fractional dead time for the trip system. Basic requirements are a high pressure sensor on the overhead vapour line, and a solenoid valve and associated trip valve.
12.4 Pressure Relief Design

There are many aspects, therefore, to the design of a pressure relief system, and in order to gain some idea of the expertise being deployed an attempt is made here to give an outline of the problem and a logical approach to it.

12.4.1 Alternatives

The first item to consider, after a system definition, is a decision on whether pressure relief is the most appropriate solution for overpressure protection. Alternative solutions are listed below :-

- Pressure limitation by design
- Pressure limiting instrumentation (trips)
- Pressure containment

Aspects used in the decision here will involve an awareness of the costs, likelihood of system failure, and codes of practice. The optimum solution may then be chosen.

12.4.2 System Definition

Assuming that the designer has chosen pressure relief as the viable alternative, it is then necessary to specify the following features for each relief device :-

- Location
- Set pressure
- Capacity
- Venting sink

We shall now draw together pressure relief aspects that were described above, and consider them under the following headings :-

- Pressure relief situations
- Overpressure sources
Pressure protection principles
Pressure protection - individual solutions
Pressure protection - consolidation and system solutions

These are now considered in turn, in terms of classification as opposed to a design strategy. The classification makes the problem of detailing a design strategy easier, and is really a structured summary of the aspects of pressure relief design described above.

12.4.3 Pressure Relief Situations

These are :-

Normal operational relief
Emergency operational relief
Fire relief

12.4.4 Overpressure Sources

There are many aspects here, but the main ones are :-

Extraneous components
  Water in hot oil
  Light hydrocarbon in hot oil
  Light components in distillation fluids

Chemical reactions
  Combustion
  Reaction runaway

Exposure to high pressure sources
  High pressure process fluids
  High pressure utilities
  Pressure raisers
  Heat exchanger high pressure side

Loss of low pressure sink

Heat input
  Burners
  Steam
  Heat transfer fluid
Atmosphere
Cooling loss
Water cooling
Air cooling
Heat transfer fluid
Reflux cooling
Subcooled feed
Thermal expansion (blocking in)
Fluid at ambient temperature
Fluid below ambient temperature
Fluid below temperature of another process fluid
Pressure transients
Water hammer (incompressible fluid)
Steam hammer
Fire

12.4.5 Pressure Protection Principles

It is not practical to design for the very worst case and so it is necessary to have some principles to select cases for design. Some of these principles might be:

- Single failure
- Utility loss cases
- Capacity credit
- Fire scenarios

Of the above, the term capacity credit needs explanation. In order to evaluate the relieving needs due to any cause, it is assumed that automatic control valves remain in the position required for normal processing flow. Unless the condition of flow through the valves changes, credit may be taken for the normal capacity of these valves, corrected to relieving conditions, providing that the downstream is capable of handling an increased flow. The decisions involved here involve evaluating response times and effects of controller settings, such as band, reset and rate.

The other principles listed above have received attention
12.4.6 Pressure Protection - Individual Solutions

It is possible to identify certain standard cases, or archetypes, that are the same in most design cases. Not all the cases have a solution, with prevention and awareness being important in these cases. Some of these are listed below:

- Water in hot oil - prevention not protection
- Light hydrocarbons in hot oil - protection not prevention
- Light components in distillation fluid - awareness
- Combustion - combustion venting - special case
- Reaction runaway - reaction venting - special case
- High pressure pipe - valve normally open or closed
- Pump, compressor - overdesign
  - low pressure side - non-return valve or special system
- Heat exchanger tube failure - pressure relief or special system
- Steam heated heat exchanger - awareness
- Cooling water failure - conventional pressure relief
- Fire - pressure relief - special case
- Pressure transients - damping chambers
- Reflux - pressure relief - special case
- Thermal expansion - pressure relief

12.5 Design Rules

The next step in gaining an insight into the design expertise is to write some rules for pressure relief design. These are described below.

12.5.1 Rules for Valve Selection

It has not been possible to identify rules for the design of a pressure relief system in the same way as flare system design. However, one area, that of relief valve selection is well suited to rules. One way forward here is for some sort of
automatic classification system, based on Ex-Tran7, as in the case of emergency isolation valves. This approach is possible because examples of valve installations could be written down in terms of properties such as material and temperature, and their values together with a class distinction such as 'safety relief valve', 'bursting disc' etc. Therefore, in this case distinctions need to be made about the types of valves in use, and thus classification expertise is involved. The following set of rules tries to capture these distinctions.

1. If inert materials are used and when shutdown and material loss can be tolerated, then a rupture disc may be suitable

   Puleo (1985)

2. If toxics are used and the system is venting to a flare system and leakage and shutdowns can be tolerated, then a rupture disc may be suitable.

   Puleo (1985)

3. If additional protection is needed, then use a rupture disc in series with a relief device.

   Puleo (1985)

4. If the temperature is high, then a safety relief valve should not be used.

   Andrew and Stockton (1979)

5. A safety relief valve should be used for gas/vapour service.

   Andrew and Stockton (1979)

6. If the system is in liquid service, then use a relief valve.

   Andrew and Stockton (1979)

7. If a valve is likely to lodge open with dirt, then use a bellows sealed relief valve.

   Bright (1972)
8. If freezing fail open is likely to be a problem, then use a pilot operated relief valve with no guiding surfaces.

Bright (1972)

9. If leakage due to corrosion or foreign body is likely to be a problem, then use either a resilient seated valve or use a rupture disc under the relief valve.

Bright (1972)

10. For storage vessels in a hot climate, a pilot operated relief valve is recommended so as to open enough to relieve the demand.

Bright (1972)

12.5.2 Rules for Pressure Relief in General

As above, the rules are written largely in production system format as this is the type of system envisaged for design of a pressure relief system. Most of the rules derive from the API design codes, and they are derived in the same way as was used in the work on flare systems.

11. A relief valve should be set to open at the MAWP of the vessel.

Bright (1972)

12. If there is more than one pressure relief valve, then the set pressures should be staggered.

Bright (1972)

13. If block valves are not sealed open or locked, then pressure relief is needed.

14. If overfilling is likely to be a problem on a vessel, then a warning to the operator is needed, and pressure relief if the vessel is likely to become full, designed for the maximum liquid pump in rate.

15. If pressure relief is needed, then design the system for
the condition(s) that require the most relief.

16. Relief may not be needed on all vessels, only those which can be isolated from the material sink.

17. If a vessel cannot withstand atmospheric pressure, and it is likely to be subjected to a vacuum, then vacuum relief is needed.

18. If closed outlets are a problem, then design for maximum liquid pump in rate (liquids) or total steam and vapour input under normal operation (vapours).

19. If cooling water failure to the condenser is a problem, then design for total vapour and steam entering less the vapour condensed by reflux.

20. If top tower reflux failure is a problem, then design for total vapour going to the condenser.

21. If sidestream reflux failure is a problem, then design for the difference between the vapour entering and leaving the system.

22. If accumulation of non-condensables is a problem, then a safety relief valve is needed designed for total steam and vapour entering under normal operation less the vapour condensed by sidestream reflux. In the case of vessels handling liquid design for the maximum liquid pump in rate.

23. If failure of automatic controls is a problem, and there is a low pressure controller, then design for total normally uncondensed vapour.

24. If failure of automatic controls is a problem, then it is not necessary to design for 'all valves closed' failure mode except where water and reflux are used.

25. If fired heaters and/or steam reboilers are in use, then
design for the maximum vapour generated and the non-condensibles from over-heating.

26. In the case of split reboiler tubes, then the requirement is for steam entering an area of 2 times the cross-sectional area of one tube.

27. If designing for abnormal chemical reactions, then use an estimate of the amount of vapour generation from both normal and abnormal operation.

28. If designing for power failures, then design for the worst possible case (dependent on local conditions)

29. If a fractionator needs relief, then design for the total vapour entering and that generated under normal operation.

30. If a reactor needs relief, then design for the product generation from a runaway reaction.

31. If air-cooled heat-exchangers need relief, then size the valve for the difference between normal and emergency flow.

32. If a surge vessel needs relief, then design for the maximum liquid inlet rate.

33. If a vessel is in need of protection and there are no fluctuations in pressure, then fit the valve to the top of the vessel.

34. If fluctuations in pressure arise at the source, then fit the pressure relief valve at a point where the pressure is stable.

35. If valves and fittings are in close proximity to the pressure source, then fit the valve at a point some distance downstream of the device.
12.5.3 Rule Types

As with the rules for flare systems, certain types of rule emerge from the list above. Rules 1-12 are all rules of alternative relief valves which make distinctions about the properties of the different valves available. As we stated above, these look well suited to an induction type of system, involving classification. The rules in their action suggest which type of valve should be adopted for particular installations. Rules 13-17 are all rules giving an indication to the solution to adopt for the overpressure problem. At this stage in the design no more information is needed apart from system definition and suggested solution strategy. These rules indicate this. The rules 18-32 are of a similar but more specific type. It is assumed by now that pressure relief is the adopted solution strategy, along with certain scenarios for certain kinds of equipment. Rules 18-24 are indications of a solution for certain overpressure situations such as accumulation of non-condensibles. Rules 25-32 are indications towards a conventional solution for archetypes such as fractionators and heat exchangers. Rules 33-35 take the design further by detailing the location requirements of relief valves. The rules are indications of a solution from various alternatives from given conditions.

One point to note here is that the rules are specialised rules. This means that not all the rules will apply to all situations, so the designer selects the rules which apply to his particular installation.

At this stage in the study it is necessary to give an indication of the design strategy employed by experts. This has been difficult to formulate for several reasons. Typically the literature gives little indication of the process of designing a pressure relief system, and experts themselves probably go about the design problem in a subconscious manner. For these reasons it was decided to carry out the initial design of a pressure relief system for a typical plant. Before the study itself, a description of the theory of the first
stages of pressure relief design must be given. In this study, it was not possible to have direct access to an expert, but some discussions were held by Professor Lees with an expert in industry, and some of the discussion is used below.

In order to reduce the problem to a manageable size, it is necessary to break the plant down into sub-systems, i.e. a decomposition is necessary.

12.6 System Decomposition

Often, if the problem can be formulated correctly, the solution becomes fairly straightforward. Therefore, if a decomposition is correct, the rest of the design problem will be simplified. It plays an important part in this particular aspect of design. In this case a decomposition is essential, and it is difficult to do without considering in detail various pressure sources.

Decomposition is a recognized method of breaking down a plant diagram into component parts. The method used determines the representation ultimately obtained. Decomposition can be done at varying levels of detail. The main idea of decomposition for pressure relief design is to show groups of vessels in well defined overpressure cause situations.

The principle used in the pressure relief survey is that of pressure breaks. The sections of plant between two breaks are at a more or less uniform pressure.

It is necessary at this stage to define pressure breaks. A pressure break is a point at which there is a large rise or fall in pressure. A section of plant between two breaks will tend to be at a more or less uniform pressure. Such a break is not necessarily declared at every point where pressure is let down, particularly if the pressure fall is trivial. Generally certain broad pressure bands are declared and used.
Figure 12.3 Benzene Plant Line Diagram
A line diagram of a benzene plant was readily available, so it was decided to use this for our own relief survey. The diagram is shown in Figure 12.3. From the initial study on the design of a pressure relief system, an expert designer will first go about a plant decomposition. The aim is to decompose the system into manageable components for pressure relief and suggest some rules for decomposition together with possible causes of overpressure. Firstly, the plant is described by the line number, vessels, pressure and control valves. These are listed below. Our default assumption at this stage is that all vessels in the plant need protection against operational overpressure. Also assumed is that the whole plant will need protection against fire.

12.7.1 List of Vessels

There are two regions of pressure that can be identified for pressure breaks. These are:

1.1 to 2.9 bar and
20.4 to 24 bar

The first stage is to list and describe all vessels together with their inlet and outlet pressures. This is listed in the following table below. Some of the lines have been split into sub-lines, as indicated in Figure 12.3 and shown with suffixes (a), (b) or (c). This makes the task of identifying pressure breaks easier, and the break can be pin-pointed to a particular item of equipment.

<table>
<thead>
<tr>
<th>Line No.</th>
<th>Control Valves</th>
<th>Vessels</th>
<th>Pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inlet</td>
<td>Outlet</td>
<td>Inlet</td>
</tr>
<tr>
<td>1 (a)</td>
<td>PCV</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>PCV</td>
<td>TK-101</td>
</tr>
<tr>
<td>Line No.</td>
<td>Control Valves</td>
<td>Vessels</td>
<td>Pressure bar</td>
</tr>
<tr>
<td>---------</td>
<td>----------------</td>
<td>---------</td>
<td>--------------</td>
</tr>
<tr>
<td></td>
<td>Inlet</td>
<td>Outlet</td>
<td>Inlet</td>
</tr>
<tr>
<td>2 (a)</td>
<td>-</td>
<td>-</td>
<td>P-101A/B</td>
</tr>
<tr>
<td>(b)</td>
<td>PCV</td>
<td>-</td>
<td>P-101A/B</td>
</tr>
<tr>
<td>(c)</td>
<td>-</td>
<td>PCV</td>
<td>E-101</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
<td>C-101</td>
</tr>
<tr>
<td>4</td>
<td>flow</td>
<td>flow</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>E-101</td>
</tr>
<tr>
<td>6</td>
<td>-</td>
<td>-</td>
<td>H-101</td>
</tr>
<tr>
<td>7</td>
<td>-</td>
<td>-</td>
<td>R-101</td>
</tr>
<tr>
<td>8 (a)</td>
<td>-</td>
<td>PCV</td>
<td>R-101</td>
</tr>
<tr>
<td>(b)</td>
<td>PCV</td>
<td>-</td>
<td>C-101</td>
</tr>
<tr>
<td>9</td>
<td>-</td>
<td>-</td>
<td>E-106</td>
</tr>
<tr>
<td>10</td>
<td>-</td>
<td>-</td>
<td>E-101</td>
</tr>
<tr>
<td>11</td>
<td>-</td>
<td>-</td>
<td>E-103</td>
</tr>
<tr>
<td>12 (a)</td>
<td>-</td>
<td>-</td>
<td>E-102</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>-</td>
<td>D-101</td>
</tr>
<tr>
<td>13</td>
<td>-</td>
<td>-</td>
<td>D-102</td>
</tr>
<tr>
<td>14 (a)</td>
<td>PCV</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>PCV</td>
<td>-</td>
</tr>
<tr>
<td>15 (a)</td>
<td>-</td>
<td>-</td>
<td>E-103</td>
</tr>
<tr>
<td>(b)</td>
<td>PCV</td>
<td>-</td>
<td>E-103</td>
</tr>
<tr>
<td>(c)</td>
<td>-</td>
<td>PCV</td>
<td>T-101</td>
</tr>
<tr>
<td>16 (a)</td>
<td>-</td>
<td>-</td>
<td>E-104</td>
</tr>
<tr>
<td>Line No.</td>
<td>Control Valves</td>
<td>Vessels</td>
<td>Pressure</td>
</tr>
<tr>
<td>---------</td>
<td>----------------</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>Inlet Outlet</td>
<td>Inlet Outlet</td>
<td>bar</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>D-103 E-104</td>
<td>2.0</td>
</tr>
<tr>
<td>17 (a)</td>
<td>-</td>
<td>P-101A/B D-103</td>
<td>2.9</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>P-102</td>
<td>2.9</td>
</tr>
<tr>
<td>18 (a)</td>
<td>PCV</td>
<td>-</td>
<td>2.9</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>PCV T-101</td>
<td>2.9</td>
</tr>
<tr>
<td>19 (a)</td>
<td>PCV</td>
<td>-</td>
<td>2.6</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>PCV E-105</td>
<td>2.6</td>
</tr>
<tr>
<td>(c)</td>
<td>-</td>
<td>E-105</td>
<td>2.6</td>
</tr>
<tr>
<td>20 (a)</td>
<td>PCV</td>
<td>-</td>
<td>2.3</td>
</tr>
<tr>
<td>(b)</td>
<td>-</td>
<td>PCV E-1</td>
<td>2.3</td>
</tr>
<tr>
<td>(c)</td>
<td>-</td>
<td>E-107</td>
<td>2.3</td>
</tr>
</tbody>
</table>

12.7.2 Pressure Breaks

It is evident from the table that significant breaks in pressure occur across the pump P101A/B and the control valve in line 15. Therefore the following vessels can be considered as being at one pressure level for the purposes of pressure relief:

- Distillation unit (not including E-105) - T-101, D-103
- Storage tank - TK-101

The pressure differences are between about 1, 11, 18 and 24 bar for these vessels.

A pressure relief survey for this system would require the following principal overpressure scenarios to be considered:
High inlet pressure (including utility failure, reflux failure, etc.)

Blocked outlets
High heat input
Loss of cooling water
Tube bursts
Fire

More specifically, the distillation column, for example, can 'see' overpressures including reflux failure and utility failure. An awareness of the consequences of light hydrocarbons in hot oil is also needed, and also the effects of fire.

The archetypes of this system, i.e., those where standard relief will suffice are the distillation unit, the furnace and the storage tank.

Due to limitations of time and lack of expertise and data this example problem has not proceeded further. It has shown, however, how an initial relief survey may proceed and the realization of archetypes, standard cases and overpressure causes. Also, various rules for decomposition have been derived. These are described in the section below.

12.7.3 Rules for Decomposition

These have been derived by writing down the actions taken during the decomposition process.

1. If there are significant differences in pressure throughout the line diagram, then pressure breaks will occur

2. If pressure breaks exist, then they must be identified and the system decomposed into manageable portions for pressure relief design.

3. Each line in the diagram should be examined for significant changes in pressure (>5 bar).
4. Lines of similar pressure should be coded for identification of pressure breaks.

5. Lines which contain control valves, heat exchangers or similar should be sub-divided either side of the equipment using suffixes.

6. If there is a vessel or heat exchanger or similar that can be physically isolated, then it must have its own pressure protection.

7. If there is a group of vessels that can be isolated from the rest of the plant, then the group can be treated as one for pressure protection purposes.

8. If a pressure break has been identified, then ensure that all pipes and vessels etc. either side of the break are at a similar pressure.

9. Those vessels which are at a similar pressure between two pressure breaks can be grouped together as a subsystem and can be treated as one vessel for pressure relief design. The designer may choose to limit this to a maximum number of vessels, say six.

10. The working pressure of a vessel/subsystem should be slightly below the design pressure of the vessel/subsystem.

11. If there is a distillation unit present on the plant, then treat it as an individual system.

12. If there is a vessel fitted with a heating element or other element which might cause pressure in the vessel, then the protecting device should be adequate to prevent the increase.

12.8 Pressure Relief Design Strategy

The following discussion on the design of a pressure
relief system is a contribution towards the understanding of how an expert designer goes about such a design. The account is based on discussions with experts in the field.

The relief survey starts with the Piping and Instrumentation diagram. This incorporates the main design decisions. Essentially this is accepted, though the relief designer may ask for a few changes.

A relief survey is then done. The system is decomposed into subsystems; this is done on the basis of what is 'manageable'. If the decomposition is too coarse or too fine, points may be missed, especially on interactions. Typically a subsystem is a functional unit such as a distillation or absorption/regeneration unit, with up to half a dozen vessels.

In the example shown in this study, decomposition has been done on a coarse scale on the basis of pressure breaks in the system. For fire relief a different and finer subdivision is used. In this case it is necessary to treat as a separate subsystem any vessel/pipe which can be shut in by an isolation valve, control valve, non-return valve or an emergency shutdown valve.

The designer typically starts with a listing of utility failures, which leads to consideration not only of cooling but also of control loop actions. He may use a set of standard utility loss cases. Next, open inlets, closed outlets and fire are considered. With fire, a set of standard cases may be used. Finally, situations such as tube burst and thermal relief are considered.

12.8.1 Pipework

The piping pressures used are a discrete set and are usually related to the ASA standard pressures for flanges, which are 150, 300, 450, 600, 900 and 1500 psig. In this set flanges are designed to withstand saturated steam at these pressures and the corresponding temperatures. The minimum
pressures used for pipework appears to vary between different companies. An oil company may standardise on 15 bar, whereas a chemical company using more expensive materials of construction may choose a lower minimum pressure such as 6 bar.

12.8.2 Vessels

There are also minimum design pressures for the main vessels on a plant. A value of 3 bar is typical due to the mechanics of the system. However it is also necessary to have a high enough pressure for effective pressure relief, and a value of about 5 bar is needed for relief to a flare header. Design practice appears to be to design a vessel for a particular pressure and to provide pressure relief for that pressure, regardless of the design pressure of the associated pipework.

12.8.3 Outline of the Design Process

The following is an outline of a possible approach to the design of a pressure relief system:

- Decompose pressure system
  - Select subsystems
- Identify pressure breaks
- Locate pressure reliefs (default assumption: pressure relief is located on each main vessel)
- Identify relief loads on each main vessel
  - Utility failure
    - Open outlet (connection to high pressure sources)
    - Closed outlet (disconnection of low pressure sink)
  - Heat input
  - Cooling loss
  - Operating excursions
  - Equipment failure
  - Fire
- Quantify relief loads on each main vessel
  - Utility failure standard cases
  - Fire standard cases

294
Select relief sink
Other part of plant
Flare header
Other closed system
Atmosphere

Determine set pressure of relief valves
Determine capacity of relief valves
Select type of relief valves

The above list is fairly brief, but it does give an idea as to the kind of methods used by designers. One point to note is that there are many subsidiary topics associated with the topics above. These are outlined below.

12.8.4 Subsidiary Topics and Design Expertise

These are areas of pressure relief design that are indirectly involved, and the designer must therefore be aware of them when designing such a system. The list shows these, and also gives an indication of the expertise involved in the design.

Legal and standards requirements
Economics and alternative solutions
Alternatives to or mitigation of pressure relief
Pressure containment
Pressure limiting instrumentation
Pressure limitation by design
Mitigation of fire relief
Fire proofing
Water drench
Ground slope
Depressurisation
Minimum venting pressure
Flare header
Atmosphere
Selection of disposal systems
Atmospheric disposal
Disposal criteria
Safe disposal
Pressure valve design and location

Special conditions
Corrosive fluids
Cold fluids

Unit operations and equipment
Furnaces
Distillation Columns
Shell and tube heat exchangers
Pressure storage
Steam systems
Pipelines

The design therefore has many aspects and is one of great expertise and probably deep structure. E.g. the study of flare system design may form just one part of the design process for pressure relief design. Knowledge of codes and legislation is required, rules are used and expertise of different types is employed. The expertise is in the form of an awareness of many different aspects such as distinctions, alternatives, archetypes, overpressure scenarios, conventional solutions and classifications. Also required is an appreciation of special cases such as combustion and runaway reaction together with a knowledge of probability regarding the likelihood of a particular situation arising.

The design of a pressure relief system seems to be one of deeper structure than that of a flare system. This is evident from the fact that a decomposition of the system is needed, whereas it is trivial for flare systems. The study suggests that after the decomposition stage, a more difficult area of the design has to be done. We have done work on the initial stages in design, and also on the final aspects, i.e. relief valve selection. It has been difficult to work out a design strategy for the middle section of the design, which includes flow determination. If the decomposition of the system is correct, then it is a fairly straightforward assumption that one relief system is needed per grouping. The actual stages in the design therefore seem to be :-
Work still needs to be done on flow definition.

Another similarity with the flare system work is that there is no obvious need for iteration in the actual design process. It is interesting to note that although iteration is a characteristic of design, the need for iteration has not come through in this work. One reason for this is possibly the use of a fairly comprehensive decision procedure and set of rules.
13 Problem Characteristics

13.1 Introduction

In previous chapters we have discussed the design problem generally and specific problems in the design of plants handling hazardous materials. We have attempted to show various problem-solving strategies for these topics, and it is now possible to establish the characteristics of the design problems and propose an outline of design generally with the techniques of artificial intelligence in mind. This has been possible by studying the type of expertise and problem-solving strategies for each topic studied.

In this chapter, we will start by giving a brief summary of the candidate topics studied, the expertise involved and the problem solving strategies employed or suggested. The study of the kind of expertise involved is an important aspect, since it gives an indication as to what type of problem-solving strategy is the most appropriate. We can also compare the methodologies of design, particularly that of Simon (1981), given in Chapter 6 with the characteristics drawn out here.

13.2 Problems Studied

We have used various strategies for solving the topics studied in this work and it is fairly evident that design problems as a whole need these strategies if any viable contribution is going to be made from expert systems in this field. In this section a summary of the nature of each problem is given together with a description of the tools used to solve the problem. In this way it is possible to match the nature of each problem with the strategy chosen for solving it.

13.2.1 Fire Protection of Storage Tanks

This topic was the first to be studied in the work, and
was used as a learning exercise. It was concluded that it is not an area of great expertise, and that perhaps it is an algorithmic problem. However, we have demonstrated a Prolog knowledge based system which suggests ways of protecting storage installations from fire. The knowledge for the program came largely from codes of practice, although some discussion with an expert did take place. The initial approach, in order to codify the knowledge was to make an algorithmic logic diagram, which was shown in Chapter 9. The diagram makes distinctions and classifications on various aspects to do with storage installations, including material type, the type of tank and the size of the installation. The resulting Prolog program has in its knowledge base rules derived from the diagram. The user then enters into a consultation with the program, answering various questions so that the program could establish the type of storage being considered, and the resulting fire protection.

The program written is not an expert system, for reasons outlined in Chapter 9. The problem, as we indicated, is probably an algorithmic one, but a Prolog program is a valid option in order to have the knowledge readily at hand. The program used forward chaining since a goal is concluded, i.e. that of fire protection, by considering the data available.

Another solution strategy was tried, that of classification by type of protection. It was concluded that fire protection for storage tanks tend have fairly unique solutions, which results in no classification of the traditional kind.

The topic is therefore characterised by the use of mainly algorithmic expertise, although some classification is also evident. The knowledge used is straightforward, with seemingly few ambiguities, making for a clear-cut solution strategy. Forward chaining appears to be the most appropriate because the solution for safe storage is reached by considering the data available, whether using a computer approach or by designing the installation manually.
13.2.2 Emergency Isolation Valves

The problem of whether or not to fit an emergency isolation valve is dependent on various factors, as outlined in Chapter 10. There, it was shown that fitting such a device is costly, and therefore the chances and consequences of a leak must be carefully evaluated. Factors considered in the decision include past leak history, temperature of the material, and the probability of a leak including its consequences.

The problem was tackled using data from an olefins plant and the rule induction method derived by Quinlan. The data available was in a format suitable for such a method, and use was made of the expert system building tools Expert-Ease and Ex-Tran7. These methods rely on examples of the domain together with a class distinction, which in this case was the binary split 'fit an EIV' or 'do not fit an EIV'. After running the examples through the software a rule was induced based on the most discriminatory attribute in the example set. The rule subsequently induced contained 11 nodes, with the attribute 'large hazard' (i.e. the item is less likely to leak, but if it does so, could release a large quantity with no way of stopping it) as the one which gave the most information.

In order to use the rules established in a more comprehensive consultation system, and so that information on rule strength and uncertainty in attribute values could be included, the expert system shell Micro-Expert was used. This system relies on the programmer to have at hand a good set of rules to code into the advice language, and since the rule induction method induced the best possible rule in the light of the data available, there was no need to derive the rules by hand. The method used for coding the induced rules depends on certain hypotheses being defined. These included HISTORY, HAZARD, and TEMP. Also, factors which indicate the strength of a rule were needed. Again, the induced rule helps here, since the attributes are listed in order of decreasing information...
gain. However, the logical factors finally used in the Micro-
Expert system were derived by defining the factors on the
basis of the induced rule, and then refining them during
subsequent testing of the system. The resulting system in-
cluded questions such as:

How certain are you that the hazard is large [-5..0..5]?

The user has then to decide whether his particular installa-
tion definitely has a large hazard, in which case he answered
'5', or whether there was some doubt.

The system was shown in Chapter 10 to work well through
the use of example consultations, and the work on emergency
isolation valves as a whole has shown one method of building a
consultative expert system from domain examples.

This problem is one of pure classification, and in this
work it has been shown how expert system tools can be used to
take the examples and use them to their full potential.

At this stage in expert system development as a whole, we
have also indicated a rather unusual method of building an
advice system by first inducing automatically the best
possible rule.

The expertise in this problem is limited to
classification and some of the problems inherent in using a
suitable tool for this type of problem have also been
highlighted.

13.2.3 Flare Systems

The problem of flare system design is certainly more
complex than the two problems described above, this being
apparent from the study carried out in Chapter 11.

The first stage in this work was to become familiar with
current awareness in flare system design, and for this a flare
system awareness aid was written. From this it became apparent that a rule-based approach was an appropriate way forward and after an extensive review of the literature, rules were derived for flare system design. The knowledge came largely from technical papers, of which there is an abundance in this field. Use was also made of an expert, and a transcript of this appeared in Chapter 11. Care was taken in the rules derivation to note the author and the date of the work, as this is important when evaluating the strength of a rule.

It was evident from the rules summary that certain types of rules exist for flare system design. These included prohibition rules, suggestions of alternatives, suggestions to a solution and typical numerical design values. It was noted that exceptions would occur to prohibition and alternatives rules. The rule types were then classified into explicit and elusive expertise. As a result of the study of rules and the literature, a design strategy and design expertise were established.

A Prolog program was also written for the design of certain aspects of the flare system problem. The design strategy indicated that a step wise approach would be suitable together with a forward chaining solution strategy. The program written was based on Winston's BAGGER problem which was a forward chaining production system. The rules for flare system design were written in the appropriate format, which included the following steps:

- check flare items
- determine flare type
- determine flare layout
- etc.

The output from the program was a list of conditions for certain aspects of flaring. The program also showed conflict resolution at work. This is an important area of design work, and is likely to be used more extensively in the future.
The expertise established for flare system design was not as explicit as in the two topics described above. This reflects the nature of the problem and the more comprehensive rules and solution strategy adopted. The expertise is in the form of awareness of alternatives, costs, probabilities, archetypes and distinctions. The design strategy indicated a forward moving linear activity, with little iteration apparently required.

13.2.4 Pressure Relief and Blowdown

The design of a pressure relief and blowdown system is probably the most complex area tackled in this study. The main source of knowledge came from the API codes of practice together with technical papers relevant to this area. Due to factors outlined in Chapter 12, no programming was undertaken, but an example problem was carried out which highlighted some of the problems faced by the designer of a pressure relief and blowdown system.

The study began with a description of the legal requirements of any system, together with some definitions of the various types of relieving device available and generic terms used. It was necessary, in order to understand the problem, to establish the causes of overpressure and how to determine individual relieving rates. Like the flare systems work, an account of the literature was given. This showed that rules for pressure system design did not exist in the same way as that of flare systems and that design strategies as such were going to be difficult to formulate. One area where rules were shown to be appropriate was that of relief valve selection and a rule induction strategy was suggested as a viable problem solving tool. This aspect of pressure relief design is, however, the most straightforward, with the most difficult area that of specification of relief requirements and determination of flows.

A logical design approach was suggested, which included alternatives such as pressure limitation by design and
pressure containment. Pressure relief situations included operational relief, emergency relief and fire relief, and various overpressure sources, typical to most plants were proposed.

As stated above, two sets of rules were derived, that for pressure relief valve selection and pressure relief design in general. These rules were again of various types, including suggestions to a solution, and rules indicating relief valve location.

The example problem carried out was on a benzene plant line diagram, and the major area tackled was that of system decomposition. It was shown that decomposition is probably one of the most complex tasks in this area, and once a good decomposition has been established, the solution becomes better defined. Rules were written for the decomposition task, and a decomposition was proposed for the benzene plant line diagram.

In the summing up, three major tasks were highlighted for the design of a pressure relief and blowdown design. These were:

Decomposition
Definition of flows
Choice of relief device

Of these, the first and the third have received attention in this work, with the definition of flows being very much dependent on the particular situation. The design strategy outlined in Chapter 12 indicated a forward moving approach, but was also apparently one of deeper structure than the other problems tackled. The expertise is similar to, although more involved than, that of flare system design.

13.2.5 Hazard Identification

Hazard identification was a study of the literature in order to identify expertise and problem-solving techniques.
The two aspects, coarse and fine scale, were found to be quite different in nature.

Coarse scale hazard identification involves design expertise, and a data-driven, forward chaining strategy seems most appropriate for aspects such as check lists. Knowledge representation in the form of frames also seems to be well suited here.

Fine scale hazard identification would be suited to the techniques of fault propagation with an expert system around it. Some problems exist in trying to automate a HAZOP study, for example, and these have been discussed in the text.

13.3 Summary Table

It is worthy at this stage to summarize the topics studied and the problem solving tools used in the form of a table. This is shown below.

<table>
<thead>
<tr>
<th>Topic Studied</th>
<th>Type of Problem</th>
<th>Search Strategy</th>
<th>Generic Tool</th>
<th>AI Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fire protection of storage tanks</td>
<td>Algorithmic</td>
<td>Forward chaining</td>
<td>Prolog</td>
<td></td>
</tr>
<tr>
<td>Emergency isolation valves</td>
<td>Classification N/A</td>
<td>ID3</td>
<td>ExpertEase</td>
<td>Ex-Tran7</td>
</tr>
<tr>
<td>Flare systems</td>
<td>Design with linear structure</td>
<td>Forward Production</td>
<td>Prolog</td>
<td></td>
</tr>
</tbody>
</table>
13.4 Problem Characteristics

In this section we discuss the characteristics of the problems we have studied together with some general aspects of the characteristics of design.

13.4.1 Characteristics of Problems Studied

We have looked at a number of different topics in the field of the design of plants handling hazardous materials with the aim of gaining a better understanding of the expertise involved and of the design heuristics use by expert designers.

There is no doubt that the topics all embody some form of expertise, albeit different in nature, with varying levels of skill. Some topics, and in particular the more straightforward ones, involve just one form of expertise, whereas other more complex topics embody many types. The expertise has been identified as being either classification, algorithmic, or a deeper type which has become apparent by studying more complex design situations. In the complex cases there is the expertise of decomposition. More straightforward design problems tend to have either a classification solution or there is an algorithm to follow, while those requiring a deeper level of expertise tend to be of the design type where decomposition plays an important part.

The expertise of a problem is reflected in the type of solution strategy adopted. For problems involving classification, rule induction methods appear to be well suited. The rules induced can, as we have shown with the emergency isolation valve problem, be used in an advice giving system. For problems involving algorithms the direction of search which seems most appropriate is forward chaining, and this is also
true of some problems involving deeper levels of expertise. Forward chaining strategy has played an important part in this work, and it is interesting to recall a comment from chapter 5 concerning expert systems. There, it was stated that the majority of serious systems were forward chaining.

13.4.2 Characteristics of Design

By studying various design topics and possible solutions to them we have been able to draw out some characteristics general to design.

One of the main criteria in artificial intelligence problem-solving is that the problem domain be represented in a state space, consisting of goals and known facts and that there be a search procedure for obtaining the goal state. We can see from this study that design problems are well represented in this form and therefore require solution strategies to be adopted. In the state space will be alternatives to the problem solution and methods must be employed in order to find the optimal solution. Computational methods would therefore involve algorithms for choosing alternatives and algorithms and/or heuristics for choosing a satisfactory solution. The search for alternatives involves a search along possible paths storing in the system memory a tree of the paths it has explored. At the end of each branch is a number which gives an indication of the expected benefit from further search. Search has the additional advantage that it gives useful information about the structure of the problem.

By nature, any state space search will involve the satisfaction of goals. This is also an inherent nature of design problems, and therefore design is a goal seeking process. This also implies that logic will play an important part in design. Simon states that standard logic is concerned with declarative statements and is well suited to assertions about the environment and to inferences from those assertions. We have shown through the use of Prolog, which is about satisfying goals, that design can be handled in this way. Prolog has another
advantage that is essential to design, that is backtracking. Simon also indicates that there are two aspects to goal-seeking systems. The first is to do with gaining information about the world and the second is to do with acting on the world. Criteria necessary here are for a store to retain the information and for information about actions. This, in relation to this study fits in well with a condition-action forward chaining system, firstly by gaining information to satisfy the conditions and then by firing the actions. These are characteristic of production systems and it therefore seems that a production system would be well suited to most design topics, providing that the knowledge can be represented in the form of production rules. This also has another advantage. Design, as we have seen, is characterized by the fact that there are often conflicts to resolve. Production systems are well suited to conflict resolution strategies, and this would therefore present little problem.

13.4 Expert Systems for Design

Design shares many of the characteristics of general problem-solving processes, and many characteristics shared by the problems that existing expert systems tackle. However, there are very few expert systems that are concerned with design, and only one, RI is in commercial use. This reflects the difficulty of representing design problems in a state space and of representing the knowledge in a suitable form. It is hoped that the characteristics of design developed here will make some contribution to the problem. The following is a discussion on the way expert systems could be used in design.

The generation of a design can be achieved through the firing of rules to transform the initial state to a final state by rule-generated operations. Such a system would use a combination of forward chaining and backtracking in order to simulate the design process. By itself, however, an exhaustive search would take place, which is not a real world situation. This could be overcome by the use of efficient heuristics.
14 Conclusions

The objectives of this project were to study the expertise in the design of plants handling hazardous materials and to suggest new and improved methods for computer aided design by studying expert systems, and by studying various design topics in relation to expert systems.

14.1 Characteristics of Candidate Topics

Two of the problems studied have been identified as classification problems. They are emergency isolation valves and pressure relief valve selection. Where examples of the domain exist, automatic rule induction holds much promise, particularly when programmed into an advice system which can handle probabilities and uncertainty.

A lot of the problems in design are probably classification type in nature. These are sometimes straightforward, but it may not always be sufficient to use a classifier to complete a problem solution, as shown by the emergency isolation valve problem.

Of the non-classification problems, such as that of fire protection of storage installations, also make classifications and distinctions, and it has been shown that a forward chaining problem solving strategy is most appropriate.

Of the more complex problems tackled, the design of a flare system is a fairly linear problem which proceeds in series and is probably without deep structure. The main characteristic to come out of this area of the work is that the quality of the rules is very important. There is much information available, with some of the rules conflicting. There is also much expertise in the literature which is not easy to codify into rules.

The program written for flare system design has a simple control strategy and this appears to be sufficient, although
there could be a problem in which rules to give priority to.

The study carried out on the design of a pressure relief system has shown that the problem has structural features. The design of such a system is in three parts. The first decomposition phase represents a large and most difficult part of the process. The second phase is that of determining the relief flows, and although not explored in detail, it may be a more complex part than those we have tackled. The final phase is that of valve selection, and this has been shown to be a classification type problem.

14.2 Characteristics of Design

By studying various topics in design it has been possible to establish characteristics of design. It has been shown that design is an area of much expertise, which comes out in various forms and types. Expertise can be algorithmic, classification, or it can be in the form of awareness and criteria for decision making.

Also, it has been shown that problems in design are well suited to the problem-solving methods used in artificial intelligence.

14.3 Expert System Tools

Expert system tools used in this work have been classification systems, advice systems and production systems. All have shown promise in one aspect of design, although production systems seem to offer the most potential. Semantic nets and frames are a possibility for hazard identification. This indicates that before any particular aspect of design is considered for an expert system approach, a thorough knowledge of the domain, problem-solving processes and of the expert's typical way of proceeding is necessary.

This study has shown the potential that expert systems hold for computer aided design. Much work still needs to be
done, particularly in large and complex areas such as pressure relief and blowdown design. It has, however, been shown through the application of problem-solving methods applied to a wide range of design problems that such an approach is worthwhile. It should also be noted that not all design problems can be solved using just one type of strategy.

It has been shown, therefore, through the study of expert system techniques and various aspects of the design of plants handling hazardous materials, that the methods used to solve such problems share much in common with the problems addressed by the tools used in artificial intelligence. It may therefore be concluded that expert systems are a viable and promising aid to the designer of plants handling hazardous materials.

14.4 Highlights of the Study

1. Classification - a large part of design, but not necessarily straightforward.

2. Classification tools - can be used to induce rules for an advice system which allows the use of probabilities and certainty factors. These are not easily handled in a classification system.

3. Non-classification - design problems of this type can probably be handled by a production system.

4. Production systems - the priorities and the quality of the rules needs careful attention.

5. Decomposition - is likely to be crucial in some design problems in order to reach a solution.
15 References

15.1 Chapter 1


12 Simon, H.A. (1975) A Student's Introduction to Engineering

15.2 Chapter 2

14 Andow, P.K. (1985) Private Communication


35 Sirola, J.J. et.al. (1971) Synthesis of System Design III: Toward a Process Concept Generator, AIChemE J., 17,


15.3 Chapter 3


38 Bobrow, D.G. et.al. (1977) GUS: A Frame Driven Dialog System, Artificial Intelligence, 8, p.155


315


15.4 Chapter 4

49 Andow, P.K. (1984) Expert Systems Short Course Notes, Department of Chemical Engineering, Loughborough University of Technology


15.5 Chapter 5


15.6 Chapter 6


15.7 Chapter 7


94 Morgester, J.J. et.al. (1979) Control of Emissions from Refinery..., Chem.Engng.Prog. 75, p.40


15.8 Chapter 8


110 Kelly, B.E. and Lees, F.P. (1986) The Propagation of faults...


15.9 Chapter 9

120 BS 2594 (1955) Horizontal Mild Steel Welded Storage Tanks (London: BSI)


122 BS 5500 (1976) Unfired Fusion Welded Pressure Vessels (London: BSI)


124 ICI (1972) Electrical Installation in Flammable Atmospheres IS/91 (Birmingham: ICI/RoSPA)

125 ICI/RoSPA (1970) Liquefied Flammable Gases, Storage and Handling IS/74 (Birmingham: ICI/RoSPA)


15.10 Chapter 10


15.11 Chapter 11


324


143 De Favari, D. et.al. (1985) Estimate Flare Radiation Intensity, Hydro. Proc., 64, 5, p.89


146 Kilbey, J.L. (1968) Flare System Explosions, Chem. Engng. Prog., 64, 6, p.49


150 Kletz, T.A. (1985b) Private Communication


325

155 Miller, P.D. et.al. (1958) The Design of Smokeless, Non-luminous Flares, API Proc. 38, p.276


160 Reed, R.D. (1972) What is Flares Proper Purge Rate, Oil and Gas J., 70, Feb 14, p.91


15.12 Pressure Relief


APPENDIX A

ID3 EXAMPLE
Consider the example given in section 6.2.2 after Quinlan. A set of objects, \( C \) were described by three attributes, each having various values:

- **Height** with values (tall, short)
- **Hair** with values (dark, red, blond)
- **Eyes** with values (blue, brown)

Two classes are possible, either '+' or '-'. The full set of examples is:

\[
\begin{align*}
\text{C} =& \text{short, blond, blue, +} & \text{short, dark, blue, -} \\
=& \text{tall, blond, brown, -} & \text{tall, dark, blue, -} \\
=& \text{tall, red, blue, +} & \text{tall, blond, blue, +} \\
=& \text{tall, dark, brown, -} \\
=& \text{short, blond, brown, -}
\end{align*}
\]

From section 6.2.2,

Expected information, \( M(C) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 \)

Consider the following diagram:
Therefore, let

\[ B(C, A) = (\text{probability that value of } A \text{ is } A_i) \times M(C_i) \]

The suggested choice of attribute to test is that which gains most information as

\[ M(C) - B(C, A) \rightarrow \text{Maximum} \]

Consider the choice of the first attribute test. 'C' contains 3 in class '+' and 5 in '-'.

Therefore, \[ M(C) = -(3/8)\log_2(3/8) - (5/8)\log_2(5/8) \]

\[ = 0.954 \text{ bits} \]

Now consider the 'height' attribute in terms of the following decision tree :-
The information still needed for a rule for the 'tall' branch is:

\[-(2/5)\log_2(2/5) - (3/5)\log_2(3/5) = 0.971 \text{ bits}\]

and for the 'short' branch:

\[-(1/3)\log_2(1/3) - (2/3)\log_2(2/3) = 0.918 \text{ bits}\]

Therefore the expected information content is:

\[B(C, 'height') = \frac{5 \times 0.971}{8} + \frac{3 \times 0.918}{8} = 0.951 \text{ bits}\]

Thus, the information gained by testing this attribute is:

\[0.954 - 0.951 = 0.003 \text{ bits}\]

For the 'hair' attribute, consider the following decision tree:

- For the 'tall' branch:
  - tall, blond, blue, +
  - tall, blond, brown, -
  - tall, red, blue, +
  - tall, dark, brown, -
  - tall, dark, blue, -

- For the 'short' branch:
  - short, blond, blue, +
  - short, dark, blue, -
  - short, blond, brown, -
The branches 'dark' and 'red' require no further information, but for the 'blond' attribute:

\[ B(C,\text{hair}) = 3*0 + 1*0 + 4*1 = 0.5 \text{bits} \]

\[ 8 \quad 8 \quad 8 \]

Therefore the information gained by testing this attribute is:

\[ 0.954 - 0.5 = 0.454 \text{bits} \]

For the 'eyes' attribute:
The information still needed to form a rule is, for the 'brown' attribute, 0 and for the blue attribute:

\[-(2/8) \log_2(2/8) - (3/8) \log_2(3/8) = 0.971 \text{ bits}\]

Therefore the expected information content is:

\[B(C, 'eyes') = 5 \times 0.971 + 3 \times 0 = 0.607 \text{ bits}\]

Thus, the information gained is:

\[0.947 - 0.607 = 0.347 \text{ bits}\]

Therefore the principle of maximising expected information gain would lead ID3 to select 'hair' as the attribute to form the root of the decision tree.
APPENDIX B

PROLOG BAGGER PROGRAM
Winston's BAGGER problem is a forward chaining system using if-then rules. It is a grocery bagging system that bags groceries in the manner of a grocery store check-out clerk. It does the fundamentals of packing, not the optimal packing. e.g. big bottles of Pepsi go in the bottom, with not too many in one bag, ice cream is protected with freezer bags, etc. BAGGER involves four steps, as shown by the following procedure description:

**BAGGER:**
1. Check to see what the customer has selected, looking over the groceries to see if something is missing, with a view toward suggesting additions to the customer.
2. Bag the large items first, with special attention to putting big bottles in first.
3. Bag the medium items, taking care to put the frozen things in freezer bags.
4. Bag the small items, putting them wherever there is room.

**Rules:**

**B1** If the step is check order  
there is a bag of potato chips  
there is no soft-drink bottle  
then add one bottle of Pepsi to the order

**B2** If the step is check order  
then discontinue the check order step  
start the bag large items step

**B3** If the step is bag large items  
there is a large item to be bagged  
there is a large bottle to be bagged  
there is a bag with < 6 large items  
then put the bottle in the bag

**B4** If the step is bag large items
there is a large item to be bagged
then put the large item in the bag

B5 If the step is bag large items
then start a fresh bag

B6 If the step is bag large items
then discontinue bag large items step
start the bag medium items step

B7 If the step is bag medium items
there is a medium item to be bagged
there is an empty bag or a bag with medium items
the bag is not yet full
the medium item is frozen
the medium item is not in an insulated freezer bag
then put the item in an insulated freezer bag

B8 If the step is bag medium items
there is a medium item to be bagged
there is an empty bag or a bag with medium items
the bag is not yet full
then put the medium item in the bag

B9 If the step is bag medium items
there is a medium item to be bagged
then start a fresh bag

B10 If the step is bag medium items
then discontinue the bag medium items step
start the bag small items step

B11 If the step is bag small items
there is a small item to bag
there is a bag not yet full
the bag does not contain bottles
then put the small item in the bag

B12 If the step is bag small items
there is a small item to bag
there is a bag not yet full
then put the small item in the bag

B13 If the step is bag small items
there is a small item
then start a fresh bag
B14 If the step is bag small items
then discontinue the bag small items step
stop

Grocery item (bread, plastic bag, medium, non-frozen).
Grocery item (glop, jar, small, non-frozen).
Grocery item (granola, card box, large, non-frozen).
Grocery item (sugar, paper bag, large, non-frozen).
Grocery item (ice cream, card carton, medium, frozen).
Grocery item (potato chips, plastic bag, medium, non-frozen).
Grocery item (pepsi, bottle, large, non-frozen).
Grocery item (wine, bottle, medium, non-frozen).
Grocery item (jam, jar, small, non-frozen).
Grocery item (orange juice, card carton, medium, non-frozen).
Grocery item (beer, cans, medium, non-frozen).
Grocery item (fish fingers, card box, medium, frozen).
Grocery item (soup, cans, small, non-frozen).

Bagger:
step (check order),
step (bag large items),
step (bag medium items),
step (bag small items),
!, fail.

Step (check order):-
item (potato chips),
item (pepsi).

item (X):-
  grocery item (X, _, _, _),
  assert (present (X)),
  nl, write (X), write (' present!'), nl, !.

item (_) :-
  nl, write (X), write ('item not present!'), nl,
  assert (item (X)).

Step (bag large items):-
large_item(X),
discover(bottle),
bad(large).

large_item(X):-
grocery_item(A,_,_,_),nl,
write('There are large items to bag'),nl,
write('----------------------------'),nl.

discover(X):-
grocery_item(A,bottle,_,_),nl,
write('Bagging '),write(X),nl,
assert(bagged_items(X)).

bag(X):-
grocery_item(A,_,X,_,),nl,
write('Bagging item'),write(A),nl,
assert(bagged_items(A)),
bag_full(A),
fail.

bag_full(bread):-
nl,tab(8),
write('******'),nl,
write('Start new bag!'),nl,
assert(new_bags(one)).

bag_full(beer):-
nl,tab(8),
write('******'),nl,
write('Start new bag!'),nl,
assert(new_bags(two)).

bag_full(_):-
nl,tab(8),
write('Bag not full!'),nl.

bag(_):-
nl,write('Items bagged!'),nl.
step(bag_medium_items):-
    medium_item(X),
    bagged(frozen),
    bag(medium).

medium_item(X):-
    grocery_item(A, _, medium, _), nl,
    write('There are medium items to bag'), nl,
    write('-------------------------------'), nl.

bagged(X):-
    grocery_item(A, _, X), nl, write(X),
    write(' item '), write(A),
    write(' put in insulated freezer bag'), nl,
    bag_full(A),
    assert(freezer_bag(X)), fail.

bagged(_):-
    nl, write('Frozen items bagged!'), nl.

step(bag_small_items):-
    small_item(X),
    bag(small).

small_item(X):-
    grocery_item(A, _, small, _), nl,
    write('There are small items to bag'), nl,
    write('-------------------------------'), nl.
York Portable Prolog Release 2.1.

Please pass on any comments on this Prolog to RSM Kirkwood via Multics Mail.

?- bagger3 consulted.
?- potato_chips present!
pepsi present!
There are large items to bag

Bagging bottle
Bagging item granola
  Bag not full!
Bagging item sugar
  Bag not full!
Bagging item pepsi
  Bag not full!
Items bagged!
There are medium items to bag

frozen item ice_cream put in insulated freezer bag
  Bag not full!
frozen item fish_fingers put in insulated freezer bag
  Bag not full!
Frozen items bagged!
Bagging item bread

******* Start new bag!
Bag not full!

Bagging item ice_cream
Bag not full!

Bagging item potato_chips
Bag not full!

Bagging item wine
Bag not full!

Bagging item orange_juice
Bag not full!

Bagging item beer

******* Start new bag!
Bag not full!

Bagging item fish_fingers
Bag not full!

Items bagged!

There are small items to bag

-----------------------------

Bagging item glop
Bag not full!

Bagging item jam
Bag not full!

Bagging item soup
Bag not full!

Items bagged!
no
?-  
[Leaving Prolog]
APPENDIX C

PROLOG PROGRAM LISTING FOR STORAGE TANK FIRE PROTECTION
/* ----------------------------------------------- ATMOSPHERIC STORAGE RULES -----------------------------------------------*/
rule('atm', 'size', '110 per cent bunds').

rule('atm', 'bunds', 'fit drain valves to bunds', 'normally closed position', 'check weekly').

rule('atm', 'layout', '15m minimum separation for tanks in one group from inside top of bund of any adjacent group').

rule('atm', 'boundary', '15m minimum separation for tanks to boundary or ignition source').

rule('atm', 'fixed_roof', 'no more than 60,000 cubic metres per bund').
rule('atm', 'floating_roof', 'no more than 120,000 cubic metres per bund').
rule('atm', 'fixed_roof', 'less_than_9m_diam', ...).
rule('atm', 'floating_roof', 'less_than_9m_diam', ...).

rule('atm', 'less_than_9m_diam', 'tanks may be grouped to total 8000ton water', 'each group at a minimum of 15m separation').

rule('atm', 'greater_than_9m_diam', ...).
rule('atm', 'fixed_roof', 'greater_than_9m_diam', 'separation 1/2diam larger tank', 'or diam smaller tank/15m whichever is least').

rule('atm', 'floating_roof', 'greater_than_9m_diam', 'separation 1/2diam larger tank', 'or diam smaller tank/6m whichever is least').

rule('atm', 'fixed_water_or_foam_spray', 'water rate 0.2gal/min/sq ft', 'foam rate 0.1gal/min/sq ft', 'mobile units available').

/* ----------------------------------------------- PRESSURE STORAGE RULES -----------------------------------------------*/
rule('pressure', 'ethylene', 'at least 60m from tanks to boundary and ignition sources').
rule('pressure', 'c_three', 'at least 45m from tanks to boundary and ignition sources').
rule('pressure', 'c_four', 'at least 30m from tanks to boundary and ignition sources').
rule('pressure', 'flamm', '15m from tanks to buildings containing flammable materials').
rule('pressure', 'road', '15m from tanks to road/rail tank filling points').
rule('pressure', 'overhead', '15m to overhead powerlines and pipe bridges').
rule('pressure', 'pipeline', '7.5m to above ground power cables and pipelines').

rule('pressure', 'diam', '1/4 sum of diams of adjacent tanks to other pressure storage tanks').

rule('pressure', 'low_press', '15m from low pressure refrigerated tank bunds and 30m from tank shells').

rule('pressure', 'spray', 'evenly applied water spray').

rule('pressure', 'insulation', 'fire proof thermal insulation-insulation to finish 200mm above base').

rule('pressure', 'pipe_water', 'water application rate of 0.06-0.2gal/min/sq ft').

rule('pressure', 'water', 'water application rate of 0.2gal/min/sq ft').

rule('pressure', 'fire_proof', 'sufficient to provide at least 2 hours protection').

rule('pressure', 'material', 'material_vermiculite cement').

#_________________________________________________________#
/# REFRIGERATED STORAGE RULES #
#_________________________________________________________#

rule('refrig', 'insulation', 'fire proof thermal insulation').

rule('refrig', 'material', 'material_cork or polyurethane or perlite powder').

rule('refrig', 'protection', 'supplemented by water and mobile water sprays').

rule('refrig', 'fire_proof', 'sufficient to provide a minimum of 2 hours protection').

rule('refrig', 'seal', 'vapour seal-outside fire-proofing or between fire-proofing and conventional insulation').

rule('refrig', 'pipewater', 'water application rate of 0.06-0.2gal/min/sq ft').

rule('refrig', 'water', 'water application rate of 0.2gal/min/sq ft').

rule('refrig', 'vessel_supports', 'vessel supports capable of withstanding a minimum of 4 hrs exposure to external fire').

rule('refrig', 'supports', 'supports made of concrete or steel clad with fire-proof insulation').

rule('refrig', 'ethylene', 'at least 90m from tanks to boundary and ignition sources').

rule('refrig', 'c_three', 'at least 45m from tanks to boundary and ignition sources').

rule('refrig', 'c_four', 'at least 15m from tanks to boundary and ignition sources').
rule('refrig', 'flammable', '15m from tanks to buildings containing flammable materials').
rule('refrig', 'road', '15m from tanks to road or rail tanker filling point').
rule('refrig', 'overhead', '15m from tanks to overhead powerlines or pipebridges').
rule('refrig', 'low_pressure', '1/2 the sum of diameters of adjacent tanks from tanks to low pressure refrigerated tank shells').
rule('refrig', 'flammable_storage', '30m from tanks to low pressure refrigerated LFG storage and flammable liquid tank shells').
rule('refrig', 'lfg', 'LFG and flammable liquids must be in separate bunds').
rule('refrig', 'pressure', '1/4 of sum of adjacent tanks from tanks to pressure storage tanks').
rule('refrig', 'hydrants', 'hydrants in at least two locations near to the storage area', '15m from the risk').
rule('refrig', 'hydrants_b', 'hydrants should conform to BS750', '-not less than 45 Ggal/min, at 6.9bar').
rule('refrig', 'access', 'roads should have a minimum width of 3.75m', 'and a minimum clearance of 3.75m').

rule('hydrogen', 'Pressure relief to atmosphere at a safe height').
rule('hydro-gaseous', 'large', 'for sites greater than 15000scf-at least 25 feet from ignition sources').
rule('hydro-gaseous', 'flammable', 'at least 50 feet from flammable storage sites containing at least 1000 US gallons').
rule('hydro-liquid', 'large', 'for sites greater than 15000scf-at least 50 feet from ignition sources').
rule('hydro-liquid', 'flammable', 'at least 100 feet from flammable storage site').
rule('ammonia', 'hazard', '15m minimum separation and cooling water spray').
rule('ammonia', 'nohazard', 'no fire hazard').
rule('lfg', 'm_chloride', 'at least 23m from tanks to boundary and ignition sources').
rule('lfg', 'e_chloride', 'at least 15m from tanks to boundary and ignition sources').
rule('lfg', 'v_chloride', 'at least 15m from tanks to boundary and ignition sources').
s').

rule('lfg', 'm_v_chloride', 'at least 15m from tanks to boundary and ignition sources').

rule('lfg', 'flammable', 'at least 15m from tanks to buildings containing flammable materials').

rule('lfg', 'road', 'at least 15m from tanks to road/rail car filling points').

rule('lfg', 'overhead', 'at least 15m from tanks to overhead power lines and pipe bridges').

rule('lfg', 'power', 'at least 7.5m to above ground power cables and important pipelines').

rule('lfg', 'pressure', 'one quarter the sum of the diameters of adjacent tanks between tanks and other pressure storage tanks').

rule('lfg', 'low_pressure', 'at least 15m from tanks to bund wall of other low pressure tanks').

rule('lfg', 'm_amines', 'at least 15m from tanks to boundary and ignition sources').
/* standard print clause */
prt([]) :- nl.
prt([H|T]) :-
  write(H), tab(1), prt(T).

/*---------------------------------------------------------------
START CONSULTATION
/*---------------------------------------------------------------

startup :-
  nl,
  write('KNOWLEDGE BASED SYSTEM FOR THE FIRE PROTECTION'), nl,
  write(' OF A STORAGE TANK'), nl, nl,
  write(' PRE-CONSULTATION OPTIONS '), nl,
  write(' ------------------------ '), nl,
  write('start a consultation (s)'), nl,
  write('list help (h)'), nl,
  write('exit from the program (e)'), nl, nl,
  write('Option...?'), nl, nl,
  read(C),
  do_command(C),
  nl.

do_command(C) :- do(C), !.
do(s) :- start_consultation.
do(h) :- go_help.
do(e) :- abort.
do(_) :- write('Not a valid option!'), nl.

start_consultation :-
  read_knowledge_base,
  material_id,
  ( retractall(material(_)),
    retractall(conds(_)),
    retractall(diam(_)),
    retractall(roof(_)),
    retractall(nonhydrocarbon(_)),
    retractall(area(_))

read_knowledge_base :-
    nl, nl,
    consult(rules), nl.

/*--------------------------------------------------------------------*/
PROCEDURE FOR HYDROCARBON IDENTIFICATION
/*--------------------------------------------------------------------*/
material_i_d :=
    nl,
    write('Is the material stored a hydrocarbon?'), nl,
    write('Answer yes(y) or no(n)...'), nl, nl,
    read(M),
    material_check(M).

material_check(yes) :-
    assert(material(hydrocarbon)),
    storage_conditions,
    !, fail.

material_check(y) :-
    material_check(yes),
    !, fail.

material_check(no) :-
    assert(material(nonhydrocarbon)),
    material_conditions,
    !, fail.

material_check(n) :-
    material_check(no),
    !, fail.

material_check(_) :-
    mistake,
    !,
PROCEDURE FOR CHECKING THE CONDITIONS OF STORAGE

storage_conds :-
    nl,
    write('What are the conditions of storage ?'), nl, nl,
    write('Type atm. pressure, or refrigerated...'), nl, nl,
    read(C),
    type_check(C).

    type_check(atm) :-
        assert(conds(atm)),
        tank_diameter,
        !,
        fail.

    type_check(pressure) :-
        assert(conds(pressure)),
        which_hydrocarbon,
        !,
        fail.

    type_check(refrigerated) :-
        assert(conds(refrig)),
        refrigeration,
        !,
        fail.

    type_check(_) :-
        mistake,
        !,
        fail.

PROCEDURE FOR ATMOSPHERIC STORAGE

tank_diameter :-
    nl, nl, /* diameter of tank ? */
write('Is the tank diameter 9m or less ?'), nl, nl,
write('Answer yes or no'), nl,
read(D),
diam_check(D).

diam_check(yes) :-
    assert(diam(less_than_9m_diam)),
    storage_roof,
    !, fail.

diam_check(no) :-
    assert(diam(greater_than_9m_diam)),
    storage_roof,
    !, fail.

storage_roof :- /* fixed or floating roof ? */
    nl, nl,
    write('Is the tank roof fixed or floating ?'), nl, nl,
    write('Type fixed or floating... '), nl,
    read(R),
    roof_check(R),
    !, fail.

roof_check(fixed) :-
    assert(roof(fixed_roof)),
    rule_match,
    !, fail.

roof_check(floating) :-
    assert(roof(floating_roof)),
    rule_match,
    !, fail.

rule_match :- /* search rule base */
    nds(C),
    diam(D),
    roof(R),
    rule(C, size, Fpa),
    rule(C, layout, Fpb),
    rule(C, boundary, Fpc),
rule(C, bunds, Fpn, Fpo, Fpq),
prt([Fpa]), nl,
prt([Fpb]), nl,
prt([Fpc]), nl,
prt([Fpn]), nl,
prt([Fpo]), nl,
prt([Fpq]), nl,
print_main_rules,
fail.

print_main_rules :-
diam(D),
coords(C),
roof(R),
rule(C, D, Fpd, Fps),
rule(C, R, Fpf),
rule(C, E, D, Fps, Fpz),
rule(C, Fph, Fpi, Fpj, Fpk),
prt([Fpd, Fps]), nl, nl,
prt([Fpf]), nl, nl,
prt([Fph]), nl,
prt([Fpi]), nl,
prt([Fpj]), nl,
prt([Fpk]), nl,
go_remove,
fail.
go_remove :-
retract(roof(_)),
retract(diam(_)),
go_again.

/*-----------------------------------------------------------------------
PROCEDURE FOR PRESSURE STORAGE
/*-----------------------------------------------------------------------

which_hydrocarbon :-
write('Is the material ethylene, C_3 or C_4 ?'), nl, nl,
write('Answer by typing ethylene, c_three, c_four....'), nl,
read(E),
material(E), !, fail.

material(ethylene) :-
  conds(C),
  rule(C, ethylene, Fpa), nl,
  write('If pressurized ethylene storage, then :- '), nl,
  write('-------------------------------------- '), nl,
  prt([Fpa]), nl,
  next, !, fail.

material(c_three) :-
  conds(C),
  rule(C, c_three, Fpa), nl,
  write('If pressurized C_3 storage, then :- '), nl,
  write('---------------------------------- '), nl,
  next, !, fail.

material(c_four) :-
  rule(pressure, c_four, Fpa), nl,
  write('If pressurized C_4 storage, then :- '), nl,
  write('----------------------------------- '), nl,
  prt([Fpa]), nl,
  next, !, fail.

material(_) :-
  nl, mistake, !, fail.

next :-
  all_rules, !, fail.

all_rules :-
  conds(C),
  write('Other separation requirements are :- '), nl,
read(S),
if_s(S),
l, fail.

if_s(yes) :-
    nl,
    write('Is the pipework carried beyond the vertical projection'), nl,
    write('of the vessel, with no pipe fittings beneath the vessel ?'), nl,
    write('Answer yes(y) or no(n)...'), nl,
    read(V),
    if_v(V),
l, fail.

if_s(y) :-
    if_s(yes),
l, fail.

if_s(no) :-
    nl,
    conds(C),
    rule(C, water, Fpj),
    prt([Fpj]), nl,
    go_again,
l, fail.

if_v(y) :-
    conds(C),
    rule(C, pipe_water, Fpj),
    prt([Fpj]),
    go_again,
l, fail.
if_v(yes), !, fail.
if_v(no) :- if_s(no), !, fail.
if_v(n) :- if_v(no), !, fail.
if_v(_) :- mistake, !, fail.
option(b) :- nl, 
conds(C), 
rule(C, insulation, Fpa), 
rule(C, protection, Fpb), 
rule(C, material, Fpc), 
rule(C, fire_proof, Fpd), 
rule(C, seal, Fpe), 
rule(C, vessel_supports, Fpf), 
rule(C, supports, Fpg), 
prt([Fpk]), nl, 
prt([Fpl]), nl, 
prt([Fp1]), nl, 
go_again, !, fail.

/*------------------------------------------------------------------------
PROCEDURE FOR REFRIGERATED STORAGE
 /*------------------------------------------------------------------------*/

refrigeration :- nl, 
conds(C), 
rule(C, insulation, Fpa), 
rule(C, protection, Fpb), 
rule(C, material, Fpc), 
rule(C, fire_proof, Fpd), 
rule(C, seal, Fpe), 
rule(C, vessel_supports, Fpf), 
rule(C, supports, Fpg),
rule(C, hydrants, Fpq, Fpr),
rule(C, hydrants b, Fps, Fpt),
write('For refrigerated storage, then :-'), nl,
write('----------------------------------'), nl, nl,
prt([Fpa]), nl,
prt([Fpb]), nl,
prt([Fpc]), nl,
prt([Fpd]), nl,
prt([Fpe]), nl,
prt([Fpf]), nl,
prt([Fpg]), nl,
prt([Fph]), nl,
prt([Fpj]), nl,
prt([Fpl]), nl,
prt([Fpm]), nl,
prt([Fpn]), nl
), !,
separation_requirements,
!, fail.

separation_requirements:-
  conds(C),
  write('Is the material ethylene, C-3 or C-4 ?'), nl,
  write('Type ethylene, c_three or c_four...'), nl,
  read(E),
  substance(E),
  fail.

substance(ethylene) :-
  rule(C, ethylene, Fph), nl,
  write('If refrigerated ethylene storage, then :-'), nl,
  write('--------------------------------------------'), nl,
  prt([Fph]), nl,
  refrig_distances,
  !,
  fail.

substance(c_three) :-
  rule(C, c_three, Fph),
  write('If refrigerated c_three storage, then :-'), nl,
  write('------------------------------------------'), nl,
  prt([Fph]), nl,
  refrig_distances,
I, fail.

```
substance(c_four) :-
  rule(C, c_four, Fph),
  write('If refrigerated c_four storage, then :-'), nl,
  prt([Fph]), nl,
  refriC_distances,
  !,
  fail.

substance(_) :-
  mistake,
  !,
  fail.

refriC_distances :-
  conds(C),
  write('Other separation requirements are :-'), nl,
  write('--------------------------------------'), nl,
  rule(C, flammable, Fpi),
  rule(C, road, Fpj),
  rule(C, overhead, Fpk),
  rule(C, low_pressure, Fpl),
  rule(C, flammable_storage, Fpm),
  rule(C, lfg, Fpn),
  rule(C, pressure, Fpo),
  (   prt([Fpi]), nl,
      prt([Fpj]), nl,
      prt([Fpk]), nl,
      prt([Fpl]), nl,
      prt([Fpm]), nl,
      prt([Fpn]), nl,
      prt([Fpo]), nl
   ),!
  water_spray_refrig,
  !,
  fail.

water_spray_refrig :-
  write('Is the ground beneath the vessel sloped away'), nl,
  write('to a catchment area ?'), nl,
  write('Answer yes(y) or no(n)...'), nl, nl,
  read(S),
  if_answer(S),
  !,
  fail.
```
if_answer(yes) :-
    nl,
    write('Is the pipework carried beyond the vertical projection'), nl,
    write('of the vessel, with no pipe fittings beneath the vessel ?'), nl,
    write('Answer yes(y), or (no)...'), nl,
    read(V),
    if_reply(V),
    fail.

if_answer(y) :-
    if_answer(yes),
    !,
    fail.

if_answer(no) :-
    nl,
    conds(C),
    rule(C, water, Fpp),
    prt([Fpp]), nl,
    go_again,
    !,
    fail.

if_answer(n) :-
    if_answer(no),
    !,
    fail.

if_answer(_) :-
    mistake,
    !,
    fail.

if_reply(yes) :-
    conds(C),
    nl,
    rule(C, pipewater, Fpp),
    prt([Fpp]),
    go_again,
    !,
    fail.

if_reply(y) :-
    if_reply(yes),
I, fail.
if_reply(no) :-
  if_answer(no),
  !,
  fail.
if_reply(_)
  !,
  fail.

/*---------------------------*/
NONHYDROCARBON PROEDURE
/*---------------------------*/

material_conds :-
  nl,
  write('Is the material stored as LFG ?'), nl,
  write('Answer yes(y) or no(n).'), nl,
  read(A),
  check_LFG(A),
  !,
  fail.

check_LFG(no) :-
  write('Ammonia, chlorine and hydrogen are covered in this program.'), nl,
  write('Type in one of the above for information about its storage'), nl,
  read(Chemical),
  chemical(Chemical), nl,
  !,
  fail.

check_LFG(n) :-
  check_LFG(no),
  !,
  fail.

chemical(hydrogen) :-
  assert(nonhydrocarbon(hydrogen)),
  nonhydrocarbon(H),
  write('Is hydrogen storage gaseous or liquid ?'), nl,
write('Type gaseous or liquid...'), nl,
read(State), nl,
hydro_protection(State), nl,
fail.

hydro_protection(gaseous) :-
assert(conds(hydro_gaseous)),
conds(G), nl,
rule(hydrogen,Fpa),
rule(G.large,Fpb),
rule(G.flammable,Fpo),
write('For gaseous hydrogen, then :-'), nl,
write('-----------------------------'), nl,
prt([Fpa]), nl,
prt([Fpb]), nl,
prt([Fpo]), nl,
go_again,
fail.

hydro_protection(liquid) :-
assert(conds(hydro_liquid)),
nl,
conds(G), nl,
rule(hydrogen,Fpa),
rule(G.large,Fpb),
rule(G.flammable,Fpo),
write('For liquid hydrogen storage, then :-'), nl,
write('-----------------------------'), nl,
nl,nl,
prt([Fpa]), nl,
prt([Fpb]), nl,
prt([Fpo]), nl,
go_again,
fail.

chemical(ammonia) :-
assert(conds(ammonia)),
nl,
write('Is there a tank containing flammable liquid'), nl,
write('near enough to cause a knock-on effect ?'), nl,
write('Answer yes(y) or no(n)...'), nl, nl,
read(R),
act_on(R), !,
fail.

act_on(yes) :- nl, nl,
      assert(area(hazard)),
      area(H),
      conds(C),
      rule(C, hazard, Fpa),
write('For ammonia storage in a hazardous area, then :-'), nl,
write('-------------------------'), nl,
prt([Fpa]),
go_again, !,
fail.

act_on(y) :-
      act_on(yes), !, fail.

act_on(no) :- nl, nl,
      assert(area(nohazard)),
      area(H),
      conds(C),
      rule(C, nohazard, Fpa),
write('For ammonia storage, in a safe area, then :-'), nl,
write('-------------------------'), nl,
prt([Fpa]), nl, !,
go_again, !, fail.

act_on(n) :-
      act_on(no), !, fail.

chemical(chlorine) :-
chemical(ammonia), !, fail.
check_LFG(yes) :-
    nl,
    write('Is the nonhydrocarbon soluble in water?'),nl,
    write('Answer yes(y) or no(n)'),nl,
    read(Answer),
    soluble(Answer), !,
    fail.

check_LFG(y) :-
    check_LFG(yes), !,
    fail.

soluble(yes) :-
    nl,
    write('If the material is one of the following'),nl,
    write('then type in the name of the material...'),nl,
    write('if not, then type no(n)'),nl,nl,
    write('methyl_chloride'),nl,
    write('viny_chloride'),nl,
    write('methyl_viny_ether'),nl,
    write('ethyl_chloride...'),nl,nl,
    read(Lfg),
    get_rule(Lfg), !, fail.

soluble(y) :-
    soluble(yes), !, fail.

soluble(no) :-
    write('Does the substance come under the category of methylamines?'),nl,
    write('Answer yes(y) or no(n)'),nl,
    read(Reply),
    go_rule(Reply), !, fail.

soluble(n) :-
    soluble(no), !, fail.

soluble(_):-
get_rule(methyl_chloride) :-
  nl,
  assert(conds(lfg)),
  conds(C),
  rule(C, m_chloride, Fpa), nl,
  write('For the storage of methyl chloride, then :-'), nl,
  write('--------------------------------------------'), nl, nl,
  prt([Fpa]), nl,
  get_other_rules,
  !,
  fail.

get_rule(vinyl_chloride) :-
  nl,
  assert(conds(lfg)),
  conds(C),
  rule(C, v_chloride, Fpa), nl,
  write('For the storage of vinyl chloride, then :-'), nl,
  write('--------------------------------------------'), nl, nl,
  prt([Fpa]), nl,
  get_other_rules,
  !,
  fail.

get_rule(methyl_vinyl_ether) :-
  nl,
  assert(conds(lfg)),
  conds(C),
  rule(C, m_v_ether, Fpa), nl,
  write('For the storage of methyl vinyl ether, then :-'), nl,
  write('--------------------------------------------'), nl, nl,
  prt([Fpa]), nl,
  get_other_rules,
  !,
  fail.

get_rule(ethyl_chloride) :-
  nl,
  assert(conds(lfg)),
  conds(C),
  rule(C, e_chloride, Fpa), nl,
  write('For the storage of ethyl chloride, then :-'), nl,
  write('--------------------------------------------'), nl, nl,
prt([Fpa]), nl,
get_other_rules,
!.
fail.

get_rule(no) :-
    nl,
    assert(conds(lfg)),
    write('For nonhydrocarbons soluble in water, then :-'), nl,
    write('----------------------------------------------'), nl, nl,
    get_other_rules,
    !.
fail.

get_rule(n) :-
    get_rule(no),
    !.
fail.

get_rule(_) :-
    mistake,
    !.
fail.

get_other_rules :-
    nl,
    conds(C),
    rule(C, flammable, Fpb),
    rule(C, road, Fpc),
    rule(C, overhead, Fpd),
    rule(C, power, Fpe),
    rule(C, low_pressure, Fpg),
    (    prt([Fpb]), nl,
    prt([Fpc]), nl,
    prt([Fpd]), nl,
    prt([Fpe]), nl,
    prt([Fpg]), nl)
    !.
fire_protection,
fail.

go_rule(no) :-
    get_other_rules,
    !.
fail.
go_rule(n) :-
    go_rule(no),
    !,
    fail.

go_rule(yes) :-
    nl,
    assert(conds(lfg)),
    conds(C),
    rule(C,m_amines,Fpa),
    write('For nonhydrocarbons soluble in water, then :-'), nl,
    write('-----------------------------------------------'), nl,
    prt([Fpa]), nl,
    get_other_rules, !,
    fail.

go_rule(y) :-
    go_rule(yes), !,
    fail.

go_rule(_) :-
    mistake, !,
    fail.

fire_protection :-
    nl, nl,
    assert(conds(pressure)),
    conds(C),
    rule(C,spray,Fph),
    rule(C,insulation,Fpi),
    write('The two fire protection options are :-'), nl,
    write('-----------------------------------------------'), nl, nl,
    prt([Fph]), nl,
    prt([Fpi]), nl,
    option_choice, !,
    fail.

go_again :-
    nl,
    write('Do you want another consultation ?'), nl,
write('Answer yes (y), or no (n) '), nl.
read(Answer),
check_up(Answer),
!, fail.

check_up(yes) :-
    start_up.

check_up(y) :-
    start_up.

check_up(no) :-
    abort.

check_up(n) :-
    abort.

check_up(-) :-
    mistake, !,
    fail.

mistake :- nl,
    write('Not a valid option !'),
    go_again,
    fail.

/*------------------------------------------------------------------------*/
HELP PROCEDURE
/*------------------------------------------------------------------------*/
go_nelp :- nl,
    write('This program will help you decide the fire protection'), nl,
    write('requirements needed for storage tanks.'), nl,
    write('Those tanks covered are atmospheric storage, pressure'), nl,
    write('storage, pressure refrigerated storage, and fully refrigerated storage for both hydrocarbons and nonhydrocarbons.'), nl,
    write('Answer each question followed by a full-stop, followed by a <carriage-return>.'), nl,
    start_up.

/* Automatic entry into program */
?- seeing(F), see(user), start_up, see(F).
APPENDIX D

EX-TRAN 7 LISTINGS FOR EIV INSTALLATION
This expert system will help you decide whether or not a particular piece of process plant needs an emergency isolation valve. It considers things such as temperature, leak history and alternatives.

That was the end of the expert system on emergency isolation valves.

Emergencw Isolation Valves

Are there other reasons for installing an emergency isolation valve?

Is the material above its autoignition temperature?

Is there an alternative way of stopping a leak?

Is the fluid gas in a compressor?

Is there a history of leaks on the item of plant?

Is the material being pumped at extremes of temperature (>30 or <180 deg C)?

Is the plant item not likely to leak, but if it does, a large amount will escape with no way of stopping it?

Is the plant item otherwise likely to leak?

Is it convenient to the process to install an EIV?

The decision is install an EIV

The decision is do not install an EIV
subroutine eiv
character*8 av, ansl, atfile
integer nrar, ansi, atn
common/userin/nrar, ansi, atn
common/userre/ans r
common/userch/anslrstfile
external Zeiv, rextsub

1 call inform (0, 5, Zeiv, rextsub, 1, 1, 2)
IF(ansl.EQ.av(atn, 1, 0))THEN
call inv(0, 5, 1.00, 5, 1.00, 2, 1, Zeiv, rextsub, 1, 1, 2)
ELSE IF(ansl.EQ.av(atn, 2, 0))THEN
call inv(0, 5, 2.00, 1, 1.00, 4, 99, Zeiv, rextsub, 1, 1, 2)
ELSE IF(ansl.EQ.av(atn, 1, 0))THEN
end if
end if
end if
end if
END

2 subroutine Zeiv
end if
end if
end if
end if
call rulfil (6, 1, 0, 1, 0, 1, 0, 5, 1.00, 1, 0)
call rulfil (0, 0, 2, 0, 5, 1.00, 1, 0)
call rulfil (0, 1, 0, 2, 1.00, 1, 0)
call rulfil (0, 1, 2, 0, 2, 1.00, 1, 0)
call rulfil (0, 2, 1, 0, 1, 1.00, 1, 0)
call rulfil (0, 3, 0, 0, 1, 1.00, 1, 0)
call rulfil (0, 3, 2, 0, 6, 1.00, 1, 0)
call rulfil (2, 2, 2, 0, 1, 1.00, 1, 0)
call rulfil (2, 2, 2, 0, 2, 1.00, 1, 0)
call rulfil (2, 2, 2, 0, 1, 1.00, 1, 0)
call rulfil (2, 2, 2, 0, 2, 1.00, 1, 0)
PROGRAM XFERT
CALL xeiv
STOP
END
SUBROUTINE USR1
END
SUBROUTINE USR2
END
SUBROUTINE USR3
END
SUBROUTINE USR4
END
subroutine :: xeiv
external srxeiv
character*51 fname, mssass, auxfil
fname='eiv. prf'
mssass='mes. dat'
auxfil='xeiv. AUX'
call driver(srxeiv, fname, mssass, auxfil)
end
subroutine srxeiv
external eiv
"call finder(mode, iprob)
if(iprob.eq. 1)call sorter(mode, i, eiv )
end
APPENDIX E

MICRO-EXPERT LISTING FOR EIV INSTALLATION
The following is a listing of the advice language
compilation from Micro-Expert for the emergency isolation
valve problem.

1
2
3
4 GOAL Fiteiv 'An eiv is needed'
5
6 BAYESIAN HAZARD LS 100 LN 0.01
7 OTHER LS 20 LN 0.01
8 HISTORY LS 10 LN 0.1
9 TEMP LS 10 LN 0.1
10 ALTER LS 0.01 LN 50
11 PRIOR 0.5
12
13 RULE OTHER 'The item is otherwise'
   ' likely to leak'
14 AND ALTER TEMP
15
16 QUESTION HISTORY 'The item has a history'
17   ' of leaks'
18 BLOCKED HAZARD 0.0 5.0
19
20 CERT
21
22
23 QUESTION ALTER 'There is an alternative'
24   ' way to stop a leak'
25
26 YESNO
27
28 QUESTION HAZARD 'The hazard is large'
29
30
31
32
LIST OF GOALS

1 Fiteiv...............4

LIST OF HYPOTHESES

1 Fiteiv...............4
2 HAZARD...............29
3 OTHER...............13
4 HISTORY...............17
5 TEMP...............35
6 ALTER...............24

This model does not use externals
## Cross-Referencing Listing

<table>
<thead>
<tr>
<th>Line</th>
<th>No</th>
<th>Name</th>
<th>Used By</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>6</td>
<td>ALTER</td>
<td>Fiteiv OTHER</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>Fiteiv</td>
<td>TOP LEVEL GOAL</td>
</tr>
<tr>
<td>29</td>
<td>2</td>
<td>HAZARD</td>
<td>HISTORY Fiteiv</td>
</tr>
<tr>
<td>17</td>
<td>3</td>
<td>HISTORY</td>
<td>Fiteiv</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>OTHER</td>
<td>Fiteiv</td>
</tr>
<tr>
<td>35</td>
<td>5</td>
<td>TEMP</td>
<td>Fiteiv OTHER</td>
</tr>
</tbody>
</table>

NO ERRORS

Advice Language Compilation NO ERRORS
Explanation

First the a priori probability is turned into odds. Then for each antecedant hypothesis the odds are multiplied by a factor. If the ceratainty of the antecedant hypothesis is -5 then this factor is equal to the LN value. If the certainty of the antecedant hypothesis is 5 then the factor is equal to the LS value.

The following section discusses some of the clauses in detail:

RULE OTHER 'The item is otherwise' 'likely to leak'
AND ALTER TEMP

Meaning: The current probability of OTHER is the lowest of the current probabilities of ALTER and TEMP.

GOAL Fiteiv 'An eiv is needed'

BAYESIAN HAZARD LS 100 LN 0.01
OTHER LS 20 LN 0.01
HISTORY LS 10 LN 0.1
TEMP LS 10 LN 0.1
ALTER LS 0.001 LN 50

PRIOR 0.5

Meaning: Let us look at the prior probability first. The value given is 0.5 which means that the probability of fitting an emergency isolation valve before any of the antecedant hypotheses have been resolved is estimated to be 0.5. For the purposes of the Bayesian calculation the a
priori probability is turned into an a priori odds of 0.5/0.5 = 1.00. If the antecedent hypotheses HAZARD has a certainty factor of 5 then these odds are multiplied by the LS factor for HAZARD.

i.e. 1.00*100 = 100

If HAZARD has a certainty factor of -5.00 then these a priori odds are multiplied by the LN factor for HAZARD.

i.e. 1.00*0.01 = 0.01

If HAZARD has a certainty factor of 0 then the a priori odds are left alone.

For intermediate values of HAZARD's certainty factor an appropriate multiplier is calculated by linear interpolation. This multiplication is repeated for each antecedent in turn as follows :-

Let us assume that

HAZARD has a certainty factor of 5
OTHER has a certainty factor of 2.5
HISTORY has a certainty factor of -1
TEMP has a certainty factor of 5
ALTER has a certainty factor of 0

The multiplication factors are therefore

HAZARD \[ 1 + (100 - 1) \times \frac{5}{5} = 100 \]
OTHER \[ 1 + (20 - 1) \times \frac{2.5}{5} = 10 \]
HISTORY  $1 + ((1-0.1)*(-1/-5)) = 0.82$

TEMP  $1 + (10 - 1)*5/5 = 10$

ALTER  $1 + (0.01 - 1)*0 = 0$

Therefore the current odds of Fiteiv are :-

$0.5*100*10*0.82 = 410$

Therefore current probability = $410/411 = 0.997$

QUESTION HISTORY 'The item has a history'

'of leaks'.

BLOCKED HAZARD 0.0 5.0

CERT

Here the reserved word 'BLOCKED' is used. It is used when some control over the order in which questions are asked is needed, and to be able to skip irrelevant questions.

In this case, we have assumed that 'large hazard' is more relevant, and that there is no point in asking about 'history' unless there is a possibility of a large hazard occurring.
APPENDIX F

PROLOG PROGRAM LISTING FOR FLARE SYSTEM DESIGN
/* Flare program, based on Winston's BAGGER */

/*
Rules, based on BAGGER for flare system design

F1 If the step is check flares
    there are ground flares present
    Then add elevated flares to system

F2 If the step is check flares
    then discontinue check flares
    start flare type step

F3 If the step is flare type
    there is an item to flare
    the material is a hydrocarbon
    the flow is low
    then use a ground flare

F4 If the step is flare type
    there is an item for flare type
    the material is a hydrocarbon
    the flow is low
    the gas is in excess
    Then use a ground flare

F5 If the step is flare type
    there is an item to flare
    the material is a hydrocarbon
    the flow is high
    the gas is not recoverable
    Then use an elevated flare

F6 If the step is flare type
    there is an item for flare type
    the material is a hydrocarbon
    the flow is low
    the flow is not continuous
    Then use an elevated flare

F7 If the step is flare type
    then discontinue the flare type step
    start the flare layout step

F8 If the step is flare layout
    there is a ground flare
    there are process units
    Then +300 feet separation

F9 If the step is flare layout
    there is an elevated flare stack
    there is an oil-water separator
    Then +200 feet separation

F10 If the step is flare layout
    there is an elevated flare stack
    there is a floating roof tank
Then +200 feet separation

F11 If the step is flare layout
    there is an elevated flare stack
    there are other process units
    Then +200 feet separation

F12 If the step is flare layout
    then discontinue flare layout step
    start the flare tip choice step

F13 If the step is flare tip choice
    there is an item for flare tip choice
    the flow is high
    a smokeless flame is required
    Then use a steam ring flare tip

F14 If the step is flare tip choice
    there is an item for flare tip choice
    there is an unsaturated hydrocarbon
    the flow is high
    a smokeless flame is required
    Then use a steam ring flare tip

F15 If the step is flare tip choice
    there is an item for flare tip choice
    the material is hydrogen sulphide
    a smokeless flame is not required
    Then use a utility flare tip

F16 If the step is flare tip choice
    there is an item for flare tip choice
    the material is methane
    a smokeless flame is not required
    Then use a utility flare tip

F17 If the step is flare tip choice
    there is an item for flare tip choice
    the material is carbon monoxide
    a smokeless flame is not required
    Then use a utility flare tip

F18 If the step is flare tip choice
    there is an item for flare tip choice
    there is a saturated hydrocarbon
    the flow is low
    a smokeless flame is required
    Then use a centre steam flare tip

F19 If the step is flare tip choice
    there is an item for flare tip choice
    there is an unsaturated hydrocarbon
    the flow is low
    a smokeless flame is required
    Then use a centre steam flare tip

F20 If the step is flare tip choice
    there is an item for flare tip choice
    there is a saturated hydrocarbon
    the flow is high
a smokeless flame is not required
Then use a centre steam flare tip

F21 If the step is flare tip choice
there is an item for flare tip choice
there is an unsaturated hydrocarbon
the flow is high
a smokeless flame is required
then use a centre steam flare tip

F22 If the step is flare tip choice
then discontinue flare tip choice step
start alternatives step

F23 If the step is alternatives
there is an item to flare
the flow is low
the gas is toxic
Then do not flare

F24 If the step is alternatives
there is an item to flare
the material is hydrogen
the flow is high
Then vent to atmosphere

F25 If the step is alternatives
there is an item to flare
the material is hydrogen
discharge frequency is low
Then vent to atmosphere

F26 If the step is alternatives
there is an item to flare
the material is hydrogen
Then use a separate vent

F27 If the step is alternatives
there is an item to flare
the material is methane
Then use a separate vent

F28 If the step is alternatives
then discontinue alternatives step
start safety step

F29 If the step is safety
there is an item for safety
the material is gas + hydrogen
the flow is high
the stack is large
Then fit oxygen alarm for 5%

F30 If the step is safety
there is an item for safety
the material is any gas
the flow is high
the stack is large
Then fit oxygen alarm for 2%
F31 If the step is safety
the system is under maintenance
Then purge with nitrogen before use

F32 If the step is safety
there is an item for safety
the gas can decompose with air
Then fit a flame arrestor

F33 If the step is safety
then discontinue safety step
start general flare rules step

F34 If the step is general flare rules
there are general flare items
the material is hydrogen
Then vent at least 10 feet above buildings

F35 If the step is general flare rules
there are general flare items
the material is ammonia
Then complete burning required

F36 If the step is general flare rules
there are general flare items
the material contains phosgene
Then complete burning required

F37 If the step is general flare rules
there are general flare items
the material contains hydrogen sulphide
Then complete burning required
/* Knowledge - type of flare system */
flare_item(hydrocarbon, lo_flow, flare_type, continuous).
flare_item(hydrocarbon, lo_flow, flare_type, excess).
flare_item(hydrocarbon, hi_flow, flare_type, non_recoverable).
flare_item(gas_mixture, lo_flow, flare_type, contains_toxics).
flare_item(hydrocarbons, lo_flow, flare_type, non_continuous).
flare_item(gas, hi_flow, alternative_to_flaring, corrosive).
flare_item(gas, hi_flow, alternative_to_flaring, reacts_explosively).
flare_item(gas, hi_flow, alternative_to_flaring, decomposes).

/* Knowledge - type of flare tip */
flare_item(saturated_hydrocarbons, hi_flow, flare_tip, smokeless_flame).
flare_item(methane, any_flow, flare_tip, smokey_flame).
flare_item(hydrogen_sulphide, any_flow, flare_tip, smokey_flame).
flare_item(carbon_monoxide, any_flow, flare_tip, smokey_flame).
flare_item(saturated_hydrocarbons, lo_flow, flare_tip, smokeless_flame).
flare_item(unsaturated_hydrocarbons, lo_flow, flare_tip, smokeless_flame).
flare_item(saturated_hydrocarbons, hi_flow, flare_tip, smokey_flame).
flare_item(unsaturated_hydrocarbons, hi_flow, flare_tip, smokey_flame).

/* Knowledge - layout of flare */
flare_item(stack, any_flow, plant_layout, oil_water_separator).
flare_item(stack, any_flow, plant_layout, floating_roof_tank).
flare_item(stack, hi_flow, plant_layout, space_short).
flare_item(ground_flare, any_flow, plant_layout, process_units).

/* Knowledge for alternatives */
flare_item(gas, lo_flow, alternative, toxic).
flare_item(_, hi_flow, alternative, hydrogen).
flare_item(_, discharge_freq_lo, alternative, hydrogen).
flare_item(_, general_discharge, alternative, hydrogen).
fluence_item(_, general_discharge, alternative, methane).

/* Knowledge - safety in flare systems */
flare_item(gas+hydrogen, hi_flow, safety, large_stack).
flare_item(gas, hi_flow, safety, large_stack).
flare_item(stack, any_flow, safety, maintenance).
flare_item(gas, any_flow, safety, decompose_in_air).

/* Knowledge - general flaring */
flare_item(hydrogen, any_flow, general, venting).
flare_item(ammonia, any_flow, general, burning).
flare_item(gas+phosgene, any_flow, general, burning).
flare_item(gas+hydrogen_sulphide, any_flow, general, burning).

/*
start :-
    nl,
    check(flares),
    step(flare_type),
    step(plant_layout),
    step(flare_tip),
    step(alternatives),
    step(safety),
    step(general),
    !, fail.
check(flares) :-
    item(ground_flares),
    item(stack).
item(X) :-
    flare_item(X, _, _, _), nl,
    write(X), tab(2), write('present !'), nl.
item (_) :-
    nl, write(X), tab(2), write('not present !'), nl, !, fail.
step(flare_type) :-
    write('***'), tab(2), write('step write('flare_type'), tab(2),
    write('***'), nl,
    item_to_flare(flare_type),
    and(lo_flow, continuous),
    and(lo_flow, excess),
    conclusion(ground_flare),
    go(elevated_flare).
go(elevated_flare) :-
    and(hi_flow, non_recoverable),
    and(lo_flow, non_continuous),
    conclusion(elevated_flare).
go(two_hundred_feet) :-
    item_to_flare(stack),
    and(_, oil_water_separator),
    and(_, floating_roof_tank),
    conflict(stack, process_units),
    conclusion(two_hundred_feet).
go(smokey_flame) :-
    conflict(hydrogen_sulphide, _, smokey_flame),
    conflict(methane, _, smokey_flame),
    conflict(carbon_monoxide, _, smokey_flame),
    conclusion(utility_flare_tip).
go(other) :-
    conflict(saturated_hydro, lo_flow, smokeless_flame),
    conflict(unsaturated_hydro, lo_flow, smokeless_flame),
    conflict(saturated_hydro, hi_flow, smokey_flame),
    conflict(unsaturated_hydro, hi_flow, smokey_flame),
    conclusion(centre_steam_tip).
go(complete_burning) :-
    conflict(ammonia,_,burning),
    conflict(gas+phosgene,_,burning),
    conflict(gas+hydrogen_sulphide,_,burning),
    conclusion(complete_burning_required).

item_to_flare(flare_type) :-
    flare_item(A,_,flare_type,_) , nl,
    write('There is an item to flare !'), nl,
    write(A), nl, nl,
    assert(material(A)),
    assert(step_is(flare_type)).

item_to_flare(ground_flare) :-
    flare_item(ground_flare,_,A,_) , nl,
    write('There is a ground flare !'), nl,
    write(A), nl.

item_to_flare(elevated_flare_type) :-
    flare_item(A,_,flare_type,_) , nl,
    write('There is an item to flare !'), nl,
    write(A), nl, nl.

item_to_flare(stack) :-
    flare_item(stack,_,A,_) , nl,
    write('There is a stack !'), nl,
    write(A), nl, nl.

item_to_flare(flare_tip) :-
    flare_item(A,_,flare_tip,_) , nl,
    write('There is an item for flare tip !'), nl.

item_to_flare(alternatives) :-
    flare_item(A,_,alternative,_) , nl,
    write('There is an item for alternatives !'), nl,
    write(A), nl, nl.

item_to_flare(safety) :-
    flare_item(A,_,safety,_) , nl,
    write('There are safety items !'), nl,
    write(A), nl, nl.

item_to_flare(safety_one) :-
    flare_item(gas,_,safety,_) , nl,
    write(gas), nl.

item_to_flare(general) :-
    flare_item(A,_,general,_) , nl,
    write('There are general flare items !'), nl,
    write(A), nl, nl.

and(X,Y) :-
    flare_item(A,X,_,Y) , nl,
    write('Conditions are'),
    tab(2), write(X), tab(2),
    write('and'), tab(2),
    write(Y), nl,
assert(conditions(X, Y)).

conflict(X, Y) :-
    flare_item(X, _, Y), nl,
    write('Conditions are'),
    tab(2), write(Y), tab(2),
    write('and'), tab(2),
    write(X), nl.

conflict(X, Y, Z) :-
    flare_item(X, Y, _, Z), nl,
    write('Conditions are'),
    tab(2), write(X),
    tab(2), write('and'), tab(2),
    tab(2), write(Y), tab(2),
    write('and'), tab(2),
    write(Z), nl.

conclucion(X) :-
    nl,
    write('*** therefore'), tab(2),
    write(X), tab(2),
    write('***'), nl, nl, nl.

step(plant_layout) :-
    write('***'), tab(2), write('step is'), tab(2),
    write('plant_layout'), tab(2),
    write('***'), nl,
    item_to_flare(ground_flare),
    and(-, process_units),
    conclusion(three_hundred_feet_separation),
    go(two_hundred_feet).

step(flare_tip) :-
    write('***'), tab(2), write('step is'), tab(2),
    write('flare_tip'), tab(2),
    write('***'), nl,
    item_to_flare(flare_tip),
    and(hi_flow, smokeless_flame),
    conflict(unsaturated_hydroc, hi_: low, smokeless_flame),
    conclusion(steam_ring_flare_tip),
    go(smokey_flame),
    go(other).

step(alternatives) :-
    write('***'), tab(2), write('step is'), tab(2),
    write('alternatives'), tab(2),
    write('***'), nl,
    item_to_flare(alternatives),
    and(lo_flow, toxic),
    conclusion(do_not_flare),
    sub_step_one(alternatives),
    sub_step_two(alternatives).

step(safety) :-
    write('***'), write('step is'), tab(2),
    write('safety'), tab(2),
    write('***').
write('***'), nl,
item_to_flare(safety),
and(hi_flow, large_stack),
conclusion(oxygen_alarm_five_per_cent),
sub_step_one(safety),
sub_step_two(safety),
sub_step_three(safety).

step(general) :-
write('***'), write('step is'), tab(2),
write('general flaring rules'), tab(2),
write('***'), nl,
item_to_flare(general),
and(any_flow, venting),
conclusion(vent_at_least_ten_feet_above_buildings),
go(complete_burning).

sub_step_one(alternatives) :-
and(hi_flow, hydrogen),
and(discharge_freq_lo, hydrogen),
conclusion(vent_to_atm).

sub_step_two(alternatives) :-
and(general_discharge, hydrogen),
and(general_discharge, methane),
conclusion(use_separate_vent).

sub_step_one(safety) :-
item_to_flare(safety_one),
and(hi_flow, large_stack),
conclusion(oxygen_alarm_two_per_cent).

sub_step_two(safety) :-
and(any_flow, maintenance),
conclusion(purge_with_nitrogen).

sub_step_three(safety) :-
and(any_flow, decompose_in_air),
conclusion(fit_flame_arrestor).