

Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

A KNOWLEDGE-BASED SYSTEM
THAT GIVES BETTER PRICE RECOMMENDATIONS
THAN A GOOD MANAGER

Ching Biu Tse

August 1991

A dissertation submitted to the Faculty of Business Studies,
Massey University, in fulfilment of the requirements for the
degree of Doctor of Philosophy.

Approved by:



Dr A C Lewis,
Supervisor

TO WHOM IT MAY CONCERN

This is to state that the research carried out for my Ph.D. thesis entitled "A Knowledge-based System that Gives Better Price Recommendations than a Good Manager" in the Marketing Department at Massey University, New Zealand, is all my own work.

This is also to certify that the thesis material has not been used for any other degree.

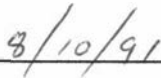


Ching Biu Tse



Dr Anthony C Lewis, Supervisor

Date:



ACKNOWLEDGEMENT

I would like to express my deep feelings of gratitude to the supervisor of my study, Dr Tony Lewis, and also to my advisor, Mr Ray Kemp, who have both been very generous in providing me with their valuable guidance during this research endeavour. The insight gained from interaction with them will continue to be a valuable asset throughout my professional life. I would also like to thank Mr Don Esslesmont, Professor Phil Gendall, Dr Mike Brennan, Mr Robert Langton, Ms Pascale Quester and Dr Peter Andrea for their helpful suggestions. I am indebted to all the managers participating in the knowledge engineering part of this study for providing me with valuable rules of how price decisions should be made and the values of the membership functions used in the system. I am also grateful to Ms M. M. Hilder of the Modern Languages Department and Ms L. Stock for their help in touching up the English in this thesis.

ABSTRACT

The objective of this research was to investigate whether it is possible to construct a computer system that provides good price recommendations for marketers in the export wool industry. The computer system is a program that stores and makes use of expert knowledge in the course of providing recommendations to a user. "Good" price recommendations are ones that closely resemble those of a consensus of experts.

Fuzzy logic, a method of inexact reasoning, is used to derive recommendations from inexact rules in the system. The system allows users to input vague expressions used in natural language, translates them into nonfuzzy values and then performs a set of operations on them to produce a nonfuzzy recommended price.

The effectiveness of the system in making good price decisions was tested by comparing the system's recommendations with consensus recommendations from a panel of experts. The correlation coefficient between the system's recommendations and the consensus recommendations was found to be higher than any of the correlation coefficients between the individual manager's first recommendations and the consensus recommendations. This suggests that the system constructed is capable of providing good price recommendations.

TABLE OF CONTENTS

	Page
Acknowledgement	i
Abstract	ii
List of Tables	vi
List of Figures	vii
Chapter 1	Introduction
1.1	Project Description 1
1.2	Organization of the Thesis 4
Chapter 2	Knowledge-based Systems in Marketing
2.1	Structure of Knowledge-based Systems 6
2.1.1	The Knowledge Base 7
2.1.1.1	Decision Rules 7
2.1.1.2	Semantic Networks 10
2.1.1.3	Frames 11
2.1.2	The Inference Engine 12
2.1.2.1	Backward Chaining 12
2.1.2.2	Forward Chaining 13
2.1.3	The User Interface 14
2.1.4	Conclusion 14
2.2	Review of Knowledge-based Systems in Marketing 15
2.3	Fuzzy Logic 23
2.3.1	The Fuzzy Logic Approach 23
2.3.1.1	The Conjunction Operator 27
2.3.1.2	The Disjunction Operator 28
2.3.1.3	The Negation Operator 30
2.3.1.4	The Conditional Operator 31
2.3.1.5	The Composition Operator 33
2.3.1.5.1	Hedging 34
2.3.1.5.2	Application of the Composition Operation 36
2.3.2	Conclusion 37
Chapter 3	Knowledge Engineering and Method
3.1	Knowledge Engineering 40
3.1.1	Introduction 40
3.1.2	Choice of Knowledge Engineering Methods in the Current Study 45
3.2	The Delphi Technique and its Application in Knowledge Engineering 46

3.3	The Application of the Delphi Technique in the Current Study	50
3.3.1	The Pilot Study	50
3.3.1.1	Method Used in the Pilot Study	50
3.3.1.2	Results of the Pilot Study	52
3.3.1.2.1	Process of Price Decision Making	52
3.3.1.2.2	Important Fuzzy Factors in the Decision Making Process	52
3.3.1.2.3	Decision Rules Used in the Decision Making Process	54
3.3.2	The Fieldwork for Initial Responses	54
3.3.2.1	Method	54
3.3.2.2	Results of the Fieldwork	61
3.3.2.2.1	Managers' Recommendations Given the Scenarios	61
3.3.2.2.2	Initial Membership Functions	61
3.3.2.2.3	Selection of Managers for Delphi Sessions.	61
3.3.3	The Delphi Sessions	62
3.3.3.1	Method	62
3.3.3.2	Results of the Delphi Study	64
3.3.3.2.1	Decision Rules	64
3.3.3.2.2	Membership Functions	64
3.3.3.2.3	Consensus Decisions	66
3.4	System Evaluation	69
Chapter 4	System Specification	
4.1	Philosophy for the Construction of TZ	72
4.2	Knowledge Stored in TZ	75
4.2.1	Membership Functions of Primary Fuzzy Variables	75
4.2.2	Relational Matrices	75
4.2.3	Formulae for Hedging Functions	76
4.2.4	Relatively Constant Facts	77
4.2.5	Accounting Data	78
4.3	Structure and Inference Mechanism in TZ ..	80
4.3.1	Components of TZ	80
4.3.2	The Inference Mechanism	82
4.4	Prolog: The Programming Language Used in TZ	89
4.4.1	Introduction to Prolog	89
4.4.2	Knowledge Representation in Prolog	90
4.4.3	Inference Procedure in Prolog	92
4.4.4	Reasons for Using Prolog as the Programming Language	96
Chapter 5	TZ, the Knowledge-based System for Price Decision Making	
5.1	Sample Run of TZ	101
5.2	Evaluation of TZ	110
5.2.1	The Objective Revisited	110
5.2.2	System Evaluation	111

Chapter 6	Conclusions and Directions for Future Research	113
References		115
Appendix 1		119
Appendix 2		123
Appendix 3		139
Appendix 4		156
Appendix 5		160

LIST OF TABLES

Table	Description	Page
3.1	Importance of Fuzzy Factors	53
3.2	Response Rates of the Delphi Sessions	63
3.3	Consensus Recommendations to Problem Scenarios	67

LIST OF FIGURES

Figure	Description	Page
2.1	Knowledge-based System Components	7
2.2	Tree-like Representation of Causes of Fall in Market Share	16
2.3	Relational Matrix	33
3.1	Stages of Knowledge Acquisition	42
4.1	Relational Matrix for Rule 1	76
4.2	Structure and Inference Mechanism of TZ	80
4.3	Flow Diagram Showing the Inference Mechanism in TZ	82
4.4	Tree Representation of the Above Example	96
4.5	Tree Solution of "member(2, [1,2,3])?"	98

CHAPTER ONE

INTRODUCTION

This chapter describes the objective of the study and provides an outline of the organization of this thesis.

1.1 Project Description

The objective of this study was to demonstrate that it is possible to construct a knowledge-based system that provides good price recommendations for a user in the export wool industry. A knowledge-based system is a computer program that stores and makes use of the knowledge of experts to make good recommendations for managers.

To evaluate the system, the recommendations the computer makes were compared with recommendations made by a set of experts. To do this, a set of scenarios describing the decision making environment was first constructed. The system was then used to generate a set of recommendations in response to these scenarios. Subsequently, expert managers were asked to make decisions given the same set of scenarios. Hence, there were eleven sets of recommendations. The first set was produced by the system, the others were produced by the ten managers. The Delphi technique was then used to arrive at a set of consensus recommendations. The consensus recommendations were assumed to be the correct solutions to the problems suggested by the scenarios. The system's recommendations and the managers' recommendations were then compared with the consensus

recommendations one by one. The closer each set of recommendations was to the consensus recommendations, the better the set of recommendations. The correlation coefficient was used as an indicator of how close each set of recommendations was to the consensus recommendations.

Fuzzy logic was used as the basis for making recommendations in the system. Fuzzy logic is a means of modelling inexact reasoning. The logic is suitable to handle the high degree of fuzziness associated with decision making variables in a marketing environment. Decision making variables in a marketing environment are mostly fuzzy. For example, the variable "competition" is fuzzy because it can take on fuzzy linguistic values like "strong", "very strong", "weak", "very weak" and so on. An expert marketer's decision making knowledge can be expressed as decision rules involving these fuzzy decision variables. These decision rules are also called fuzzy relations. An expert makes use of these decision rules in arriving at a decision. The following is an example. If "competition" and "price change" are two important fuzzy variables in a marketer's decision making environment, then a possible fuzzy relation between the two variables is:

If competition is **strong**, then price change should be **negative**.

The knowledge-based system reported in this thesis is an attempt to represent and make use of fuzzy knowledge in the problem area of price decision making in the export wool

industry so that recommendations generated by the system are better than those produced by a good manager. Such a system is an important starting point for further development of expert systems in marketing.

In order to construct the knowledge-based system, a formal means of representing fuzzy variables and fuzzy relations must be established. In addition, a calculus must be established to enable recommendations to be deduced from the fuzzy relations and the fuzzy variables. The representation schema and calculus used in this thesis were first formalized by Zadeh (1965, 1968, 1973, 1975a, 1975b, 1976a, 1976b, 1979).

The system is written in Turbo Prolog and is implemented on an IBM PCAT microcomputer.

1.2 Organization of the Thesis

The rest of the thesis is divided into five chapters. Chapter 2 contains a comprehensive review of knowledge-based systems for marketing decision making. Section 2.1 of Chapter 2 describes the various components of a knowledge-based system. Section 2.2 then reviews and evaluates the various knowledge-based systems constructed so far in the area of marketing decision making. Section 2.3 attempts to justify the use of fuzzy logic in the construction of the system.

Knowledge engineering and the problems of knowledge acquisition are discussed in Chapter 3, section 3.1. The Delphi technique is put forward in section 3.2 as an ideal method to use for transferring the experts' knowledge into a set of decision rules. The method of conducting the Delphi sessions is discussed in section 3.3. Finally, section 3.4 puts forward a method for testing the effectiveness of the knowledge-based system constructed.

Chapter 4 contains the system specification and describes the system in detail. Section 4.1 describes the philosophical basis for the construction of the system. Section 4.2 describes how the knowledge is represented in the system. Section 4.3 describes the inference mechanism in the system. It also provides a detailed description of the structure of the system. Section 4.4 provides the justification for using Prolog as the programming language for the system.

In Chapter 5, a sample run of the system is provided and an evaluation of the system is also made. Finally, Chapter 6 provides conclusions and directions for future research.

CHAPTER TWO

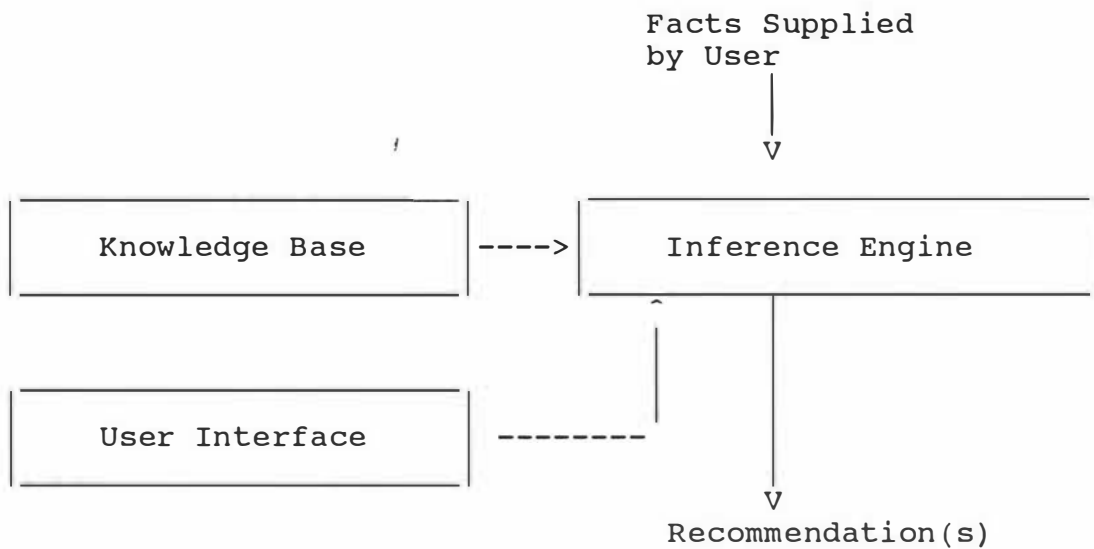
KNOWLEDGE-BASED SYSTEMS IN MARKETING

This chapter is divided into three sections. Section 2.1 gives a general description of the different components of a knowledge-based system. Section 2.2 reviews a number of applications of knowledge-based systems in marketing. Section 2.3 describes fuzzy logic and attempts to illustrate the point that fuzzy logic is a suitable tool to use when modelling a system with the high degree of vagueness characteristic of the marketers' decision making environment.

2.1 Structure of Knowledge-based Systems /

Knowledge-based system architecture is centered around a knowledge-base and an inference engine. The inference engine enables inferences to be drawn using the knowledge, in the form of facts and decision rules, stored in the knowledge-base. The following is a diagrammatic representation of a knowledge-based system:

Figure 2.1 Knowledge-based System Components



2.1.1.1 The Knowledge Base

The knowledge base contains all the decision making knowledge of experts in the problem domain. Based on the knowledge in the knowledge base and the facts supplied by the user if necessary, the inference engine deduces a recommendation for a user. The decision making knowledge is usually represented as decisions rules, semantic networks and frames.

2.1.1.1.1 Decision Rules

The decision making knowledge in the problem area can be represented in the form of decision rules. These decision rules are made up of a condition component and an action component. The action is recommended if the conditions are satisfied. Rules without a condition component are called facts. Some facts are stored in the system and some are supplied by the user during run time.

Most decision rules in knowledge-based systems are rules of thumb used by experts in arriving at their everyday decisions. These rules are sometimes called heuristics. The following is a typical rule stored in the system:

If the cashflow is favorable, then the price change is positive.

A typical fact contained in the knowledge base is:

Mark-up is 2.1%.

Derivation of a recommendation involving these rules is usually done by modus ponens. The following deduction schema is called modus ponens:

Given	If A then B
and	A

then	B may be inferred

For example, given the above decision rule "If the cashflow is favorable, then the price change is positive." and the user's input of a favorable cashflow situation, then the recommendation from the system will be "the price change is positive".

Fuzzy logic uses an extension of modus ponens called generalized modus ponens. Generalized modus ponens enables deductions to be drawn when the decision rules and the facts contain fuzzy linguistic values. Generalized modus ponens

is shown as below:

Given	If A' then B'
and	A"

then	B" may be inferred

where A', B' A" and B" are all propositions involving fuzzy linguistic values.

The following is an example of generalized modus ponens:

Given	If the market situation is favorable, then the price change is positive.
and	The market situation is very favorable.

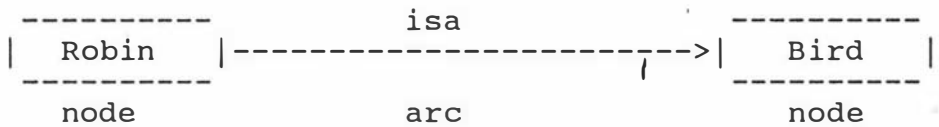
then	The price change is large positive.

The four fuzzy linguistic values used in the above schema are "favorable market situation", "positive price change", "very favorable market situation" and "very positive price change". The first "if ... then ..." clause describes the fuzzy relationship between "favorable market situation" and "positive price change". The fuzzy relationship can be represented by a relational matrix (Zadeh 1973). The second fuzzy proposition "the market situation is very favorable" is represented by a membership function. The conclusion "The price change is large positive" is deduced by the formula provided by Zadeh (1973). The formula will be described in section 2.3.1.5.

2.1.1.2 Semantic Networks

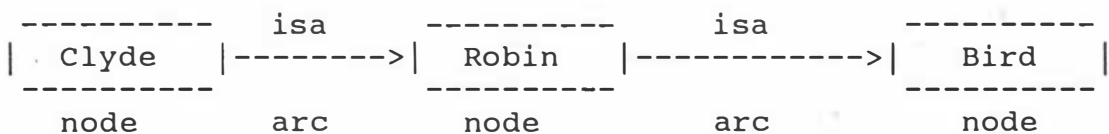
Besides representing knowledge in the form of conditional statements, it is possible to represent knowledge by means of semantic networks.

A semantic network consists of nodes and arcs connecting the nodes. Nodes are used to represent objects or concepts in the problem domain, and the arcs represent the relations between them. Both the nodes and arcs are labeled. An example given in Barr and Feigenbaum (1981) is as follow:



The above network represents a fact "All robins are birds".

The fact that Clyde is a robin can be added to the network by introducing an extra node into the above figure. This is shown below:



The semantic network representation of knowledge not only enables the representation of relationships between objects, but it also enables deductions to be drawn. For example, the fact that Clyde is a bird can be deduced by following the arcs connecting the objects in the network.

2.1.1.3 Frames

Frames were originally proposed by Minsky (1975). They are used to represent knowledge about objects and events typical to specific situations. A frame contains slots, the places where knowledge about objects is inserted. The following example is a frame representing the generic concept of chair (Barr and Feigenbaum 1981):

Chair Frame

Specialization-of:	furniture
No-of-legs:	an integer (default = 4)
Style-of-back:	straight, cushioned,

Slots

Value of slots

The above frame contains three slots: Specialization-of, No-of-legs and Style-of-back. The value of the slot "Specialization-of" is "furniture". The default value of the slot "No-of-legs" is 4. The slot "Style-of-back" can take on a number of possible values such as "straight", "cushioned" and so on.

A frame of a particular chair has the same slots "inherited" from the chair frame, but the contents could be made more specific. Deduction in the frame representation can be done by attaching a procedure to a slot to determine its value. An example to explain deduction in the frame representation is the addition of an extra slot called "price" to the above

frame. The value of the slot can be worked out as below:

```
If customer-type = "retailer"
then price = mark-up-price * 70%
else if customer-type = "wholesaler"
    then price = mark-up-price * 60%
    else price = mark-up-price.
```

2.1.2 The Inference Engine

The derivation of a recommendation in a knowledge-based system is done by the inference engine containing derivation strategies. There are basically two different derivation strategies, backward chaining and forward chaining.

2.1.2.1 Backward Chaining

Backward chaining is goal-directed. A goal is often formulated as a hypothesis to be tested. Often the hypothesis testing entails formulating and testing intermediate hypotheses (or subgoals). The following is an example of backward chaining using three decision rules:

Rule 1: If A and C then F

Rule 2: If B then C.

Rule 3: If F then G.

Assuming that A and B are known to be true, the hypothesis to be tested is "G is true".

Using backward chaining, rule 3 is selected to start the derivation because rule 3 says that "G is true if F is

true". That is, to test the hypothesis that "G is true", it is necessary to test the intermediate hypothesis that "F is true". To test this intermediate hypothesis, rule 1 is selected.

According to rule 1, F is true if A and C are both true. At this stage, the two intermediate hypotheses are A and C. A is known to be true already. If C can be shown to be true, then F is true.

To determine if C is true, rule 2 is used because the conclusion part of rule 2 is C. According to rule 2, C is true if B is true. It is known already that B is true, therefore C is also true. Since both A and C are true, F is true, and G must be true as well.

2.1.2.2 Forward Chaining

Forward chaining, on the other hand, is data driven, or bottom-up. It starts from the available information as it comes in and tries to draw conclusions that are appropriate to the goals.

Using the same example as described above, and assuming that the same hypothesis is tested and the same facts are known, forward chaining starts the derivation with the premises that A and B are true.

Rule 2 is now selected to start the derivation because B is the premise of the rule. This will establish that C (the conclusion of the rule) is true and C is now a fact.

Because both A and C are true, rule 1 can be applied and it can be established that F is true. Since F is true, rule 3 is now "fired", producing the conclusion that G is true.

2.1.3 **The User Interface**

The user interface is an essential component of a knowledge-based system. An effective user interface provides clear instructions and help facilities to make it easy for a user to use the system. It is essential for ensuring user acceptance of the system. With a computer operated system, a typical help facility provides instructions as to how to proceed when the help function key is pressed.

2.1.4 **Conclusion**

On the basis of this general introduction to the structure of a knowledge-based system, section 2.2 reviews a number of applications of knowledge-based systems in Marketing, and draws some general conclusions from the review.

2.2 Review of Knowledge-based Systems in Marketing

Only a few knowledge-based systems have been constructed for supporting marketing decision making. The following is a review of these systems.

Levine, Drang and Edelson (1986) built an expert system to assess the chance of a sale being concluded given the personality traits of the salesperson and the consumer. The purpose of the system was to provide advice to salespersons as to the best approach to be used in a sales situation. The system does not handle vagueness in the inference process. No evaluation was done on the effectiveness of the system in providing quality recommendations.

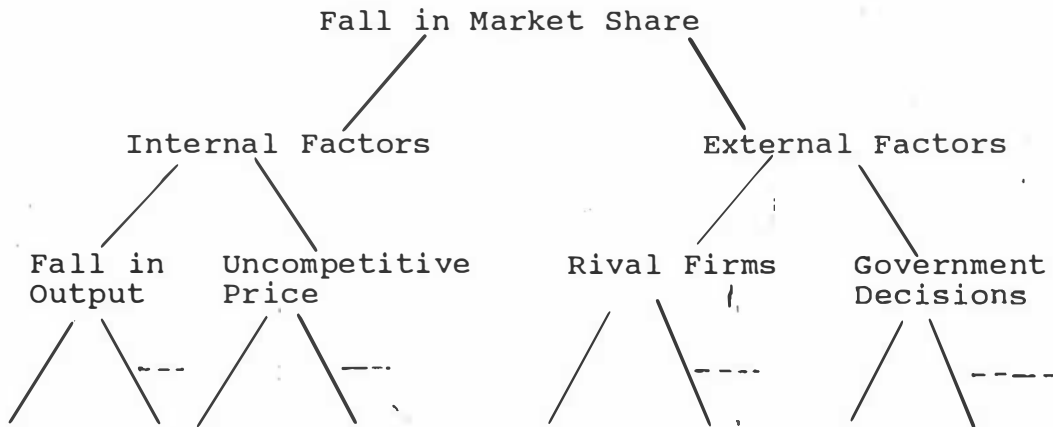
Tse and Syed (1988) developed a consulting system for competitive analysis and planning control to support business decision making in industries with a high degree of product differentiation and multiproduct production facilities. The knowledge in the system is represented by frames. The system does not handle situations of vagueness.

The system was used to determine the market share of various producers in the US polypropylene industry in 1983. The system's estimates were compared with the estimates made by SRI International. SRI's estimates were derived from market research results. The two sets of estimates were found to agree with each other closely.

Dhananjayan, Janakiraman and Sarukesi (1988) reported a system for identifying the responsible factors for a firm's

fall in market share using multiple-loop causal factor interaction modelling. In the system, all factors affecting market share are arranged in a tree-like structure as shown below:

Figure 2.2 Tree-like Representation of Causes of Fall in Market Share



Multiple-loop causal factor interaction modelling refers to a systematic depth-first search with backtracking to traverse the tree shown above for all the possible causes to a fall in market share. The system does not handle vagueness. No evaluation was done to determine whether the system was successful.

Mockler (1989) reported three prototype systems of knowledge-based systems. The first prototype system was designed for managers of health care companies. The system's objective was to assist a manager to evaluate a new product's potential in the local market and make recommendations as to whether a specific product should be introduced, and if so, how. The second system was a

database oriented system. The system was made up of two data bases: a customer data base and a product data base. The system asks a user a number of questions about the customer and the product and then uses the decision rules stored in the system to give specific media selection recommendations for consumer product companies. The third prototype system was designed to help sales managers establish for each salesperson fair and reasonable sales quotas for a brand of mouthwash in a local sales territory. The system uses production rules and backward chaining and the information provided by the user to reach its recommendation. The information provided by the user includes individual past sales performance, demographics, economic conditions, seasonality, competition and so on. None of the three prototype systems has the capacity to deal with situations of vagueness. The prototype systems were evaluated by the experts involved in the building up of the system. All the prototype systems were judged to be performing successfully.

Parkinson and Parkinson (1987) reported several applications of expert systems. One was in media planning, another was in retail site selection. The third one was for assessing consumer reactions to changes in the market mix. The fourth one was for determining competitors' reaction to the changes in a company's marketing strategies. Still another one was for building up econometric models of markets. All these systems were in the pilot stage of development. No

evaluation has been done on these systems.

From the above review, it can be seen that the number of applications of knowledge-based systems in Marketing is still very small. In fact, most systems developed so far are prototype systems only.

It appears that few of the systems were tested for effectiveness. For those systems that have been evaluated, evaluation was, in the main, carried out by the experts from whom the knowledge was obtained. A more objective method of evaluation would be preferred.

The only case where such objective criteria have been established was in the evaluation of Tse and Syed's system. Even so, there is still one problem with the method used in the evaluation of the system in that only data from 1983 were used. Data spanning years outside the year in which the data was gathered should be used to evaluate the estimates produced by the system before the system could be judged as performing at an acceptable level.

None of the systems developed has the capacity to handle the high degree of fuzziness characteristic of a marketing manager's decision making environment.

The lack of a tool for dealing with fuzziness was also identified by Zadeh (1973) and Ackoff (1979). They observed that, as the complexity of a problem increases, the ability to analyze it in precise and relevant terms decreases. There is a threshold beyond which complexity, precision and

significance can no longer coexist. Traditional science-oriented methodologies are incapable of dealing with a manager's decision making environment because these tools are reductionistic, analysis-oriented, and mechanistic (Ackoff 1979). Techniques in operations research and applied mathematics are not suitable for domains with a high degree of fuzziness. When these techniques are adapted for the analysis of humanistic systems, in particular the manager's decision making problem, they are bound to fail (Zadeh 1975b). As Zadeh (1973) put it, "the closer one looks at a real-world problem, the fuzzier becomes its solution".

An alternative approach is needed for real-life humanistic systems such as political, economic and business systems. Much of decision making in the real world takes place in an environment in which the goals, the constraints and the consequences of possible courses of action are not known precisely. Unless a tool can be developed for handling this kind of "fuzziness", any attempt to simulate management decision making on a computer will have limited success.

A typical way used by the systems reviewed above to circumvent the need to deal with the high degree of fuzziness is to build up levels or gradations for each fuzzy factor. An example of a decision rule from the media selection system reported in Mockler (1989) illustrates the approach:

```

If    customer-type = 'teenager' and
      media-preference = 'radio' and
      buying-decision = 'irrational' and
      product-type = 'convenience' or
      product-type = 'impulsive' and
      target-market = 'selective' and
      product-involvement = 'low' and
      geographic-reach = 'local' or
      geographic-reach = 'regional' and
      advertising-requirements = 'repetition-short-
                                duration-messages'

then media = 'radio'

```

In the above rule, the fuzzy factor, product-involvement, is divided into two levels: high and low. Similarly, another fuzzy factor, buying-decision, is divided into two levels: rational and irrational. There are two problems with this approach of representing knowledge about fuzzy factors in the form of levels. First, the system can only deal with a certain number of factor levels. For example, in the above example, the system can handle two levels of product-involvement and two levels of buying rationality. No provision has been made for other levels of product involvement such as "very high level of product involvement", "very very high level of product involvement", "slightly high level of product involvement", and so on. Similarly, other levels of buying rationality are not taken into account. The fact that there are an infinite number of levels for each factor is not dealt with. Because the

number of combinations expands exponentially, the system becomes too large to deal with all possible combinations of product-involvement levels and degree of rationality in buying-decision.

The second problem with the approach is that the levels might fail to be mutually exclusive when the decision making variable is fuzzy. For example, a particular level of product involvement could be categorized as a "high level of product involvement" or a "slightly high level of product involvement" simultaneously. The problem becomes more serious the fuzzier the decision making variable is.

To solve these problems, fuzzy logic can be used. Fuzzy logic tackles the first problem by using a set of hedge functions and logical operators. By using these functions and operators, fuzzy quantities can be manipulated in like manner as numbers are manipulated by arithmetic operators, thus avoiding the need to build up decision rules for each and every possible combination of factor levels. Details of the mechanism involved will be given in section 2.3.2.

The second problem is tackled in fuzzy logic by regarding factor levels as fuzzy subsets. Grades of membership of different magnitudes can be assigned to a given decision making situation reflecting the grades of membership of the situation in the different fuzzy subsets. For example, the grade of membership of a particular level of product involvement in the fuzzy subset "high level of product

involvement" could be 0.9 while its grade of membership in the fuzzy subset "slightly high level of product involvement" could be 0.2. That is, a particular fuzzy situation can belong to more than one fuzzy subset at the same time with different grades of membership. This makes fuzzy logic a good representation mechanism for fuzziness.

In the next section, the mathematics of fuzzy logic will be explored and its usefulness in simulating marketers' decision making under incomplete knowledge will be further elaborated.

2.3 Fuzzy Logic

This section explains how fuzzy logic can be a useful technique to use when there is a need to deal with a huge number of possible combinations of factor levels as mentioned in section 2.2. Section 2.3.1 describes fuzzy logic as an algebra and introduces the various useful operators. Section 2.3.2 provides the justification for the choice of fuzzy logic by showing how the algebra can be usefully applied in situations involving large number of fuzzy factors and factor levels.

2.3.1 The Fuzzy Logic Approach

Fuzzy logic is based on the concept of a fuzzy subset¹. The idea of a fuzzy subset can best be illustrated by comparing it with an ordinary set.

The main difference between classical set theory and fuzzy subset theory is in enumerability. In classical set theory it is possible to enumerate the elements belonging to the set. In fuzzy subset theory it is not possible to enumerate the elements belonging to the subset.

¹ Kaufmann (1975) gives the following explanation regarding the use of the term fuzzy subset instead of fuzzy set: "Why do we use the term fuzzy subsets instead of fuzzy set? This is because a fuzzy set will never be a concept proper to the present theory; the reference set will always be an ordinary set, that is, such as one defined intuitively in modern mathematics, that is again, a collection of well-specified and distinct objects. It is the subsets that will be fuzzy."

According to Zadeh (1973), a fuzzy subset A of a universe of discourse U is characterized by a membership (or compatibility) function $u : U \rightarrow [0,1]$ which associates with each element y in U a number $u_A(y)$ in the interval $[0,1]$ representing the grade of membership (or degree of compatibility or degree of truth) of y in A . That is, if A is a finite subset of U with elements y_1, y_2, \dots, y_n then the fuzzy subset can be expressed as:

$$A = (u_A(y_1)/y_1, u_A(y_2)/y_2, \dots, u_A(y_n)/y_n)$$

In the above formula, the expression above the stroke, $u_A(y_i)$, is the grade of membership of y_i in the fuzzy subset A where y_i is the i th element of U and $i = 1, 2, \dots, n$.

The theory of ordinary sets is a particular case of fuzzy subsets. The characteristic function $u(y)$ is defined as $u_A(y) = 1$ if $y \in A$ and $u_A(y) = 0$ if $y \notin A$.

An example can be used to illustrate the idea of a fuzzy subset. Consider the fuzzy subset "slight increase in price". To characterise this fuzzy subset, it is necessary to identify the universe of discourse. The universe of discourse in this case is the range of possible price increases. Suppose that the range of possible price increases is from \$1 to \$10, then the fuzzy subset "slight increase in price" can be expressed as below:

$$\text{slight price increase} = (1/\$1, 1/\$2, 0.9/\$3, 0.9/\$4, \\ 0.8/\$5, 0.76/\$6, 0.5/\$7, 0.3/\$8, 0/\$9, 0/\$10)$$

The number above the stroke represents the grade of membership of the price increase stipulated under the stroke. Hence, the grade of membership of a \$1 increase in price and the grade of membership of a \$2 increase in price in the fuzzy subset "slight price increase" are both 1. That is, it is 100% certain that a \$1 price increase and a \$2 price increase should belong to the fuzzy subset "slight price increase". On the other hand, the grade of membership of a \$3 increase in price in the fuzzy subset "slight price increase" is 0.9. That is, it is less than 100% certain that a \$3 increase in price should be categorized as a "slight price increase". The membership value of 0.9 reflects the degree of vagueness of the situation when the decision maker is uncertain as to whether or not a \$3 increase in price should be categorized as a "slight price increase". The other membership values can be interpreted in like manner. Finally, when the membership value is zero, it means that the corresponding price increase does not belong to the fuzzy subset "slight price increase" at all.

Like traditional boolean logic, fuzzy logic can be described mathematically as an algebra by considering it as a system containing a set of fuzzy propositions and a number of binary² operators. In order that these operators can be

A binary operator is a single-valued function assigning for any two elements in an algebra a value in the algebra. For example, the addition operator '+' in arithmetic assigns a single value 5 to '2 + 3' where 5, 2 and 3 are all elements of the algebra of addition on natural numbers.

applied to the fuzzy propositions to produce new fuzzy propositions, the original fuzzy propositions in the system must be expressed in the form of fuzzy subsets of predetermined universes of discourse. Consider the following simple case when the system contains the following three fuzzy propositions:

Slight price increase;

Large price increase; and

Favorable cashflow.

The first two fuzzy propositions can be represented in the form of fuzzy subsets by referring to the universe of discourse of possible price increases. The fuzzy subset for "slight price increase" has been shown above. The fuzzy subset for "large price increase" is shown below:

large price increase = (0/\$1, 0/\$2, 0.1/\$3, 0.2/\$4,
0.3/\$5, 0.4/\$6, 0.5/\$7, 0.6/\$8, 0.75/\$9, 1/\$10)

To get the fuzzy subset representation for the last fuzzy proposition, consider the simplified case that there are only five possible positive bank balances: \$1 million, \$2 million, \$3 million, \$4 million and \$5 million. A possible fuzzy subset representation for the fuzzy proposition is as below:

favorable cashflow = (0.2/\$1 million, 0.4/\$2 millions,
0.8/\$3 millions, 0.9/\$4 millions, 1/\$5 millions)

After all the fuzzy propositions have been transformed into fuzzy subsets, binary operators can be defined and applied to these fuzzy subsets to get new fuzzy subsets. Five operators can be defined for the algebra. They are the conjunction operator, the disjunction operator, the negation operator, the conditional operator and the composition operator. Each one of these operators are discussed below.

2.3.1.1 The Conjunction Operator, \wedge

The conjunction operator, \wedge , can be defined as the minimum of the grades of membership of an element in the original fuzzy subsets. The conjunction operator is commonly labelled as "and". Symbolically, it can be represented by the following formula:

$$u_{S_i \wedge S_j}(y) = \min(u_{S_i}(y), u_{S_j}(y)) \text{ ----> 2.1}$$

where y is an element in the universe of discourse;

\min stands for minimum; and

S_i and S_j are any two fuzzy subsets in the universe of discourse.

Using the first two fuzzy propositions mentioned above, the new fuzzy proposition "slight price increase and large price increase" can be obtained by applying formula 2.1 to the two fuzzy subsets representing the fuzzy propositions "slight price increase" and "large price increase". The resulting fuzzy proposition is represented by the following fuzzy subset:

slight price increase and large price increase =

(0/\$1, 0/\$2, 0.1/\$3, 0.2/\$4, 0.3/\$5, 0.4/\$6,
0.5/\$7, 0.3/\$8, 0/\$9, 0/\$10)

The above fuzzy subset could be interpreted as "medium price increase".

2.3.1.2 The Disjunction Operator, \vee

The second operator is the disjunction operator. It is commonly labelled as "or". Using the same symbols above, the disjunction operator, \vee , can be defined by the following formula:

$$u_{Si \vee Sj}(Y) = \max(u_{Si}(Y), u_{Sj}(Y)) \text{ -----} \rightarrow 2.2$$

where max stands for maximum.

Using the first two fuzzy subsets above, a new fuzzy proposition "slight price increase or large price increase" can be obtained using the or operator. The resulting fuzzy proposition is represented by the following fuzzy subset:

slight price increase or large price increase =

(1/\$1, 1/\$2, 0.9/\$3, 0.9/\$4, 0.8/\$5, 0.76/\$6,
0.5/\$7, 0.6/\$8, 0.75/\$9, 1/\$10)

The above fuzzy subset could be interpreted as "either small or large price increase".

The min and max operators mentioned above are only two of the possible operators for conjunction and disjunction of fuzzy subsets. Other operators could be defined for the

conjunction and disjunction of fuzzy subsets. Examples are Yager's operators (in Zimmermann 1985) and the Hamacher operators (Hamacher 1976).

Bellman and Giertz (1973) addressed the problem of choice of operators for conjunction and disjunction as follows. Consider two fuzzy statements with membership values $u_A(x)$ and $u_B(x)$, where $u_A(x)$ and $u_B(x)$ are both between 0 and 1. In this case, A and B are two fuzzy subsets of interest and x is an element in the universe of discourse. The membership values of the conjunction and disjunction of the two statements are the values of the two membership functions f and g respectively. The values of f and g are in the interval $[0,1]$, where $f(u_A(x), u_B(x)) = u_A(x) \wedge u_B(x)$, $g(u_A(x), u_B(x)) = u_A(x) \vee u_B(x)$. Bellman and Giertz felt that the following restrictions are reasonably imposed on f and g:

- a) f and g are nondecreasing with respect to $u_A(x)$ and $u_B(x)$. That is, $u_A \wedge u_B(x)$ or $u_A \vee u_B(x)$ cannot decrease when $u_A(x)$ or $u_B(x)$ increases.
- b) The associative law holds for the two operators.
- c) f and g are symmetric, that is:

$$f(u_A(x), u_B(x)) = f(u_B(x), u_A(x)) \text{ and}$$

$$g(u_A(x), u_B(x)) = g(u_B(x), u_A(x))$$

Hence, the associative law holds for the two operators.

d) For x_1 and x_2 in the universe of discourse, if $u_A(x_1) = u_B(x_1) > u_A(x_2) = u_B(x_2)$, then the membership values of x_1 in "A or B" or "A and B" are greater than those of x_2 .

e) $f(u_A(x), u_B(x)) \leq \min(u_A(x), u_B(x))$ and $g(u_A(x), u_B(x)) \geq \max(u_A(x), u_B(x))$.

This means that accepting the intersection of the two statements requires more while accepting the truth of one statement or the other requires less than accepting each individual one alone.

f) $f(1,1) = 1, g(0,0) = 0$.

g) $u_A(x) \wedge (u_B(x) \vee u_C(x))$ is equivalent to $(u_A(x) \wedge u_B(x)) \vee (u_A(x) \wedge u_C(x))$.

$u_A(x) \vee (u_B(x) \wedge u_C(x))$ is equivalent to $(u_A(x) \vee u_B(x)) \wedge (u_A(x) \vee u_C(x))$.

Bellman and Giertz (1973) showed that the min and max operators are the only operators for conjunction and disjunction that satisfy the above criteria. The min and max operators were therefore used in the system reported in this thesis.

2.3.1.3 The Negation Operator, \neg

The third operator is the negation operator, \neg . The operator is usually given the label "not". It can be defined by the following formula:

$$u_{\neg Si}(Y) = 1 - u_{Si}(Y) \text{ -----} \rightarrow 2.3$$

For example, the fuzzy proposition "not slight price increase" can be represented by a fuzzy subset using formula 2.3. The resulting fuzzy subset is shown below:

$$\begin{aligned} \text{not slight price increase} = & (0/\$1, 0/\$2, 0.1/\$3, \\ & 0.1/\$4, 0.2/\$5, 0.24/\$6, 0.5/\$7, 0.7/\$8, 1/\$9, \\ & 1/\$10) \end{aligned}$$

It can be shown that the above negation operator satisfies the restrictions imposed by Giertz and Bellman (1973) on a complementation function. These restrictions are as follows:

- a) $u_{\neg Si}(Y)$ depends only on $u_{Si}(Y)$.
- b) If $u_{Si}(Y) = 0$ then $u_{\neg Si}(Y) = 1$; and
if $u_{Si}(Y) = 1$ then $u_{\neg Si}(Y) = 0$.
- c) The function is continuous and monotonically decreasing when $u_{Si}(Y)$ increases.
- d) The function is involutive.
- e) A certain change in the value of $u_{Si}(Y_1)$ is accompanied by the same amount of change in the value of $u_{\neg Si}(Y_1)$.

2.3.1.4 The Conditional Operator, \supset

The fourth operator is the conditional operator. The

conditional operator actually describes a relation between the elements of two fuzzy subsets. It is used to symbolize an if ... then ... relationship between two fuzzy propositions. Mathematically, a fuzzy relation R from a fuzzy subset X to a fuzzy subset Y is a fuzzy subset of the Cartesian product $X * Y$. If x_1, x_2, \dots, x_n are the elements of X and y_1, y_2, \dots, y_n are the elements of Y , R can be represented as below:

$$R = (u_R(x_1, y_1)/(x_1, y_1), u_R(x_1, y_2)/(x_1, y_2), \dots, \\ u_R(x_1, y_n)/(x_1, y_n), \\ u_R(x_2, y_1)/(x_2, y_1), \dots, \\ u_R(x_n, y_2)/(x_n, y_2), \dots, \\ u_R(x_n, y_n)/(x_n, y_n))$$

In the above formula, the expression above the stroke is the grade of membership of the ordered pair below the stroke.

The result of applying the conditional operator to two fuzzy subsets C and S will be a two-dimensional relational matrix. The formula for the conditional operator, \supset , is as below:

$$u_{C \supset S}(y_i, z_j) = \max((1 - u_C(y_i), u_S(z_j)) \dots \dots \dots 2.4$$

where y_i is the i th element in the fuzzy subset C ; and

z_j is the j th element in the fuzzy subset S .

The above relation is associative, transitive, symmetric and reflexive.

Hence, the fuzzy decision rule "if the cashflow is favorable then the price change is slight positive" can be symbolized

by applying the conditional operator to the two corresponding fuzzy subsets. Let the fuzzy subset denoting the fuzzy proposition "the cashflow is favorable" be C and the fuzzy subset denoting the fuzzy proposition "the price change is slight positive" be S, then the following relational matrix can be obtained by equation 2.4:

Figure 2.3 Relational Matrix

		S									
		\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$10
C	\$1m*	1	1	0.9	0.9	0.8	0.8	0.8	0.8	0.8	0.8
	\$2m	1	1	0.9	0.9	0.8	0.76	0.6	0.6	0.6	0.6
	\$3m	1	1	0.9	0.9	0.8	0.76	0.5	0.3	0.2	0.2
	\$4m	1	1	0.9	0.9	0.8	0.76	0.5	0.3	0.1	0.1
	\$5m	1	1	0.9	0.9	0.8	0.76	0.5	0.3	0	0

* m stands for million.

2.3.1.5 The Composition Operator, \circ

Finally, the composition operator, denoted by the symbol \circ , can be used to make an inference based on a given fact and a decision rule. For example, given the fuzzy fact C' and the decision rule $C \supset S$, the composition of these two propositions gives the fuzzy subset S'. The operator is defined as:

$$\mu_{C' \circ C \supset S}(Y_j) = \max(\min(\mu_{C'}(Y_i), \mu_{C \supset S}(Y_i, Z_j)))$$

-----> 2.5

Assume that the fuzzy subset C' represents the fuzzy proposition "cashflow is very favorable". Using the concept of hedging suggested by Zadeh (19776a) and described in the next section, it is possible to derive the membership function for C' .

2.3.1.5.1 Hedging

Hedging works on linguistic variables³. In the linguistic approach, a role comparable to that of a unit of measurement is developed by hedging primary fuzzy subsets. That is, other subsets in the term set can be generated through the

³ Zadeh (1973) defined a linguistic variable as characterized by a quintuple $(x, T(x), U, G, M)$ in which x is the name of the variable; $T(x)$ (or simply T) denotes the term-set of x , that is, the set of names of linguistic values of x ; G is a syntactic rule (which usually has the form of a grammar) for generating the names, $T(x)$ of x ; and M is a semantic rule for associating with each linguistic value its meaning, $M(x)$, which is a fuzzy subset of U . U is the universe of discourse.

Linguistic variables are used extensively in all natural language. Natural languages consist of linguistic variables that take on fuzzy values. For example, the linguistic variable "high" might take on syntactic values like "high", "not high", "somewhat high", "very high", "not very high", "very very high", "high but not very high", "quite high", "more or less high". These syntactic values are collectively known as the term set, $T(X)$, of the linguistic variable "high". Each of the values in the term set is composed of a semantic value as well. The semantic value is the membership function of the fuzzy subset concerned. Once the membership function of the primary linguistic variable "high" is defined, the membership function of other linguistic values can be determined by hedging.

use of linguistic hedges such as "more or less", "very", "quite", "extremely" and so on. Each one of these hedges can be represented as a function of the primary fuzzy subset "high". The following hedge function (Zadeh 1976a) shows how the fuzzy subset "very high" can be obtained from the primary fuzzy subset "high". The hedge function in this example defines the membership function of "very high" as the square of the membership function of "high":

$$u_{H'}(Y) = u_H(Y)^2$$

where H represents the primary fuzzy subset "high"; and

H' represents the fuzzy subset "very high".

The following are other possible hedge functions suggested by Zadeh (1976a). In the formulae below, x stands for a primary linguistic variable.

$$\text{very } x = x^2$$

$$\text{very very } x = (\text{very } x)^2 = x^4$$

$$\begin{aligned} \text{very not exact} &= \text{very inexact} \\ &= (\neg \text{exact})^2 \end{aligned}$$

$$\text{not very exact} = \neg(\text{very exact}) = \neg(\text{exact}^2)$$

$$\text{plus } x = x^{1.25}$$

$$\text{minus } x = x^{0.75}$$

$$\text{plus plus } x = \text{minus very } x$$

$$\text{highly } x = \text{minus very very } x$$

In general, hedge functions can be defined in the following manner:

$$f(x) = x^a$$

where a is some real value.

Thus, it can be seen that a key feature of the fuzzy linguistic approach has to do with the use of hedging in providing a substitute for the basic notion of a unit of measurement so that a calculus can be established for manipulating vague and imprecise concepts. This basic unit of measurement accounts for much of the power of the fuzzy linguistic approach for modelling fuzziness in humanistic systems.

2.3.1.5.2 Application of the Composition Operation

With the use of hedging, it is possible to calculate the membership function of C' using the formula $u_{C'}(y) = u_C(y)^2$

where C represents the fuzzy subset "favorable cashflow" described on page 33.

$u_C(y)$ represents the membership value of y in the fuzzy subset C ; and

C' represents the fuzzy subset "very favorable cashflow".

The fuzzy subset C' can be expressed as below:

very favorable cashflow = (0.04/\$1 million,
0.16/\$2 millions, 0.64/\$3 millions,
0.81/\$4 millions, 1/\$5 millions)

Applying formula 2.5 to C' and the relational matrix in figure 2.3, the induced fuzzy subset S' is:

(1/\$1, 1/\$2, 0.9/\$3, 0.9/\$4, 0.8/\$5, 0.76/\$6, 0.5/\$7,
0.3/\$8, 0.2/\$9, 0.2/\$10)

The fuzzy subset S' can be labelled as "a little bit more than slight price increase" because the grades of membership of large price increases are larger than the corresponding membership values in the fuzzy subset "slight price increase" shown on page 24.

With the above five operators, inferences in fuzzy logic can be performed mechanically on fuzzy subsets. The formulae given can also be generalized to cases involving more than two fuzzy subsets.

2.3.2 Conclusion

Fuzzy logic is the appropriate technique to use when there is a need to deal with a large number of fuzzy variables and combinations of factor levels as described in section 2.2. The reason is that fuzzy logic enables a mathematical relation to be established between fuzzy linguistic variables. Through the use of hedging functions, recommendations can be made easily using the composition operator described above. There is no need to exhaust all

the possible decision rules relating the different levels of fuzzy variables. In the fuzzy logic approach, it is sufficient to have one decision rule relating each pair of primary fuzzy variables.

Consider the following example of a decision rule describing the conditional relationship between favorable cashflow and a positive change in price:

If the cashflow is favorable, then the price change is positive.

Suppose that the current cashflow situation is "very favorable", without the use of fuzzy logic it is not possible to create a system to recommend the optimal level of price change unless there is a decision rule in the knowledge base that describes the relationship between "very favorable cashflow" and degree of price change. For example, only if the following decision rule is stored in the knowledge base:

If the cashflow is very favorable, then the price change is large positive.

will a (nonfuzzy) system be able to recommend a "large positive change" in price.

Using the fuzzy logic approach, however, the second decision rule does not need to be explicitly stored in the knowledge base. This is because the fuzzy subset "very favorable cashflow" can be defined as a function of the primary fuzzy

subset "favorable cashflow" as follow (Zadeh 1976a):

$$\text{very favorable cashflow} = (\text{favorable cashflow})^2$$

If the first decision rule is represented in the form of a relational matrix in the knowledge base, then an induced decision can be derived based on the max-min compositional rule of inference shown in section 2.3.1.5 above. In other words, in the fuzzy logic approach, it is not necessary to exhaust all the possible combinations of all the possible levels of each one of the fuzzy decision variables in the decision making environment. This is a very important advantage of fuzzy logic over the other approaches because while the number of decision rules grows exponentially with the number of fuzzy variables, the number of hedge functions grows only proportionally. As an example, consider the case when there are only two fuzzy factors and that each factor is made up of nine levels, then the experts involved in the construction of the system will have to provide 81 decision rules if fuzzy logic is not used. Using the fuzzy logic approach, however, these experts will only need to provide one decision rule involving the primary fuzzy subsets and 18 hedge functions. Comparatively speaking, the task of establishing these hedge functions is much less burdensome than that of establishing all possible decision rules.

CHAPTER 3

KNOWLEDGE ENGINEERING AND METHOD

The first step in building a knowledge-based system is knowledge acquisition. The process of knowledge acquisition is commonly known as knowledge engineering. This chapter is divided into four sections. Section 3.1 describes the process and the popular methods of knowledge engineering. The problems involved in the process are also discussed in the section. Some additional problems arising from knowledge acquisition from marketing managers are also highlighted. To solve these problems, the Delphi technique was put forward as a good method to use in this research. Section 3.2 describes the Delphi technique and the justifications for adopting the technique. Section 3.3 describes how the Delphi technique was actually applied in the research. Section 3.4 puts forward a method for testing the effectiveness of the knowledge-based system constructed.

3.1 Knowledge Engineering

3.1.1 Introduction

The term knowledge engineering was coined by Feigenbaum (1977) as a process of reducing a large body of knowledge to a precise set of facts and rules. Potential sources of problem-solving knowledge are experts, textbooks, databases, case studies and personal experience. Shaw and Gaines (1987) defined knowledge engineering as "the acquisition, elicitation, structuring and encoding of knowledge for application in inferential, goal-directed, explanatory,

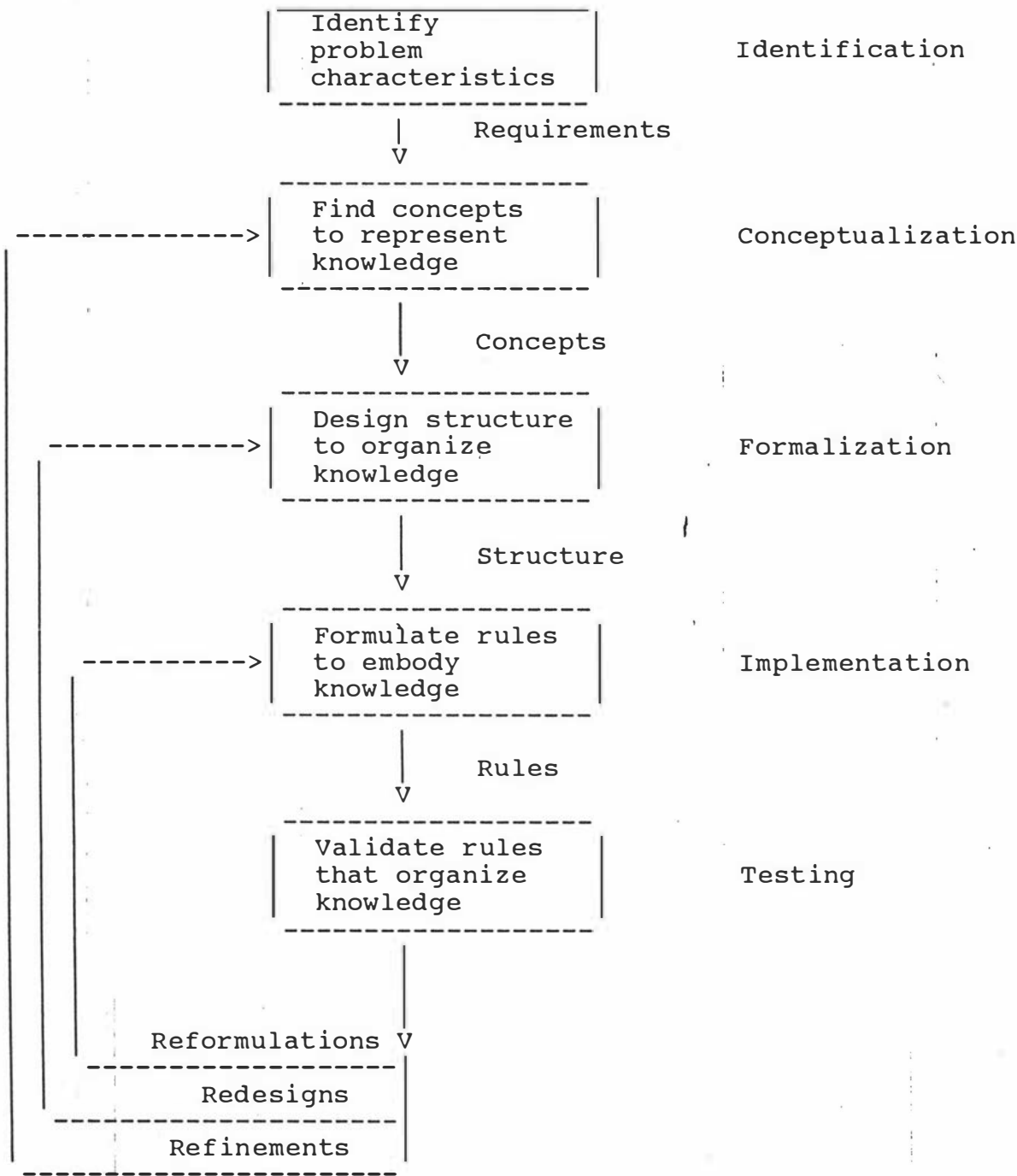
decision and action support systems".

Basically, there are two approaches to knowledge engineering (Waterman 1986): observational and intuitive. The observational approach is sometimes referred to as protocol analysis. This method selects a sample of the problem space and the experts are observed in the process of solving the problem. The method of solving the problem is recorded. Smith and Baker (in Boose (1986)), the builders of Dipmeter Advisor, used novel problems and then recorded the problem solving process.

The intuitive method is based on self-introspection. It requires experts to describe their cognitive processes in solving a problem and coming to a recommendation.

The steps in conducting a knowledge engineering exercise have been described by a number of authors. Buchanan and Shortliffe (1984) offer the following step-by-step approach to knowledge acquisition:

Figure 3.1 Stages of Knowledge Acquisition



Hayes-Roth et al. (1983) also outlined similar stages of expert system knowledge acquisition that correspond roughly to traditional system analysis and design, with more

emphasis placed on incremental and gradual development.

Wielinger and Breuker (in Merry 1985) defined three stages of knowledge engineering. The first stage is knowledge identification, when knowledge engineers record what experts report on how they solve a problem. The second stage is known as knowledge conceptualization, during which knowledge engineers try to build up a formal description of the primitive concepts in the reasoning processes. These primitive concepts are entities in the problem domain and the relations amongst these entities. The last stage is epistemological analysis. At this stage, knowledge engineers attempt to uncover the conditional relationships among the primitive concepts identified in the knowledge conceptualization stage.

There are three problems associated with knowledge engineering. The first and the most serious problem is that no scientific model has been established for the choice of methods to acquire knowledge from experts. The most popular method is to conduct unstructured or semi-structured face-to-face interviews with experts. Other methods rely upon observations. Computer programs have also been constructed to automate the acquisition of knowledge from experts. The choice of a particular method, however, can be an arbitrary decision made by the knowledge engineer.

The second problem associated with knowledge engineering is that experts typically do not structure their decision making in any formal way, and they may have great difficulty

recognising and describing the steps of their reasoning. This problem of knowledge transfer has been analysed by a number of researchers. Dixon (1981) discovered that much human activity is not accessible to awareness. Collins (1985) studied knowledge transfer processes among scientists and found that some knowledge might not be accessible to experts because these experts are not aware of the significance of all the steps involved in solving a problem. Bainbridge (in Merry 1985) notes that there is no necessary correlation between verbal reports and mental behaviour. That is, introspective verbal reports of how a problem is resolved may not describe accurately and completely the actual mental activities involved in solving the problem.

The third problem with knowledge engineering is that if methods involving interviews and observations are used, different kinds of interviewer, interviewee and observation biases (Green, Tull and Albaum 1988) can affect the quality of the knowledge acquired from the experts. These biases arise from the social interaction between the knowledge engineer and the experts invited to participate in the construction of the system. Many cognitive defences have been found to impede effective communication (Freud 1914, Green, Tull and Albaum 1988).

3.1.2 Choice of Knowledge Engineering Methods in the Current Study

Three problems arose with the choice of common interview-based methods or observation-based methods in the current study. One was that these methods have never been tried in a problem domain characterised by a high degree of fuzziness. The problem domain of a marketer is overwhelmingly fuzzy. There is often no correct solution to a marketing problem.

Another problem was that marketers participating in the knowledge engineering process came from diverse backgrounds with respect to experience and expertise. It was very likely that knowledge engineers would get a large amount of conflicting knowledge from these expert marketers. In this connection, a technique that is less vulnerable to the psychological problems mentioned above and at the same time allows marketers to come to a certain degree of consensus was required for this study.

Finally, the experts that were involved in this study were scattered about in different locations. Time and cost made frequent group face to face meetings infeasible. A more cost effective method was required. Having taken all these factors into consideration, the Delphi technique appears to be a good technique to use in this research. The technique is described in greater detail in the next section. Additional justifications for adopting the method are also given in the section.

3.2 The Delphi Technique and its Application in Knowledge Engineering

According to Linstone and Turoff (1975), the Delphi technique¹ is a method for structuring a group communication process so that a channel of feedback and an opportunity for the participants to revise their opinions is provided. A high degree of anonymity for the participants is maintained throughout the entire process. This avoids a number of problems in communication arising from face-to-face interactions among the experts. For example, it eliminates the dysfunctional effect in a group communication process when certain group members, who might not have exceptional expertise, dominate the group (Carzo 1963).

The Delphi technique involves a monitor who designs the questionnaire. The questionnaire is then sent to a group of experts. After the questionnaire is returned, the monitor summarizes the results and, based upon the conclusions reached, formulates another questionnaire for the next iteration. Each respondent is then given at least one chance of revising his or her judgments based upon examination of the group response. The respondents may have to go through several iterations until a certain degree of consensus is reached.

¹ The Delphi technique was one of the spinoffs of defense research. In the early 1950's, the US Air-Force sponsored Rand Corporation started the "Project Delphi" study to look into the possibility of obtaining the most reliable consensus of opinions of a group of experts by a series of intensive questionnaires interspersed with controlled opinion feedback.

Brightman and Verhoeven (1986) criticised non-Delphi type problem solving groups for their failure to derive a consensus. They reason that group leaders often do not possess the necessary interpersonal skills nor the understanding of group dynamics to be effective, and group problem solving may encourage the suppression of personal beliefs that may invoke conflict. The Delphi method, according to these authors, may be more effective in achieving a consensus belief.

In a situation where experts hold different views of the reality and where the reasoning mechanism is complicated, there may be personal conflicts (resulting from their different recommendations about the same problem (Linstone and Turoff 1975)). These conflicts would affect the quality of the knowledge extracted from the knowledge engineering process. The Delphi technique avoids this by protecting the individual from bringing up an idea that turns out to be idiotic and results in a loss of face. It also avoids the difficulty of publicly contradicting individuals, and of unwillingness to abandon a position once it is made public.

The Delphi technique is actually a process of negotiation among the experts. In the process of negotiation, experts continually add, verify and revise their perceptions of

reality until a certain degree of consensus is arrived² at. Dalkey, Brown and Cochran (1970) studied the feedbacks in different Delphi sessions. The experiment suggested that the experts were sensitive to the feedbacks. Most were interested in the opinions of the other group members and desirous of moving in the direction of the consensus.

The Delphi technique has been applied in a wide range of areas such as forecasting and future studies with very satisfactory results. Early studies by Turoff (1970), for example, used the Delphi technique on major policy issues and the results were effective. More recent application studies by Clouser (1986), Brancheau and Wetherbe (1987), Trewatha, Hampton, Vaught and Parker (1988), Yong, Keng and Leng (1990), Ray and Sahu (1990) all found the Delphi technique useful in the application areas described in their research reports.

An example of the application of the technique in marketing was the study conducted by Ray (1988). Ray formed a Delphi panel and found that the results were useful for redesigning

² The philosophical basis of the Delphi technique can be viewed from the Hegelian dialectical perspective (Mitroff and Turoff in Linstone and Turoff 1975). The idea of the Hegelian approach is that truth is conflictual. The truth of a system is a complex interplay between a thesis and a counter thesis. These two theses engage each other in a persisting debate over the true nature of the system to draw forth a new plan that synthesizes the thesis and the counter thesis. In a Delphi session, different experts might have different views about the truth of a system. The Delphi sessions form the basis for these different opinions to interact to get to the truth. In the process, individual experts also learn from one another in a protected environment while studying others and the world.

business strategies and refining markets. Gibson and Miller (1990) applied the Delphi technique for regional economic diversification problems and found that the method presented a very useful approach to the analysis of complex, multidimensional problems. Hossein and Khorramshahgol (1990) used the Analytic Delphi Method in location decisions with great success. The Analytic Delphi Method incorporates intangible and tangible factors into the decision making process by integrating mathematical models for multicriteria decision making with the Delphi method.

The Delphi process can be automated by a computer system. Fraser et al (1989), for example, constructed a system called Conflict Resolver (CR) for mediating disputes arising among competing expert systems. The CR uses a Delphi-like mediation to reach a compromise resolution.

This section has discussed the nature of the Delphi technique and its applications. The next section, section 3.3, concentrates on how the technique was actually used in this study.

3.3 The Application of the Delphi Technique in the Current Study

The entire knowledge engineering process was divided into three stages. The first stage, the pilot study, is described in section 3.3.1. During the second stage, fieldwork was conducted to obtain the initial responses of the marketers to a set of questions about how price decisions should be made in the export wool industry. The second stage is described in section 3.3.2. Three Delphi sessions were conducted in the third stage which is described in section 3.3.3. The summarized results were then used for the construction of the actual system.

3.3.1 The Pilot Study

3.3.1.1 Method Used in the Pilot Study

The first stage is the pilot study. The study had three objectives. The first was to collect some background information on the environment of price decision making in the wool industry to enable a better formulation and identification of the system and the problem. The second was to obtain a set of heuristic decision rules involving primary fuzzy variables. The third was to collect basic information to build up the questionnaire for collecting the required price recommendations and the values of membership functions of the linguistic variables in stage two of the study.

Invitation letters were sent to all wool exporters in the

North Island and the letters were followed up by telephone calls. The list of wool exporters was obtained from the 1989 Statistical Handbook published by the New Zealand Wool Board. Fourteen companies were willing to participate in the preliminary personal interviews. The letter of invitation and the questionnaire used in the preliminary interviews can be found in Appendix 1.

The questionnaire in Appendix 1 is an unstructured questionnaire. The questions formed the framework for discussion with the wool marketers. After each question was asked, responses and clarifications were probed until all the necessary information had been obtained.

The questionnaire is divided into five parts. The first part looks into the process managers go through in quoting a price for a transaction. The second part asks the managers to give the important factors they would consider in the price decision making process. How each one of these factors would affect their decisions was asked in the third part. In part four, marketers were asked to suggest the situations under which they would recommend a small positive change in price, a positive change in price, a very positive change in price, a small negative change in price, a negative change in price and a very negative change in price. The last part asks managers under what situations they would not change the price. The responses were used to build up a set of decision rules relating the primary fuzzy variables.

3.3.1.2 Results of the Pilot Study

3.3.1.2.1 Process of Price Decision Making

The method that a manager uses in recommending a price is basically cost-plus pricing. First of all, the costs of the amount of wool to be sold are obtained from the company's accounting records. These cost include the cost of the wool and additional cost items like freight, insurance and so on. Then an average margin of around 2% is added to the total cost. The final quoted price is adjusted up or down depending on basically three fuzzy factors in the decision making environment. These three fuzzy factors are the company's cashflow situation, the market situation and the competitive situation.

3.3.1.2.2 Important Fuzzy Factors in the Decision Making Process

The importance of a decision making variable was assessed on the basis of the total number of times the variable was quoted as important by the manager. The following table shows the important factors and the number of times they were quoted as important:

Table 3.1 Importance of Fuzzy Factors

Factor	Importance
Market Situation	14
Exchange Rate	10
Cashflow Situation	8
Competition	7
Interest Rate	4
Payment Terms	2
Importance of Customer	2
Inflation	1
Company Policy	1

To simplify the construction of the system, only factors that were quoted as important by at least half of the managers were specifically dealt with by the system. Hence, the last five factors in Table 3.1 were not incorporated into the knowledge-based system reported in this thesis. Furthermore, since only companies that hedge against exchange rate fluctuations through a bank were subjects of this study, the system did not need to take care of exchange rate as a decision making variable in an explicit manner. When the bank has taken over all the risks associated with exchange rate fluctuations, the system only needs to include the cost of hedging against currency fluctuations in calculating the total cost of the wool to be sold to a customer in the price decision making process. Thus, only three fuzzy variables were dealt with by the system. These three factors are market situation, competitive situation

and cashflow situation. This is not realistic, but provides a suitable compromise for the prototype described in this research.

3.3.1.2.3 Decision Rules used in the Decision Making Process

A number of decision rules were obtained from the managers. The following two are examples of these rules:

If cashflow is favorable, then price change is small positive.

If cashflow is unfavorable, then price change is small negative.

3.3.2 The Fieldwork for Initial Responses

3.3.2.1 Method

In the second stage of the study, a questionnaire was sent to all the companies listed as wool exporter in the Statistical Handbook 1989 mentioned above. The list includes companies in the North Island as well as the South Island. The total number of companies is 112, after excluding all those who have stopped trading. Companies having the same address were counted as one company. The questionnaire and the covering letter are shown in Appendix 2.

Before the second questionnaire was sent out, it was pretested by asking managers in Palmerston North the questions contained in the questionnaire. Peculiar

responses were noted during the pretest. For example, if certain questions need to be repeated or amplified before they were understood, these questions were simplified or rewritten. Managers were also asked their opinions of the questionnaire so that improvements could be made before it was sent out.

During the survey, nonrespondents were contacted by phone and if necessary, an appointment was made so that a personal interview could be conducted to assist the managers in answering the questionnaire. This saved time and effort in travelling as the managers were widely dispersed throughout the country. The number of usable questionnaires returned was 28, making up a response rate of 25%. Of the 84 companies, 5 did not fill in the questionnaire because they were in financial difficulty, and therefore did not want to be involved in the survey. Another 23 companies refused to participate because they were not involved in wool export, although they were listed as wool exporters in the Statistical Handbook. The remaining 56 nonrespondents either were not interested in the survey or could not participate because of company policy.

Although the response rate was only 25% and only ten managers were finally invited to participate in the Delphi sessions (see section 3.3.2.2.3 below), all the companies were performing well in the industry. Their managers were often consulted on television or appeared in newspapers regarding the strategies wool exporters should take in the

downturn of the market situation. This is important because the system to be constructed is supposed to give quality recommendations to users. To enable quality recommendations to be given, the knowledge stored in the system must come from successful managers.

Most reports on expert systems or knowledge-based systems developed do not mention the number of experts invited. In fact, none of the knowledge-based systems reviewed in section 2.2 specified the number of experts involved in knowledge engineering. After a review of studies involving the application of knowledge-based systems in other business areas, it was found that only two of them reported on the number of experts involved. In the Capital Expert System developed by Texas Instrument for making capital investment decisions and reported in AI Interactions (1986), only one expert was involved. Another developed by Southern California Edison for predicting summer electricity consumption in the Southern California area (Gibson and Cortese 1989) used only one expert. As far as evaluation is concerned, developers of both systems reported substantial savings in their operating expenses after the implementation of the system. The details of how the evaluations were done were not given.

It appears that some systems developed in non-business areas also involved few experts. A good example is the system called REVA (Rotating Equipment Vibration Advisor) developed by Stones and Webster Incorporated (Meyer and Curley 1989)

for the diagnosis, analysis and prediction of equipment failure. Only two experts were involved in the construction of the system. No evaluation was done on the effectiveness of the system.

So far, no study has been done on the optimal number of experts that should be invited to participate in the construction of knowledge-based systems. The feeling obtained from reviewing previous studies is that the controlling factor is availability. Most systems only involve experts who are readily available. In this connection, more research is required.

Coming back to the fieldwork, the questionnaire used was divided into three sections. The first section required the managers to recommend the percentage price change in response to each of the given scenarios. The data obtained in this section were used for testing the effectiveness of the system in making price recommendations.

Thirty scenarios were presented. Each scenario was made up of a combination of different levels of the three most important fuzzy factors in the decision making process. There are 729 possible scenarios because each fuzzy factor is made up of 9 levels. Of these thirty scenarios, six involved one primary fuzzy variable only. Two involved very extreme scenarios. The reason why only two extreme scenarios were included in the set of scenarios was that these extreme scenarios are rare in practice. The other

twenty-two scenarios were picked up randomly from the scenarios involving at least two fuzzy variables, none of them had extreme levels. The following example shows a scenario without extreme levels:

"Very favorable market situation" and "favorable cashflow situation" and "strong competition"

A typical example of a scenario involving an extreme level is as below:

"Very very favorable market situation" and "neither favorable nor unfavorable cashflow situation" and "neither strong nor weak competition"

In response to the scenarios, managers were asked to recommend a percentage price increase/decrease.

The second section of the questionnaire contained questions for identifying the membership functions for the primary fuzzy variables used in the decision rules obtained in the pilot study. One of the findings in the pilot study was that when managers were asked to describe how they determined the outlook of the market, they responded by saying that they first looked at a graphical display of the movements of the market indicators in the past twelve months, then they extended the graph into the future. When the extended portion of the graph is upward-sloping, the market situation is going to be favorable. Otherwise, it is unfavorable. Hence, two different sets of figures, each of which contained seven diagrams depicting seven different

possible slope positions of the extended part of the graph, were used to obtain the membership functions of favorable market situation and unfavorable market situation. The diagrams can be found in Appendix 2.

The pilot study also showed that the way these managers evaluated how favorable or unfavorable the cashflow situation was in the same way that they evaluated the market situation. Hence, another two sets of diagrams similar to those used in obtaining the membership functions for market situations were used for obtaining the membership functions for the cashflow situations. The diagrams can be found in Appendix 2.

Based on the pilot survey, the difference between a company's price and its competitor's price was used by the manager as the basis for evaluating the degree of competition in the market. Hence, for fuzzy subsets involving "competitive situation", the degree of competition was assessed in the following manner. First, the difference between a company's price and its strongest competitor's price was computed. Then the difference was expressed as a percentage of the company's price. The percentage was used as a measure of the degree of competition. The range of these percentages is from -15% to 15%. These percentages were used for the membership functions of "favorable competitive situation" and "unfavorable competitive situation".

The third section collected information about the number of years of marketing experience of the manager answering the questionnaire and whether the company hedges against currency fluctuations. The purpose of the second question in this section was to select experienced managers to participate in the Delphi sessions. Only experienced managers were regarded as experts.

The first question of section three was concerned with whether or not the company hedged against exchange rate risk through a financial institution. It was decided to exclude companies that speculate in the foreign exchange market because it was thought that the pricing decisions would be made in a different way by those who hedge than by those who do not, and it is thought better to have a homogeneous group of experts than a heterogeneous one. O'Brien (1990) and Newquist (1986) suggested that one cardinal principle in the construction of a knowledge-based system is that the system should aim at supporting decision making in a small and well-defined problem area. The problems faced by a group of homogeneous marketers should be more well-defined and narrower in scope than a heterogeneous group of marketers, and a system aimed at supporting the former group should have a higher probability of success.

3.3.2.2 Results of the Fieldwork

3.3.2.2.1 Managers' Recommendations Given the Scenarios

Each manager who participated in the study recommended price changes in response to the given scenarios. An example of a set of such recommendations can be found under column one on page 140 of Appendix 3.

3.3.2.2.2 Initial Membership Functions

The managers were also asked to give the membership functions for the fuzzy linguistic variables used in the decision rules obtained from the pilot study. The following is an example of the membership function for small positive change in price given by one of the managers:

(0.90/1%, 0.87/2%, 0.65/3%, 0.40/4%, 0.30/5%, 0.27/6%,
0.25/7%, 0.23/8%, 0.22/9%, 0.20/10%, 0.20/11%,
0.18/12%, 0.18/13%, 0.18/14%, 0.18/15%)

3.3.2.2.3 Selection of Managers for Delphi Sessions

Responses to the first two questions in section three of the second questionnaire were used to select managers to participate in the Delphi sessions. The participants were screened and eventually only ten managers were qualified to participate in the Delphi sessions. Two criteria were used for selecting these ten managers. First, they must have at least ten years of working experience in the industry. Second, the companies they serve must always hedge against currency fluctuations in the foreign exchange market. In

fact, all these managers were very experienced marketers. The average number of working experience of these managers was over 26 years. The least experienced one had twelve years of working experience, the most experienced one thirty-six years.

3.3.3 The Delphi Sessions

3.3.3.1 Method

Three Delphi sessions were held to allow the experts participating in the study to refine their judgments. Three sessions were organised because Linstone and Turoff (1975), Erffmeyer, Erffmeyer and Lane (1986) found that a point of diminishing returns is reached after three or four rounds. They also found that excessive repetition upset participants. Most commonly, three rounds proved to be sufficient because ninety-nine percents of the change in opinions occur in the first three rounds (Brockhoff in Turoff 1975).

During each iteration, a summary of the responses of the previous session were sent to each manager together with the manager's own responses to the questionnaire in the previous iteration. Appendix 3 shows an example of the questionnaire sent to the managers in an iteration. A covering letter was sent together with the questionnaire. The covering letter invited the managers to revise their rules and membership functions if necessary after they had read the group responses.

Since the decision rules involving the primary fuzzy variables form the bases for drawing inferences in the system, these rules were also included in the three Delphi sessions to ensure that the managers could come to a degree of consensus on them.

The three Delphi sessions took place over a three-month period from May to August 1990. The following table shows the response rates of the three Delphi sessions:

Table 3.2 Response Rates of the Delphi Sessions

	Number of Questionnaires Sent	Number of Questionnaires Returned
Round 1	10	10
Round 2	10	7
Round 3	10	4

A high attrition rate is common for the last one or two Delphi sessions in using the Delphi technique. For this particular study the problem should not threaten the reliability of the consensus obtained because those who dropped out made very few revisions of opinions in the Delphi session prior to their dropping out. In other words, it is safe to assume that they did not want to change their opinions any further after the last time they responded, and that the average of these last responses from the ten managers could be taken as the consensus required.

3.3.3.2 Results of the Delphi Study

3.3.3.2.1 Decision Rules

A consensus was obtained on the following six decision rules. Each rule describes the conditional relationship between two primary fuzzy variables.

Rule 1: If cashflow is favorable, then price change is small positive.

Rule 2: If cashflow is unfavorable, then price change is small negative.

Rule 3: If market outlook is favorable, then price change is small positive.

Rule 4: If market outlook is unfavorable, then price change is small negative.

Rule 5: If competition is strong, then price change is small negative.

Rule 6: If competition is weak, then price change is small positive.

3.3.3.2.2 Membership Functions

Consensus were also obtained for the membership functions.

The membership function for favorable cashflow is:

(0.25/diag1, 0.67/diag2, 0.85/diag3, 0.80/diag4,
0.70/diag5, 0.67/diag6, 0.66/diag7)

Diag1 to diag7 represent seven possible slope positions depicting seven possible favorable cashflow situations as described in section 3.3.2.1. These diagrams can be found in Appendix 2.

The membership function for unfavorable cashflow is:

(0.27/diag1', 0.50/diag2', 0.87/diag3', 0.85/diag4',
0.81/diag5', 0.80/diag6', 0.77/diag7')

Again, diag1' to diag7' represent seven different slope positions representing seven different levels of unfavorable cashflow situations. These seven diagrams can also be found in Appendix 2.

The membership function for favorable market outlook is:

(0.38/fig1, 0.57/fig2, 0.74/fig3, 0.65/fig4, 0.65/fig5,
0.67/fig6, 0.67/fig7)

The membership function for unfavorable market outlook is:

(0.48/fig1', 0.76/fig2', 0.81/fig3', 0.75/fig4',
0.64/fig5', 0.65/fig6', 0.65/fig7')

Fig1 to fig7 and fig1' to fig7' can all be found in Appendix 2.

The membership function for strong competition is:

(0.43/1%, 0.51/2%, 0.60/3%, 0.61/4%, 0.60/5%, 0.67/6%,
0.68/7%, 0.70/8%, 0.71/9%, 0.73/10%, 0.75/11%,
0.77/12%, 0.79/13%, 0.80/14%, 0.80/15%)

The membership function for weak competition is:

(0.75/1%, 0.57/2%, 0.42/3%, 0.24/4%, 0.36/5%, 0.39/6%,
0.37/7%, 0.37/8%, 0.38/9%, 0.36/10%, 0.33/11%,
0.32/12%, 0.31/13%, 0.31/14%, 0.30/15%)

The membership function for small positive change in price is:

(0.95/1%, 0.89/2%, 0.45/3%, 0.38/4%, 0.30/5%, 0.29/6%,
0.25/7%, 0.22/8%, 0.21/9%, 0.20/10%, 0.20/11%,
0.20/12%, 0.20/13%, 0.20/14%, 0.20/15%)

The membership function for small negative change in price is:

(0.95/1%, 0.91/2%, 0.54/3%, 0.31/4%, 0.27/5%, 0.28/6%,
0.28/7%, 0.25/8%, 0.24/9%, 0.24/10%, 0.23/11%,
0.23/12%, 0.22/13%, 0.21/14%, 0.20/15%)

3.3.3.2.3 Consensus Decisions

Table 3.3 summarizes the results of the three Delphi sessions on the decisions given by the managers in response to the thirty scenarios described in section 3.3.2.1.

Table 3.3 Consensus Recommendations to Problem Scenarios

Market outlook	Cashflow situation	Competition situation	Group Average
Unfavorable	Favorable	Very weak	-0.65%
Very very favorable	Neither favorable nor unfavorable	Neither strong nor weak	2.65%
Very favorable	Very favorable	Very strong	2.3%
Very favorable	Favorable	Strong	2.25%
Very favorable	Very favorable	Strong	2.35%
Very favorable	Unfavorable	Weak	2.15%
Very favorable	Unfavorable	Very strong	1.5%
Very favorable	Unfavorable	Strong	1.4%
Very favorable	Very unfavorable	Very strong	1.1%
Very very unfavorable	Neither favorable nor unfavorable	Neither strong nor weak	-1.9%
Favorable	Very favorable	Strong	1.5%
Favorable	Very favorable	Weak	1.15%
Favorable	Favorable	Strong	0.9%
Favorable	Very unfavorable	Very strong	0.1%
Favorable	Very unfavorable	Weak	0.4%
Favorable	Neither favorable nor unfavorable	Neither strong nor weak	0.5%
Unfavorable	Neither favorable nor unfavorable	Neither strong nor weak	-1.1%
Unfavorable	Very favorable	Very strong	-1.35%
Unfavorable	Favorable	Very strong	-1.15%
Unfavorable	Favorable	Strong	-0.95%
Unfavorable	Unfavorable	Strong	-1.78%
Neither favorable nor unfavorable	Favorable	Neither strong nor weak	0.55%

Market outlook	Cashflow situation	Competition situation	Group Average
Neither favorable nor unfavorable	Unfavorable	Neither strong nor weak	-0.2%
Unfavorable	Very unfavorable	Very strong	-2.05%
Neither favorable nor unfavorable	Neither favorable nor unfavorable	Strong	-0.2%
Neither favorable nor unfavorable	Neither favorable nor unfavorable	Weak	-0.1%
Very unfavorable	Very favorable	Very strong	-1.75%
Very unfavorable	Favorable	Strong	-1.45%
Very unfavorable	Very unfavorable	Very strong	-3%
Very unfavorable	Very unfavorable	Weak	-3.03%

3.4 System Evaluation

The objective of this research was to build a computer-based wool price decision making system that performs at least as well in the decision making task as individual experts in the industry.

Basic to any expert system is the premise that experts make good recommendations, and if the manner in which they arrive at those recommendations is modelled and incorporated in a system, then the system will also make good recommendations, providing the modelling is adequate. If, under the same conditions, the system makes the same recommendations as an expert then the system can be regarded as emulating an expert and the modelling has achieved its purpose. The study does one step further, it attempts to emulate the recommendations of a consensus of experts, as achieved by the Delphi method, supposing the consensus recommendation to be better than any individual expert recommendation.

Individual experts make good recommendations, that is why they are regarded as experts, but in any given situation, one expert's recommendation will differ from another's, and the same applies to the manner in which they make their recommendations. That is, within a group of experts there will be a variation in the manner in which they make their recommendations, and in any particular situation there will be a variation in the recommendations they make. The consensus of the manner in which they make their recommendations is thus regarded as a better way to make a

recommendation than that of any particular individual, and the consensus recommendation is regarded as a better recommendation than that of any particular individual.

The system described in this thesis models the consensus of experts in the manner in which they make their recommendations, and the system is evaluated by comparing the decision making performance of the system with that of the experts, when each recommendation is measured against the consensus recommendation. If the system more closely emulates the consensus recommendation than most of the experts then the system is regarded as performing at least as well as individual experts.

Thus the Delphi method has to be employed on three occasions. Twice in the system building phase; to achieve consensus regarding the membership functions and to achieve consensus regarding the decision rules, and once in the evaluation stage; to achieve consensus regarding the recommendations. The consensus recommendations provide a standard against which to measure both the individual expert's recommendations and the system's recommendations.

Applying the above method to the construction and evaluation of the knowledge-based system reported in this study, the first two kinds of consensus - the consensus on the membership functions and the consensus on the decision rules - were stored in the system and used to arrive at recommendations which were then compared with the consensus

on the recommendations. In other words, the evaluation involves a set of consensus recommendations and ten sets of individual recommendations, all of which were produced by the marketers in response to the thirty problem scenarios described in section 3.3.2.1, and another set of recommendations produced by the computer system given the same set of problem scenarios.

Having obtained all the recommendations described above, the correlation coefficients between the individual managers' recommendations and the consensus recommendations were computed. This produced ten correlation coefficients. Meanwhile, the correlation coefficient between the system's recommendations and the consensus recommendations was also obtained. By comparing the system-consensus coefficient with the individual manager-consensus coefficients, a judgment could be made as to how effective the system was in generating price recommendations, and whether or not the system was performing at least as well as the experts in the study. If the system-consensus coefficient was larger than most of the manager-consensus coefficients, then the system could be judged as performing at least as well as a good manager. On the other hand, if the system-consensus correlation coefficient was smaller than most of the other coefficients, then the system could be said to be performing worse than a good manager.

CHAPTER 4

SYSTEM SPECIFICATION

This chapter is divided into four sections. Section 4.1 describes the basic philosophy for building up the system. Section 4.2 describes the knowledge in the knowledge base. Section 4.3 then goes on to describe how the knowledge stored in the system is used and how the control mechanism draws inferences. The entire control procedure is represented by a flowchart in this section. Section 4.4 introduces Prolog, the programming language used for the construction of the system, and the reasons behind the choice of Prolog for this project.

4.1 Philosophy for the Construction of TZ

TZ is the name given to the system reported in this thesis. The basic philosophy for the construction of the system is evolutionary prototyping (Tate 1990).

A prototype is not the real thing. It imitates the real thing only in certain essential aspects. A prototype enables a crude form of the actual system to be developed and tested. If the prototype meets the requirement, then the actual system is developed. Otherwise, the prototype is revised until it meets the requirement. In evolutionary prototyping, once the prototype is considered satisfactory, the prototype forms the basis for the construction of the application system.

The reason for adopting the above approach was that the attempt to develop a knowledge-based system for price

decision making was the first of its kind. There is a high degree of risk and uncertainty in developing such an application system. There was only a rough idea at the beginning that the system would be used for generating price recommendations. The specification of the different components in this system became more explicit after the three Delphi sessions. Nevertheless, details of the control procedures and how these different components are related were still somewhat unclear. The process of building up TZ was full of trial and error. Some components in the system were rewritten several times before they were accepted. The experience obtained after building up TZ is that it is seldom possible to get a software system right at the first time and that a prototype approach was found very useful in the development of TZ.

Another advantage of using the evolutionary approach was that it helped to restrict the scope of the initial system. The original version of TZ only performed fuzzy inferences involving the primary fuzzy subsets. Once the performance of this small system was found satisfactory, other subsystems, such as the one for handling user-system dialogues, were appended to the main program incrementally. As a result, the system became gradually more and more comprehensive. Thus, the evolutionary approach enabled modules to be built up one after another and judged (by the user and the system developer) for their quality before they were incorporated into the system. This is in

accordance with what Lehman (1989) proposed for software development. Lehman suggested that software had to evolve, undergo continuous adaptation and change. It had to be treated as an ever-to-be-adapted organism, not as a to-be-produced-once artifact.

Finally, the evolutionary approach is also consistent with the idea that TZ will one day be upgraded to a system that can support all the marketing functions in an organization. Modules can be added one after another in an evolutionary manner to the main system until it can provide functional support for all aspects of marketing decision making in the export wool industry.

4.2 Knowledge Stored in TZ

The basic knowledge stored in TZ is the decision rules extracted from the managers after the process of knowledge engineering. Each rule describes the conditional relationship between two primary fuzzy variables. These decision rules have been reported in section 3.3.3.2.1.

There are two rules for each fuzzy variable used by the system. In the future upgrading of the system, more rules can be added when more fuzzy variables are incorporated into the decision making process.

Besides the above decision rules, there are five other kinds of knowledge stored in the system. They are described below:

4.2.1 Membership Functions of Primary Fuzzy Variables

The membership functions of the primary fuzzy variables and the associated universes of discourse can be found in section 3.3.3.2.2.

4.2.2 Relational Matrices

Each of the six primary decision rules is represented in the form of a relational matrix in the knowledge base. The formula for obtaining the relational matrices can be found in section 2.3.1.4. The relational matrix for the first decision rule is given below:

Figure 4.1 Relational Matrix for Rule 1

		Favorable Cashflow						
		.25	.67	.85	.8	.7	.67	.66
small positive price change	.95	.95	.95	.95	.95	.95	.95	.95
	.89	.89	.89	.89	.89	.89	.89	.89
	.45	.75	.45	.45	.45	.45	.45	.45
	.38	.75	.38	.38	.38	.38	.38	.38
	.3	.75	.33	.3	.3	.3	.33	.34
	.29	.75	.33	.29	.29	.3	.33	.34
	.25	.75	.33	.25	.25	.3	.33	.34
	.22	.75	.33	.22	.22	.3	.33	.34
	.21	.75	.33	.21	.21	.3	.33	.34
	.2	.75	.33	.2	.2	.3	.33	.34
	.2	.75	.33	.2	.2	.3	.33	.34
	.2	.75	.33	.2	.2	.3	.33	.34
	.2	.75	.33	.2	.2	.3	.33	.34
	.2	.75	.33	.2	.2	.3	.33	.34

The relational matrices for the other five decision rules can be found in Appendix 4.

4.2.3 Formulae for Hedging Functions

A number of formulae for the hedging functions used in the system were also stored in the knowledge base. These formulae were the ones suggested by Zadeh (1976a). These formulae are as follows:

- 1 Very very favorable = favorable⁴
- 2 Very favorable = favorable²
- 3 Slightly favorable = favorable^{0.75}
- 4 Very very unfavorable = unfavorable⁴
- 5 Very unfavorable = unfavorable²
- 6 Slightly unfavorable = unfavorable^{0.75}

- 7 Very very strong = strong⁴
- 8 Very strong = strong²
- 9 Slightly strong = strong^{0.75}
- 10 Very very weak = weak⁴
- 11 Very weak = weak²
- 12 Slightly weak = weak^{0.75}

4.2.4 Relatively Constant Facts

Two sets of facts remain relatively stable in the price decision making process. The first one is the mark-up used in the cost-plus pricing procedure. The average mark-up was found to be 2.1%.

The second set of facts stored in the knowledge base is the weights given to individual fuzzy variables. These weights are needed because the system makes a recommended price change for each fuzzy input. Since there are three fuzzy inputs, there are three such recommendations. Weights must therefore be assigned to these three recommended price changes so that a single recommendation can be output by the system.

The weights were obtained by piecewise regression. The data used were those obtained by section one of the second questionnaire. To save the number of degrees of freedom, the assumption was made that each input variable has only five levels. Also, for simplicity, the assumptions that the effects of the variables are additive and that there is no interaction between the different levels of the input

variables were made.

Four dummy variables were required for each one of the fuzzy variables. The standardised beta coefficients of the dummy variables resulting from the analysis could be interpreted as the importance of the corresponding levels of the input variables. The weight assigned to each variable is then the average of the standardised beta coefficients associated with the variable. The weight calculated for the first variable, cashflow, was 0.86. For degree of competition, the weight was found to be 0.28. For market outlook, the weight was 2.26. These weights were rescaled so that they summed to one. As a result, the weights were as follows:

cashflow`	0.253
competition	0.082
market outlook	0.665

4.2.5 Accounting Data

An accounting subsystem was also constructed which enables the main program to draw accounting information from the subsystem in the process of making a price recommendation for a user.

Accounting data that was used in the price recommendation process include the up-to-date operating expenses figures, the sales figure and the stock figure. In the sample run of the system in section 5.1, all the figures used were made-up figures for the purpose of showing how a price

recommendation was generated by the system. The role played by these data in the price recommendation process can be found in the system specification in section 4.3.2.

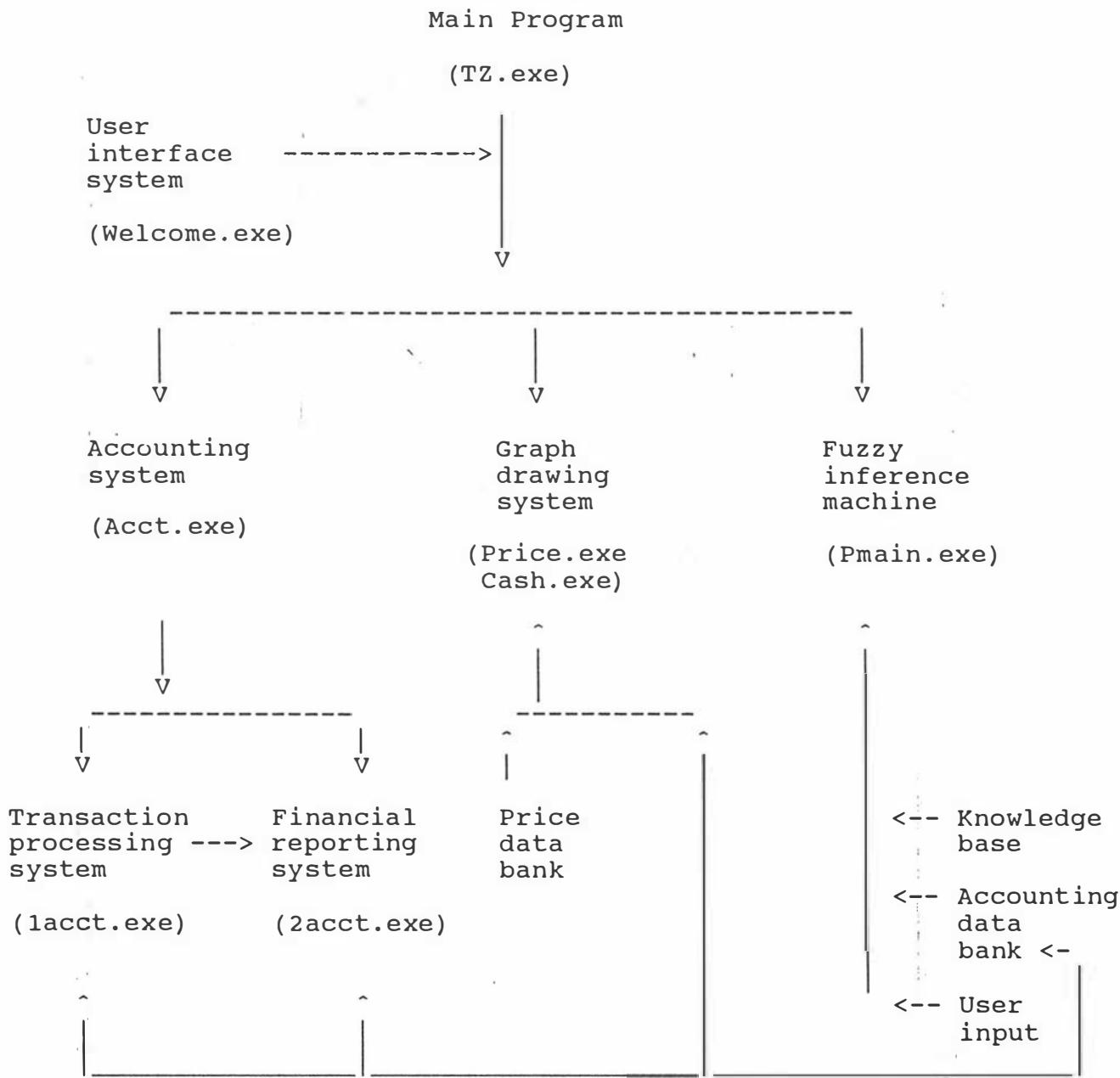
Since the system was evaluated by comparing the percentage price change recommended by the system and the percentage price change recommended by the individual managers in response to the set of problem scenarios in the second questionnaire, the use of make-up figures did not affect the result of the evaluation. In fact, these make-up figures can be replaced easily by actual and genuine data when the system is put into use by a company.

4.3 Structure and Inference Mechanism in TZ

4.3.1 Components of TZ

The following figure shows the main components in the system:

Figure 4.2 Structure of TZ



The top level main program drives the various subsystems in the knowledge-based system. All the "exe" programs shown inside the brackets are stand-alone executable codes of the source programs written in Prolog. The source code for these programs can be found in Appendix 5. The documentation is contained within the program.

The main program controls the execution of four subsystems. These four subsystems are the user interface subsystem, the accounting subsystem, the graph drawing system and the fuzzy inference machine.

Of the four subsystems, the most important one is the fuzzy inference machine. The fuzzy inference machine applies fuzzy logical operators to the knowledge stored in the knowledge base, and accounting data retrieved from the accounting data bank, and information supplied by the user to come to a price recommendation. The inference process and the knowledge involved in the process are described in section 4.3.2 below.

The second subsystem under the main module is the graph drawing system. This subsystem uses the data from the price data bank to produce a graphical display of the movement of wool prices in the previous twelve months. It also draws a similar diagram for the cashflow movements in the previous twelve months.

The third subsystem is the accounting information system. The accounting information system is further divided into

two smaller subsystems. The first one is the transaction processing system which accepts input accounting data and updates the balances in the books. The second subsystem, the financial reporting system, outputs financial statements.

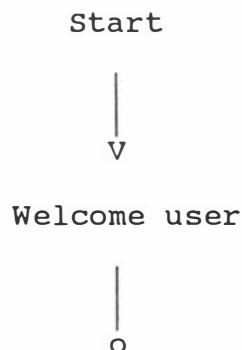
Finally, the user interface subsystem is responsible for providing "help" facilities to make it easy for the user to use the system.

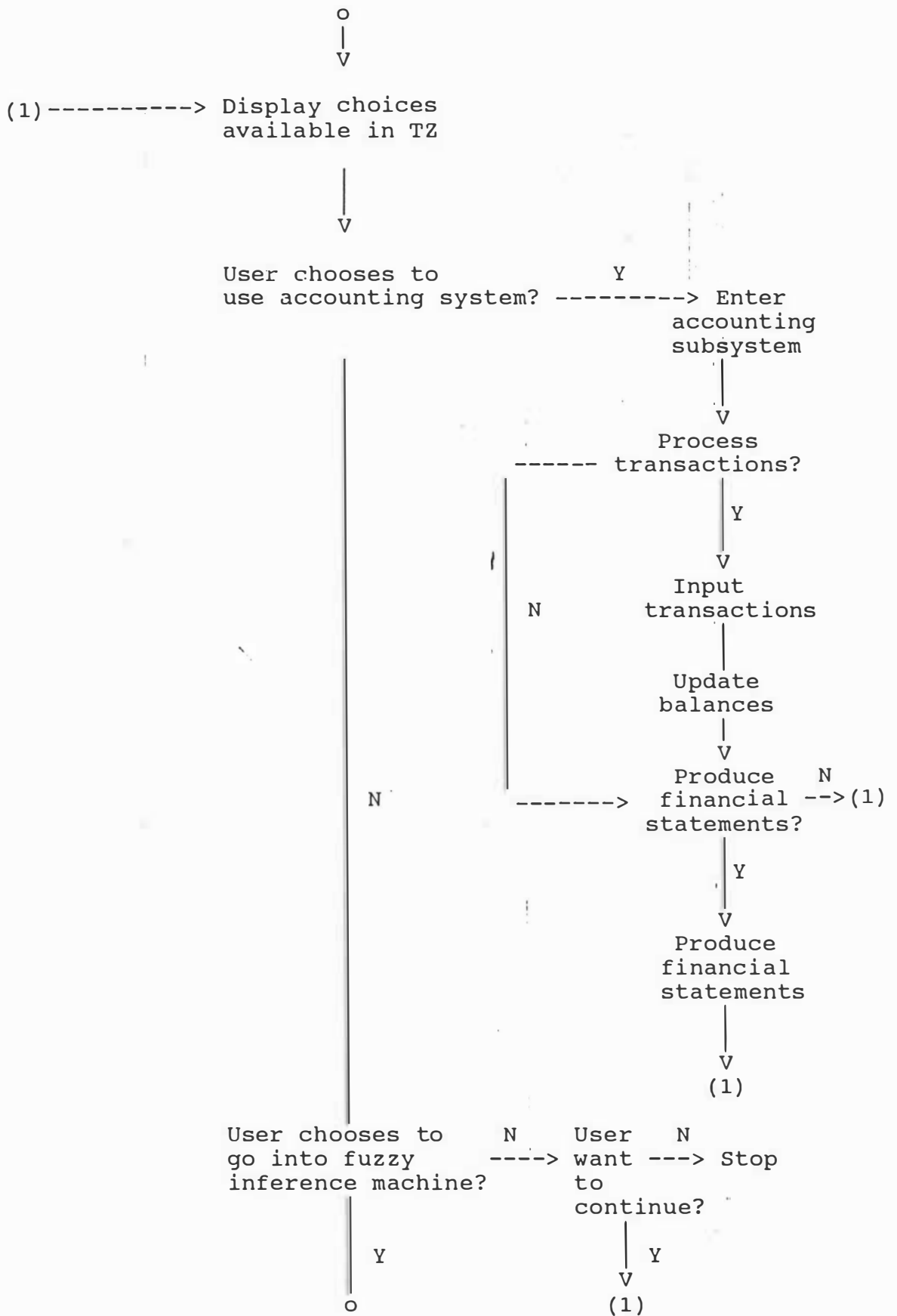
Having introduced briefly the various components of the system, the discussion below will focus on how the main program and the fuzzy inference machine recommend a price to a user and the role played by the various subsystems in this process of price recommendation.

4.3.2 The Inference Mechanism

The inference mechanism in TZ can be represented by the following flow diagram:

Figure 4.3 Flow Diagram Showing the Inference Mechanism in TZ







Display graph showing
movement of market
indicators in
the past twelve months
using data from the
data bank



User input of his/her
judgment of market outlook



Display graph showing
cashflow movement in
past twelve months



User input his/her perception
of cashflow situation



User input his/her judgment
of competitive situation



Apply hedge functions to
user's inputs



o
|
v

Apply max-min composition
rule of inference

|
v

Calculate price change due
to input of market outlook

|
v

Calculate price change due
to input of cashflow situation

|
v

Calculate price change due to
input of competitive situation

|
v

Calculate weighed average price
based on the above three price
changes

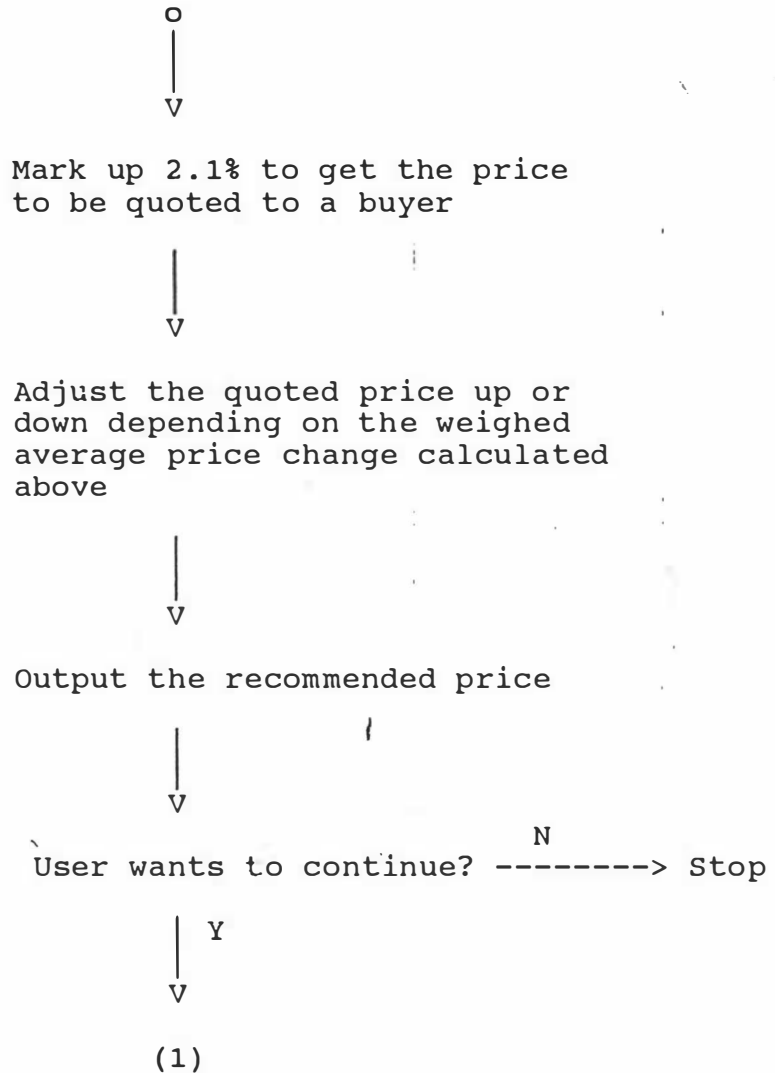
|
v

Obtain from the accounting
subsystem various cost data

|
v

Calculate the total cost of
wool to be sold

|
o



As mentioned above, the main program is the controlling top module of the system. When the system is executed, the main program drives the user interface subsystem. This subsystem presents the first menu and welcomes the user. In addition to providing a brief introduction to the system, help functions are provided to make it easy for the user to use the system.

After the welcoming windows have been presented, the system invites the user to choose between the accounting system and the knowledge-based system. If the knowledge-based system

is chosen, the main program outputs a general description of the nature of the knowledge-based system. Then the user is presented with a graphical display of the price movements of wool in the previous twelve months. The graphical display is produced by one of the programs in the graph drawing subsystem, price.exe. The display is produced to simulate an actual step a manager goes through in making a recommendation. One of the results of the preliminary study (see section 3.3.1) is that a typical manager looks at a graphical display of previous price movements to form an opinion on how favorable the market is. The opinion affects the extent to which the user adjusts the price that is quoted to a buyer.

After the graphical display, the user's opinion of how favorable the market situation is is solicited. The input is in the form of vague and fuzzy linguistic values.

The next screen shows a graphical display of the cashflow situation of the company in the previous twelve months. The display is produced by cash.exe in the graph drawing system. Again, the purpose of drawing the cash flow diagram is to simulate an actual step that a manager goes through in the decision making process, as described by the managers in the preliminary study. After the diagram has been displayed, the user's judgment of the cashflow situation of the company is inputted into the system.

The last piece of information required is the user's

perception of the competitive situation. This piece of information is obtained by the main program. Again, fuzzy linguistic inputs are allowed.

After all the above information has been obtained, the main program passes control over to the fuzzy inference machine.

The fuzzy inference machine performs a set of fuzzy operations on the inputs to produce a price recommendation. The inference machine applies the max-min composition rule of inference described in section 2.3.1.5 to determine the impact of each fuzzy factor on the price to be quoted to a buyer. The results of applying the rule are three price changes due to the three fuzzy factors input by a user. These three price changes are weighed using the constants in section 4.2.4 to determine the overall price adjustment to be made to the price quotation.

The main program then asks for the amount of wool to be sold to a buyer. The cost of the wool and the various overhead figures are then obtained from the accounting data bank so that the overall cost figure can be calculated. A mark-up of 2.1% is then applied to the overall cost figure to get a preliminary price quote for the shipment. The final price recommended will be the preliminary price quote adjusted by the percentage price change produced by the inference machine.

4.4 Prolog: The Programming Language Used in TZ

This section is devoted to the discussion of how the knowledge is represented and how the inference mechanisms are implemented in Prolog. This section is divided into four parts. Section 4.4.1 introduces Prolog and describes the nature of the language. Section 4.4.2 describes how the knowledge is represented in Prolog. Section 4.4.3 describes how the inference mechanism is implemented in Prolog. Section 4.4.4 justifies the choice of Prolog as the programming language for the system.

4.4.1 Introduction to Prolog

Prolog is a language developed specifically for logic programming. Prolog is a logic dedicated language and it can be described from the declarative and procedural perspective.

A program written in a procedural language provides a sequence of operations which controls the execution of the job to be carried out. The order of the sequence of operations is important.

A program written in a declarative language consists of a collection of logic statements about the problem in some arbitrary order. It is the logic contained in the statement rather than the order which is important.

As a simple illustration, consider the statement "P if Q, R" in Prolog. From the declarative viewpoint, the statement

can be read as "P is true if Q and R are true". From a procedural viewpoint, it can be interpreted as: to solve P, first solve the subproblem Q and then the subproblem R.

Most traditional programming languages are procedural languages. Ordering of program statements is important in procedural languages. Using the above example, subproblem Q must be solved first before the subproblem R is solved if the statement "P if Q, R" is implemented in a procedural language. One of the goals of logic programming is to develop a language that is free from the requirement that program statements must be executed one at a time and in the predefined order.

Prolog is basically a declarative type of language. However, this statement must be made with reservation because ordering is sometimes important in Prolog.

4.4.2 Knowledge Representation in Prolog

In Prolog, all knowledge is represented in the form of Horn clauses. A Horn clause is a special type of clause. A clause is a conditional statement with a condition part and a conclusion part. The following is an example of a clause:

Grandmother(X,Y) or Grandfather(X,Y) if Parent(X,Y)
and Parent(Y,Z).

The above clause says that X is the grandmother or grandfather of Y if X is a parent of Y and Y is a parent of Z.

If a clause is made to contain only one conclusion, a Horn clause is obtained (Horn 1951). Any problem that can be expressed by the knowledge representation formalism discussed in section 2.1.1 can be reexpressed by Horn clauses. An example of a typical Horn clause is as follow:

Grandparent(X,Z) if Parent(X,Y) and Parent(Y,Z).

The Horn clause says that X is a grandparent of Z if X is a parent of Y and Y is a parent of Z. "Grandparent(X,Z)" is known as the head of the Horn clause and "Parent(X,Y) and Parent(Y,Z)" is the body of the Horn clause. There are three predicates in this clause: Grandparent(X,Z), Parent(X,Y) and Parent(Y,Z). A predicate describes the relationship between two objects. For example, the predicate Grandparent(X,Y) says that the relationship between X and Y is that X is a grandparent of Y.

Horn clauses can be represented directly by Prolog. For example, the above clause is represented by Prolog as:

grandparent(X,Z) if parent(X,Y), parent(Y,Z).

As seen from the example, the only things necessary to do to represent the clause in a form that can be interpreted by the Prolog compiler are to replace "and" by a comma, and to change the initial letters of the predicate names to lower case letters.

4.4.3 Inference Procedures in Prolog

The derivation of a conclusion is very convenient involving Horn clauses. The basic inference mechanism is backward deduction as described in section 2.1.2.1. An example that shows the inference mechanism is given below. Assume that the knowledge base contains the following clauses:

```
grandparent(X,Z) if parent(X,Y), parent(Y,Z). ..... 4.1
parent(X,Y) if mother(X,Y). ..... 4.2
parent(P,Q) if father(P,Q). ..... 4.3
mother(eva, john). ..... 4.4
father(peter, eva). ..... 4.5
```

Clause 4.1 has been explained above. Clause 4.2 means that X is a parent of Y if X is a mother of Y. Clause 4.3 says that P is a parent of Q if P is a father of Q. Clause 4.4 is a fact. It says that Eva is the mother of John (in the Horn clause given above, Eva and John must be in small letters because Prolog interprets every identifier that begins with a capital letter as a variable). The last clause asserts that Peter is the father of Eva.

Suppose now it is necessary to determine whether or not Peter is a grandparent of John. In Prolog, this query is made by keying into the computer "grandparent(peter,john)" in response to the "goal:" prompt. After receiving the query, Prolog selects the first clause that matches the

pattern of the input query, which in this case is `grandparent(peter,john)`. The first clause that matches the input predicate is clause 4.1 in the knowledge base. The result of pattern matching is that the X variable in the first clause is instantiated to peter and the Z variable is instantiated to john. Hence, the first clause becomes:

```
grandparent(peter,john) if parent(peter,Y), parent(Y,john).
```

..... 4.6

The clause asserts that Peter is a grandparent of John if Peter is a parent of Y and Y is a parent of John. Hence, to determine if the predicate "`grandparent(peter,john)`" is true or not, Prolog has to determine whether or not `parent(peter,Y)` is true and whether or not `parent(Y,john)` is true.

To determine if "`parent(peter,Y)`" is true or not, Prolog searches again in the knowledge base to see if there is any Horn clause with a head that matches "`parent(peter,Y)`". The first clause that matches the predicate is clause 4.2. After instantiation, clause 4.2 becomes:

```
parent(peter,Y) if mother(peter,Y).
```

If `mother(peter,Y)` is true, then `parent(peter,Y)` is true. To do this, Prolog checks whether or not `mother(peter,Y)` is true. There is no clause in the knowledge base that matches `mother(peter,Y)` because the only clause with the mother predicate is `mother(eva,john)` and this predicate does not match `mother(peter, Y)` no matter how Y is instantiated. As

a result, the clause "parent(peter,Y) if mother(peter,Y)" fails.

Although Prolog fails to prove that parent(peter,Y) is true based on clause 4.2, Prolog will still look through the knowledge base to see if there are other clauses that are similar to 4.2 to enable it to deduce that parent(peter,Y) is true. Clause 4.3 is the one that will be considered next. After instantiation (P is instantiated to peter and Q is instantiated to Y), clause 4.3 becomes:

```
parent(peter,Y) if father(peter,Y).
```

This time the body of the Horn clause matches clause 4.5 in the knowledge base - father(peter,eva) if Y is instantiated to eva. Having successfully obtained a matching predicate, Y is instantiated to eva and the conclusion that parent(peter,eva) is true is inferred.

After Y is instantiated to eva, clause 4.6 becomes:

```
grandparent(peter,john) if parent(peter,eva),  
                           parent(eva,john).      .... 4.7
```

parent(peter,eva) has been proved to be correct. If parent(eva,john) is also true, then grandparent(peter,john) can be inferred as true as well. To prove parent(eva,john), Prolog searches through the knowledge base and try to do pattern matching again. The first clause that matches parent(eva,john) is 4.2. After instantiation, 4.2 becomes:

parent(eva,john) if mother(eva,john). 4.8

That is, eva is a parent of john if eva is the mother of john. Clause 4.5 matches the body of 4.8. Thus, mother(eva,john) is true and so is parent(eva,john).

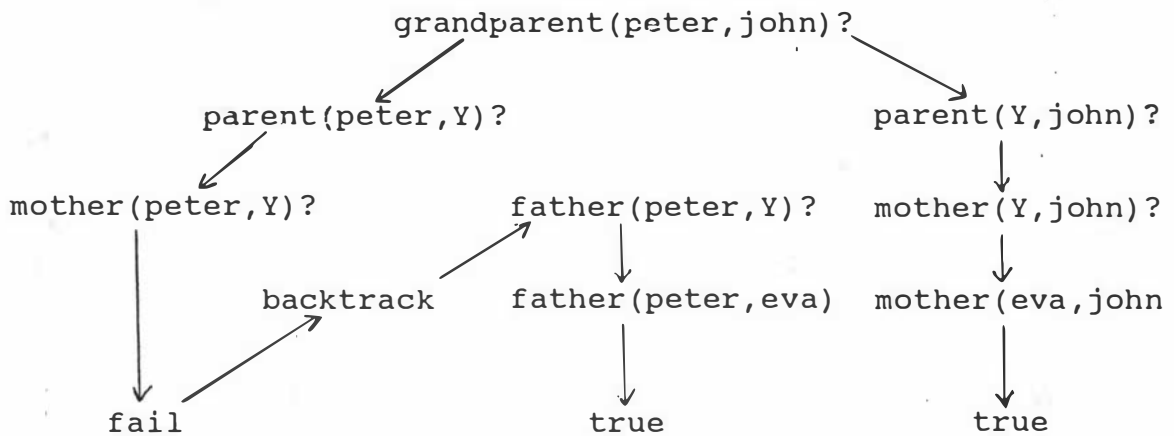
As both two conditions of 4.7 are true, grandparent(peter,john) is also true. Prolog will output "true" in response to the original query.

There are three interesting aspects of the above inference mechanism. The first one is that the resolution process for a query in Prolog is backward deduction. To solve a problem, Prolog tries to reduce it to smaller problems. These small problems are further reduced into smaller problems until the smallest problem is obtained which can be solved by a single clause in the knowledge base.

The second interesting aspect is that a step called unification is necessary in the derivation process. That is, if two matching Horn clauses contain variables, the head of the Horn clauses must be instantiated so that they become identical with each other before the resolution. Instantiation is done through substitution.

The third interesting feature in the control mechanism is that Prolog uses depth-first search with backtracking. This can be illustrated by representing the process of answering the query in the above example in the form of a tree as below:

Figure 4.4 Tree Representation of the Above Example



Prolog explores one branch of the search space (the entire tree) at a time. When it reaches a tip of the tree and if the query cannot be proved to be true, it backtracks and searches an alternative branch as close to the tip as possible. The process continues until either the query is proved to be true or the entire tree has been searched. Prolog outputs the message "False" in the latter case. Prolog assumes that a query must be false when it cannot be proved to be true.

4.4.4 Reasons for using Prolog as the Programming Language

The reasons for choosing Prolog as the programming language for the system are as follows:

- 1 A Prolog program for a given application typically requires only one tenth as many program lines as the corresponding Pascal program (Turbo Prolog 2.0 User's Guide 1988). This is mainly due to the use of recursive data structures and backtracking in Prolog.

Recursion can be done particularly effectively in Prolog because of the use of trees and lists as the basic data structures in the language. The following program is a typical example of the use of recursion in finding if "Name" is a member of a given list. Comments in the program begin with the characters "/*" and end with the characters "*/".

domains

```
namelist = name* /* namelist is defined as a list
                  of names */
name = symbol /* all the names are strings of
               characters */
```

predicates

```
member(name,namelist)
```

```
/* member is a predicate that describes the
relationship between the objects "name" and "namelist".
The predicate says that "name" is a member of
"namelist". */
```

clauses

```
member(Name, [Name|_]).
```

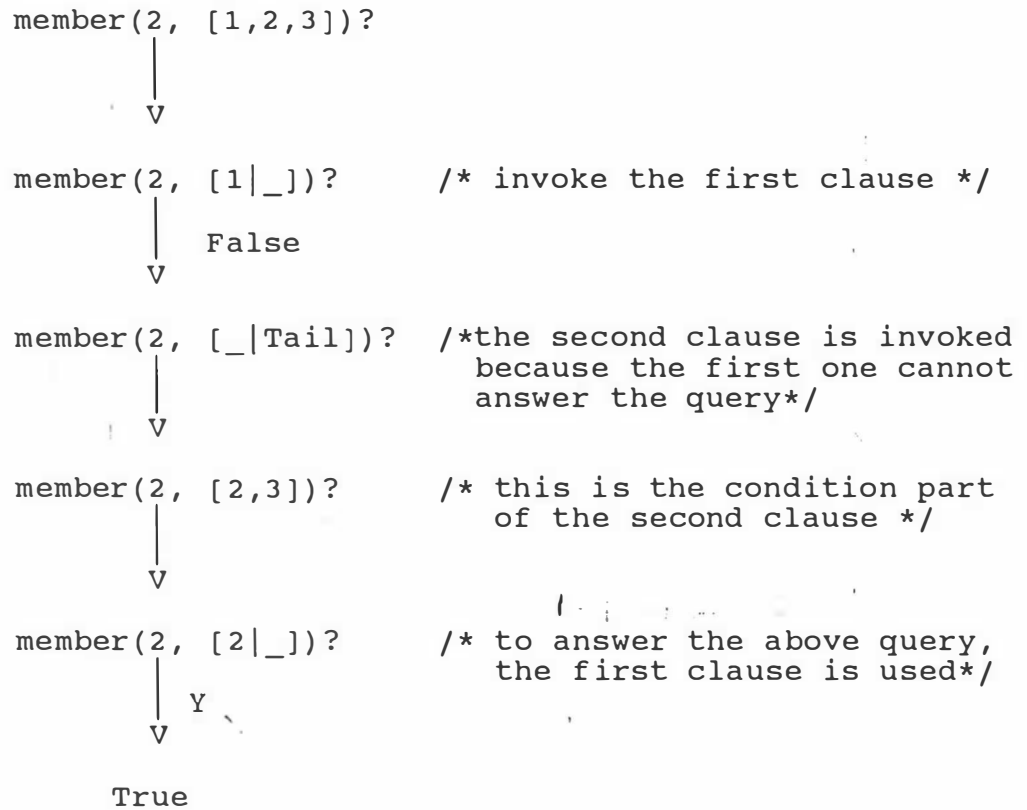
```
/* Name is a member of the list if Name is the first
element of the list. */
```

```
member(Name, [_|Tail]) :- member(Name, Tail).
```

```
/* Name is a member of the list if Name is a member of
the tail. */
```

The second clause above shows that the data structure "member(Name, Tail)" is defined within another data structure "member(Name, [_|Tail])". Using the above two clauses, the way the query "member(2, [1,2,3])?" is solved can be represented by the following tree:

Figure 4.5 Tree Solution of "member(2, [1,2,3])?"



The use of recursion often reduces the size of a program substantially.

On the other hand, selective backtracking and thoughtful application of the built-in predicate "cut" ("cut is a predicate that is built into the Prolog system) can significantly reduce the search space. This is because "cut" prunes part of the search tree that would normally be generated as a result of backtracking. It is impossible to backtrack across a "cut" to subgoals placed before the "cut". In other words, when a "cut" is encountered as a goal, the system becomes committed to all previous decisions

made. Any attempt to resatisfy the goal between the parent goal and the "cut" goal will fail. "Cut" makes the program work faster and occupy less computer memory because backtracking points do not need to be recorded for later use.

- 2 Prolog simulates backward inferencing which is consistent with the conceptual model of a marketing expert system proposed by Tse (1988, 1989).
- 3 When the system is written in Prolog, declarative knowledge can be added into its knowledge base when the system is upgraded. The declarative knowledge for price decision making can be obtained from standard textbooks, research reports and case studies (Tse and Alexander 1990c).
- 4 Programs written in Prolog have a high degree of referential transparency. Programs lacking referential transparency contain variables whose values change during the execution of the program, making the results of execution uncontrollable. Programs written in procedural languages like Pascal often cannot achieve a high degree of referential transparency. The following is an example of such a program written in Pascal:

```
program alan;  
var flag: boolean;  
function f(n:integer) : integer;  
begin
```

```

        if flag then f:=n
        else f:=0;
        flag:= not flag;
    end;

    begin
        flag:=true;
        writeln(f(10));
        writeln(f(10))
    end.

```

Upon execution of the program, two outputs can be obtained. The first output is ten. The second output is zero although it is expected that the answer should be ten. Of course, the problem is basically due to the poor skill of the programmer. Yet, programs written in a procedural language tend to have a high risk of committing this error because of the procedural nature of the control mechanism. The procedural nature makes program lines dependent on one another, resulting in variables with unpredicted values. For small programs the problem can be avoided with special care taken by the programmer. However, when the size of the program increases, the control of the problem becomes more difficult. For a declarative language like Prolog, the problem seldom occurs because each clause is self contained and is independent of other clauses in the program.

- 5 Extensive debugging facilities are also available in Prolog, the most outstanding of which is the compiler directive "trace". A trace program is available so that the execution of a program can be traced.

CHAPTER FIVE

TZ, THE KNOWLEDGE-BASED SYSTEM FOR PRICE DECISION MAKING

This chapter is divided into two sections. Section 5.1 gives a sample run of the system. Section 5.2 reports the results of system evaluation.

5.1 Sample Run of TZ

The programs for TZ are listed in Appendix 5. The documentation is contained within the program. The following is a sample run of the program. A sample run of the program can be obtained by typing "tz" with the Prolog system programs in the background.

The first window welcomes the user:

```
----- Welcome -----  
Key in 'H' for Help.  
  
Please type your name then  
Enter. Alan
```

The message at the top of the window instructs the user to key in "H" if help is required.

After the name is keyed in as shown above, the following

window appears:

Welcome to TZ, the Twilight Zone, Alan.

I am a Knowledge-based System to help you make better price decisions.

I am created by Alan Tse at the Department of Marketing, Massey University.

Please key in today's data in the following format:

DD/MM/YYYY. 14/07/1990

The date is required for the accounting system to process the input data. After the date is inputted into the system, the following window appears which allows the user to choose between using the accounting subsystem and the marketing knowledge-based system.

----- TZ - Marketing Knowledge-based System -----
Please press 'H' for help.

Please select one of the following two facilities available in this system, then press return.

1. Accounting System
2. Marketing Knowledge-based System

My choice is: 1

Suppose option 1 was chosen as shown above. The system then displays the facilities under the accounting system:

----- Accounting System -----

Please press 'H' for help.

Please select one of the following two facilities available in this accounting system, then press return.

1. Input Accounting Data
2. Output Financial Statements

My choice is:

Appendix 5 provides the listing program for the above two accounting subsystems. The "Input Accounting Data" subsystem accepts input transaction data and updates the balances in the books. The "Output Financial Statements" subsystem uses account balances in the books to construct the balance sheet, the trading account and the profit and loss account.

After the user has finished using the accounting subsystem, the user can go back to the main menu by pressing "Esc". Suppose this time the user chooses option 2.

----- TZ - Marketing Knowledge-based System -----
Please press 'H' for help.

Please select one of the following two facilities available in this system, then press return.

1. Accounting System
2. Marketing Knowledge-based System

My choice is: 2

After receiving the number "2" from the user, the system produces a graph showing the movements of the market indicators in the past twelve months. The diagram is preceded by the following description:

----- TZ - Marketing Knowledge-based System -----
Please press 'H' for help.

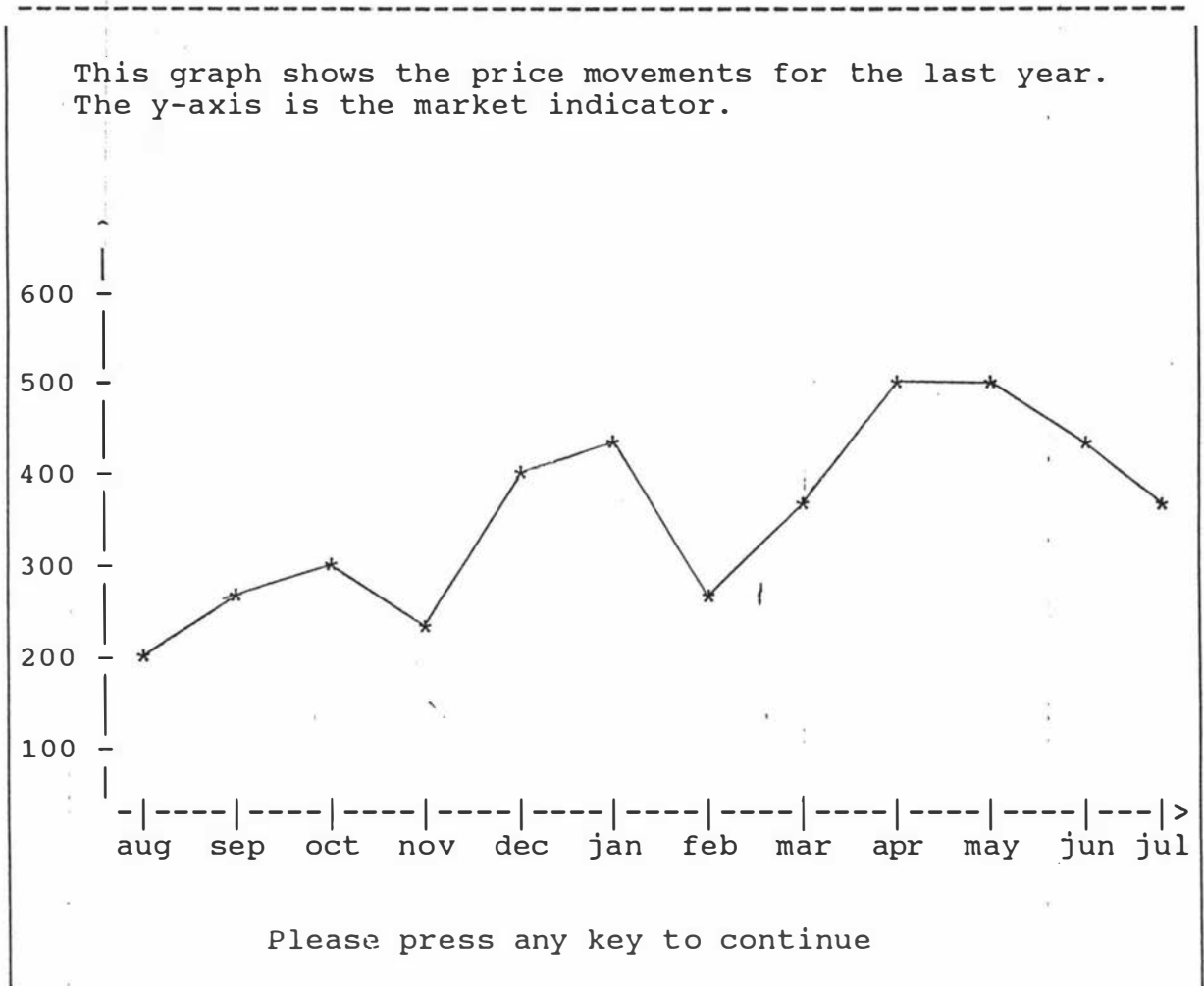
I am going to ask you to input your evaluation of the current market situation.

But before you input your judgment, I am going to show you a graphical display of the movements of the prices based on the market indicators of the last twelve months.

It is hoped that the diagram will be able to help you to identify whether the wool market is going up or coming down.

Please press any key to continue.

A sample graph showing the market situation is shown below:



After the above diagram has been displayed, the user's perception of the market situation is required by the system:

-----Please tell me your market situation-----

Please select one of the following options that best describes your perception of the current market situation:

1. Very very favourable
2. Very favourable
3. Favourable
4. Slightly favourable
5. Neither favorable nor unfavorable
6. Slightly unfavourable
7. Unfavourable
8. Very unfavourable
9. Very very unfavourable

Please indicate your choice by keying in the appropriate number: 3

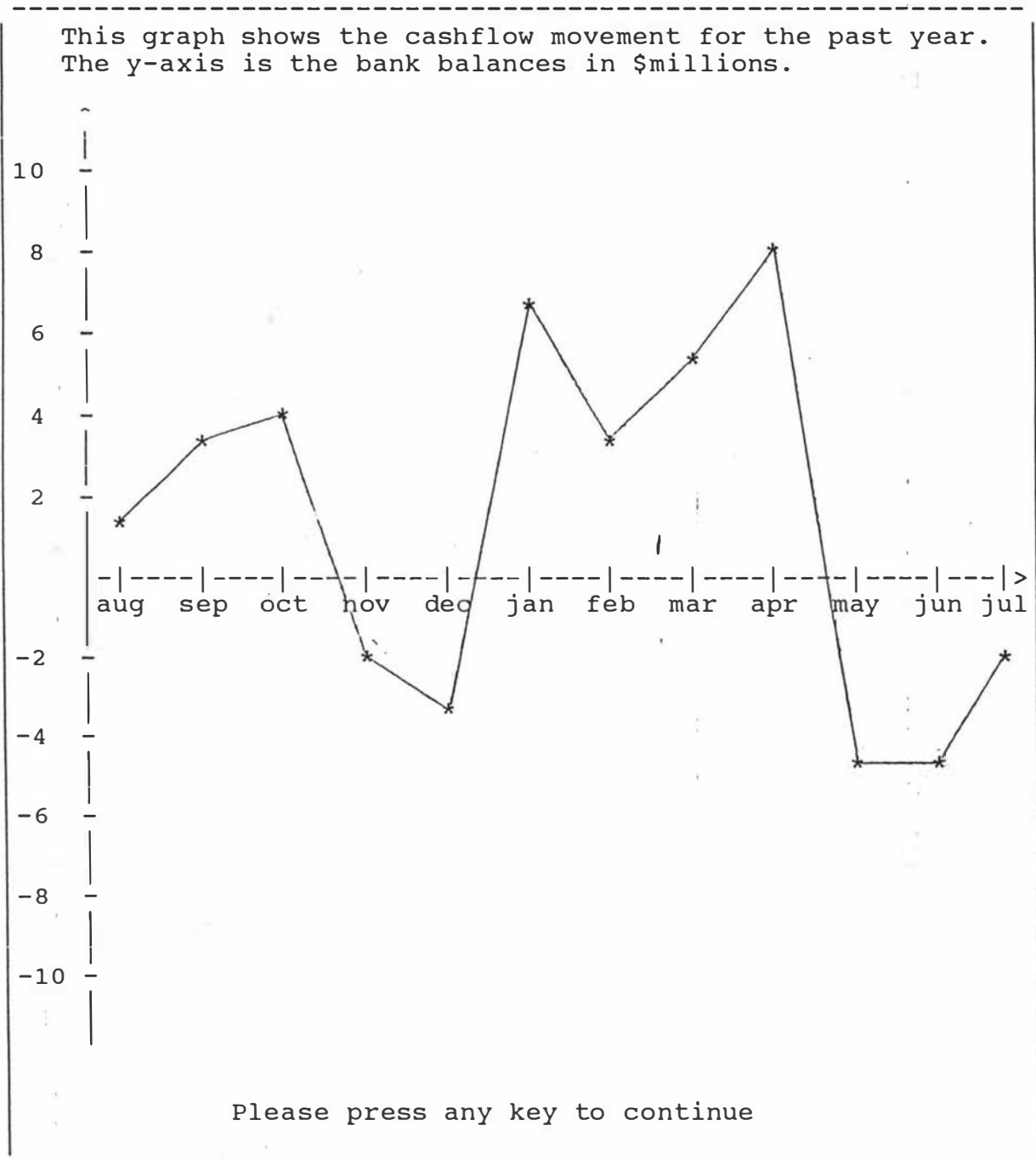
The next graph shown by the system is a cashflow situation diagram. The cashflow diagram shows the movement of the cashflow situation of the company in the past twelve months using the data in the accounting subsystem. The diagram is preceded by the following message:

----- TZ - Marketing Knowledge-based System -----

Now I am going to show you the cashflow situation of your company in the previous twelve months.

Please press any key to continue.

The following is an example of such a diagram:



The user is then prompted for the perception of cashflow situation:

-----Please tell me your cashflow situation-----

Please select one of the following options that best describes your current cashflow situation:

1. Very very favourable
2. Very favourable
3. Favourable
4. Slightly favourable
5. Neither favorable nor unfavorable
6. Slightly unfavourable
7. Unfavourable
8. Very unfavourable
9. Very very unfavourable

Please indicate your choice by keying in the appropriate number: 4

The next piece of information required to be inputted by the user is the competitive situation: 1

-----Please tell me your competitive situation-----

Please select one of the following options that best describes your current competitive situation:

1. Very very favourable
2. Very favourable
3. Favourable
4. Slightly favourable
5. Neither favorable nor unfavorable
6. Slightly unfavourable
7. Unfavourable
8. Very unfavourable
9. Very very unfavourable

Please indicate your choice by keying in the appropriate number: 2

TZ then asks for the amount of wool to be shipped to the buyer. Finally, it outputs the recommended price after going through all the calculations described in section 4.3. The output is shown below:

The recommended price for the 100 kg of wool you are going to sell to your customer is \$613.4.

Please press any key to continue.

The last window then asks whether or not the user wants to go on using the system. Suppose the response from the user is "n", then the following is produced:

----- TZ - Marketing Knowledge-based System -----

Do you wish to continue using the Marketing knowledge-based system?

(Please answer y/n) n

Thanks for using the system.

Please press any key to exit.

5.2 Evaluation of TZ

In this section, the objective of the research is first revisited in subsection 5.2.1 and then the results of the evaluation are reported in subsection 5.2.2.

5.2.1 The Objective Revisited

The primary objective of this research, as described in section 1.1 of the thesis, was to build up a knowledge-based system that performs better than most expert managers in providing price recommendations to a user. This is important because if the result is positive, then further investment and research can be put into the building up of larger scale systems, with the ultimate objective of storing all marketing decision making expertise in a computer system. The cost of building up a large scale knowledge-based system is very high. No commitment should be made without first research into the feasibility of the project.

If the result of the research is negative, further research must be done as to whether there are other better techniques that could be used to enable a successful system to be built. Although it has been explained in section 2.3.2 that fuzzy logic is a desirable technique to use given the high degree of fuzziness in a marketer's decision making environment, the technique has never been actually used as a means of modeling inexact reasoning in marketing. Hence, the application described in this thesis can be used as a testing ground for the applicability of the method in the

building up of knowledge-based systems in marketing. Should the method be found to be ineffective, modification can be sought so that a revised version of the logic, or possibly a new type of logic, can be developed that could enable a system to be constructed successfully for the marketing function.

5.2.2 System Evaluation

As described in section 3.4, the evaluation of the system was done by first of all creating a number of testing scenarios. These scenarios can be found in section one of the questionnaire in Appendix 2.

Individual managers were then asked to make recommendations in response to these problem scenarios. The Delphi method was then employed to enable the experts to achieve consensus regarding the recommendations to the problem scenarios. The consensus recommendations provided a standard against which to measure both the individual expert's recommendations and the system's recommendations.

Three sets of recommendations and eleven correlation coefficients were eventually obtained according to the procedure described in section 3.4. The correlation coefficient between the system's recommendations and the consensus recommendations was found to be 0.9337. The correlation coefficients between the individual managers' recommendations and the consensus recommendations ranged from -0.5380 to 0.9073.

It can be seen that the system outperforms, on the selected criterion, all the individual managers' recommendations. This indicates that the system is actually performing better than all the managers' in providing price recommendations, if it is accepted that the consensus recommendations are the best available recommendations. In other words, it appears possible for the system to achieve the consensus recommendations without going to the trouble of obtaining the consensus.

Of course, the conclusion made in the previous paragraph is based on a single sample. Nevertheless, the result is a piece of positive evidence that lends support to the idea that it is possible to construct a system that outperforms a good manager.

CHAPTER SIX

CONCLUSIONS AND DIRECTIONS FOR FUTURE RESEARCH

Since the result of the evaluation reported in section 5.2 is positive, it can be concluded that the knowledge-based system constructed for the wool industry is performing better than a good manager in providing price recommendations. Although the evaluation was based on a single sample of thirty scenarios and the system was specific to the export wool industry, the result is an encouraging piece of positive evidence.

Also, the use of fuzzy logic appears to be an effective knowledge representation technique as well as an effective method for modeling inexact reasoning. The method at least has been shown to be a good one to use for the building up of knowledge-based systems for supporting price decision making in the export wool industry.

Although the system appears to be a success in the current application, further research needs to be conducted in the following areas to upgrade the performance of the system in an incremental manner as suggested in section 4.1.

First of all, TZ has not fully automated a manager's judgment of the fuzzy variables described in this research. In order that a genuine expert system can be developed, further study is required as to how all expert judgments may be incorporated in the system constructed.

Second, more decision variables need to be incorporated into

the system. The more variables incorporated into the system, the better the performance of the system. This must be done in an incremental fashion as there are a large number of variables in the decision making environment.

Third, TZ can only handle price decisions in the wool industry. Other elements of the marketing mix in the industry should be incorporated as well.

Fourth, the hedge functions used in the system are the ones suggested by Zadeh (1976a). These functions might not be the best functions to use in the current application. Studies need to be done to determine the best functional forms to use in the current application.

Finally, TZ is only suitable for supporting price decision making in the export wool industry. It will not be useful for other industries.

REFERENCES

- Ackoff, R. L. 1979 "The Future of Operations Research is Past;" Journal of the Operational Research Society 30,2.
- AI Interactions 1986 "Capital Investment Expert System;" October, pp. 3-4.
- Barr, A., Feigenbaum, E. A. 1981 "The Handbook of Artificial Intelligence;" William Kaufman.
- Bellman, R., Giertz, M. 1973 "On the Analytic Formalism of the Theory of Fuzzy Sets;" Information Sciences 5, pp. 149-56.
- Boose, J. H. 1986 "Expertise Transfer for Expert System Design;" Elsevier, Chapter 1.1.
- Brancheau, J. C. and Wetherbe, J. C. 1987 "Key Issues in Information Systems Management;" MIS Quarterly, Vol. 11, pp. 23-45.
- Brightman, H. J. and Verhoeven, P. 1986 "Why Managerial Problem Solving Groups Fail?" Business, Vol. 36, pp. 24-9.
- Buchanan, B. G., Shortliffe, E. H. 1984 "Rule-based Expert Systems, The Mycin Experiments of the Stanford Heuristic Programming Project;" Addison Wesley.
- Carzo, R., Jr., 1963 "Some Effects of Organization Structure on Group Effectiveness;" Administrative Science Quarterly 7, pp.393-424.
- Clouser, E. R. 1986 "How the Delphi Panel Prognosticates the Future;" Risk Management, Vol. 33, pp. 30-41.
- Collins, H. M. 1985 "Changing Order: Replication and Induction in Scientific Practice;" London, Sage.
- Dalkey, N. C., Brown, B., Cochran, S., 1970 "The Delphi Method IV: Effect of Percentile Feedback and Feed-in of Relevant Facts;" Rank Corporation.
- Dhananjayan, R. S., Raman, V. J., Sarukesi, K. 1988 "Application of Expert System Techniques for Analyzing Firm's Fall in Market Share;" Proceedings of the 2nd International Workshop on Artificial Intelligence.
- Dixon, N. 1981 "Preconscious Processing;" Wiley.
- Erffmeyer, R. C. and Erffmeyer, E. S. 1975 "The Delphi Technique: An Empirical Evaluation of the Optimal Number of Rounds;" Group and Organization Studies, Vol. 11, pp. 120-8.
- Feigenbaum, E. A. 1977 "The Art of Artificial Intelligence, Themes and Case Studies of Knowledge Engineering;" IJCAI 5,

pp.1014-29.

Fraser, N. M., Hipel, K. W., Kilgour, D. M., McNeese, M. D. and Snyder, D. E. 1989 "An Architecture for Integrating Expert Systems;" *Decision Support Systems*, Vol. 5, No. 3, pp. 263-76.

Freud, S. 1914 "Psychopathology of Everyday Life;" Benn.

Gibson, S. and Cortese, A. 1989 "AI System to Monitor Southern California Electricity;" *Computerworld*, Jan. 2, p. 64.

Gibson, L. J. and Miller, M. M. 1990 "A Delphi Method for Planning "Preemptive" Regional Economic Diversification;" *Economic Development Review*, Vol. 8, pp.34-41.

Green, P. E., Tull, D. S. and Albaum, G. 1988 "Research for Marketing Decisions;" 5th ed., Prentice-Hall.

Hamacher, H. 1976 "On Logical Connectives of Fuzzy Statements and their Affiliated Truth Functions;" *Proc. 3rd European Meeting Cybern. & Syst. Res.*, Vienna (Austria).

Hayes-Roth, F., Waterman, D. A., Lewnet, D. B. 1983 "Building Expert Systems;" Addison-Wesley.

Horn, A. 1951 "On Sentences which are True of Direct Unions of Algebras;" *Journal of Symbolic Logic*, 16, pp.14-21.

Hosseini, A. and Khorramshahgol, R. 1990 "Analytic Delphi Method (ADM): A Strategic Decision Making Model Applied to Location Planning;" *Engineering Costs and Production Economics*, Vol. 20, No.1, pp. 23-8.

Kaufmann, A. 1975 "Introduction to the Theory of Fuzzy Subsets, Vol 1;" Academic Press, chapters 54-56.

Lehman, M. M. 1989 "Uncertainty in Computer Applications and its Control through the Engineering of Software;" *Journal of Software Maintenance*, Vol. 1 No. 1, pp.3-28.

Levine, R. I., Drang, D. E., Edelson, B. 1986 "A Comprehensive Guide to AI and Expert Systems;" McGraw-Hill.

Linstone, H. A., Turoff, M. 1975 "The Delphi Method, Techniques and Application;" Addison Wesley.

Merry, M. 1985 "Expert Systems 85;" *Proceedings of the Fifth Technical Conference of the BCS Specialist Group on Expert Systems*, Cambridge University Press.

Meyer, M. and Curley, K. 1989 "Expert System Success Models;" *Datamation*, p. 36.

Minsky, J. 1975 "A Framework for Representing Knowledge;" in P. Winston ed. *The Psychology of Computer Vision*, McGraw-Hill.

- Mockler, J. 1989 "Knowledge-based Systems for Management;" Prentice-Hall.
- Newquist, H. 1986 "Expert Systems: The Promise of a Smart Machine;" Computer-World, Jan. 13.
- O'brien, J. A. 1990 "Management Information Systems: A Managerial End User Perspective;" Addison-Wesley.
- Oddie, G. 1989 "Introduction to Logic;" 34305 Study Guide, Department of Philosophy, Massey University.
- Parkinson, L. K., Parkinson, S. T. 1987 "Using the Microcomputer in Marketing;" McGraw-Hill.
- Ray, D. 1988 "Delphi Forecasting: Market Research Method of the 1990s;" Marketing News, Vol. 22, pp. 17.
- Ray, P. K. and Sahu, S. 1990 "Productivity Management in India: A Delphi Study;" International Journal of Operations & Production Management, Vol. 10, No. 5, pp. 25-51.
- Shaw, M. L. G., Gaines, B. R. 1987 "Advances in Interactive Knowledge Engineering;" in Research and Development in Expert System III ed. Bramer, M. A., pp.111-22.
- Tate, G. 1990 "Prototyping: Helping to Build the Right Software;" Information and Software Technology, Vol. 32, No. 4, pp.237-44.
- Trewatha, R. L., Hampton, R., Vaught, B. C. and Parker, S. 1988 "A Multiple Branch Location Model: A Method to Analyze Site Selection Factors;" Akron Business and Economic Review, Vol. 19, pp. 66-75.
- Tse, A., Kemp, R., Greenwood, J., Eagle, C., 1988 "Maintaining Flexibility in Expert System Design;" Proceedings of the Third New Zealand Conference on Expert Systems, pp.143-54.
- Tse, A. 1989 "A Prolog-based Expert System for Price Decision Making under Incomplete Knowledge;" in Expert Systems in Economics, Banking and Management, eds L. F. Pau, J. Motiwalla, Y. H. Pao and H. H. Teh, North-Holland, pp.299-308.
- Tse, A. Alexander, P. 1990 "Price Decision Rules for an Expert System;" Marketing Bulletin, Vol. 1, May, pp.46-9.
- Tse, D. and Syed, J. R. 1988 "An Integrated Consulting System for Competitive Analysis and Planning Control;" in Ernst, C. J. ed. Management Expert Systems, Addison-Wesley.
- Turbo Prolog User's Guide, Borland International 1988.
- Turbo Prolog Reference Guide, Borland International 1988.
- Turbo Prolog Toolbox, Borland International 1988.

Turoff, M. 1970 "The Design of a Policy Delphi;" Technological Forecasting and Social Change 2, No.2.

Waterman, D. A. 1986 "A guide to Expert System;" Addison Wesley.

Yong, Y. W., Keng, K. A. and Leng, T. L. 1989 "A Delphi Forecast for the Singapore Tourism Industry: Future Scene and Marketing Implications;" European Journal of Marketing, Vol. 23, pp. 15-26.

Zadeh, L. A. 1965 "Fuzzy Sets;" Information and Control 8, pp. 338-353.

Zadeh, L. A. 1968 "Probability Measures of Fuzzy Events;" Journal of Mathematical Analysis & Applications 23, pp.421-27.

Zadeh, L. A. 1973 "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes;" IEEE Trans. System, Man & Cybernetics, SMC-3, pp. 28-44.

Zadeh, L. A. 1975a "The Concept of Linguistic Variable and its Application to Approximate Reasoning;" Information Science, Part I: 8, pp.199-249; Part II:8 pp.301-57; Part III: 9, pp.43-80.

Zadeh, L. A., Fu, K. S., Tanaka, K., Shimura, M. 1975b "Fuzzy Sets and their Applications to Cognitive and Decision Processes;" Academic Press.

Zadeh, L. A. 1976a "The Linguistic Approach and its Application to Decision Analysis;" in Ho Y. C., Mitter S. K., (eds): Directions in Large-Scale Systems, Plenum.

Zadeh, L. A. 1976b "A Fuzzy-Algorithmic Approach to the Definition of Complex or Imprecise Concepts;" International Journal of Man-Machine Studies 8, pp.249-91.

Zadeh, L. A. 1979 "A Theory of Approximate Reasoning" in Hayes J. E., Michie D., Kulick L. I. (eds), Machine Intelligence 9, Wiley, pp.149-94.

Zimmermann, H. J. 1985 "Fuzzy Set Theory and its Application;" Kluna-Nijhoff.

APPENDIX 1

INVITATION LETTER FOR THE PRELIMINARY STUDY

Dear Sir/Madam,

I am undertaking research into the price decision making behaviour of wool exporters in New Zealand. The objective of the study is to build up a computer system that simulates price decision making in the industry.

The most important phase of the project is to conduct personal interviews with successful wool marketers like yourself to collect expert opinions on how price decisions should be made. The decision making methods would then be stored in the computer for simulating price decision making.

I will, in a few days' time, get in touch with you again to see if further arrangements could be made regarding the interview. Meanwhile, if you have any queries, please feel free to let me know on 69099 extn. 8083.

Thank you for your kind attention in this matter.

Yours sincerely,

Alan Tse

QUESTIONNAIRE FOR PRELIMINARY STUDY

DEPARTMENT OF MARKETING
MASSEY UNIVERSITY

PRELIMINARY SURVEY OF PRICE DECISION MAKING
METHODS IN THE EXPORT WOOL INDUSTRY

- 1 Could you please tell me in general how you quote a price on the wool you sell to your customers?

- 2 Would you please tell me what important factors you would take into account when you make your price decisions?

a)

b)

c)

d)

e)

f)

g)

h)

i)

j)

- 3 How does each one of the above factors affect your everyday price decisions?

(PROBE FOR EACH ONE OF THE FACTORS MENTIONED IN QUESTION 2)

- 4 Just then I collected from you several important factors affecting your price determination. What I want to do next is to get your expert opinions on how price decisions should be made.

Now, under what situations would you recommend a small positive change in price?

Under what situations would you recommend a positive change in price?

Under what situations would you recommend a very positive change in price?

Under what situations would you recommend a small negative change in price?

Under what situations would you recommend a negative change in price?

Under what situations would you recommend a very negative change in price?

Could I say that under all other situations you would not change the price of your product?

Yes ☐ --> IF YES, END THE INTERVIEW

No ☐ --> IF NO, GO TO Q.5

5 Under what situations would you not change the price of your product?

THANK YOU FOR YOUR KIND COOPERATION.

APPENDIX 2

COVERING LETTER AND QUESTIONNAIRE FOR SECOND STAGE OF STUDY

COVERING LETTER

Department of Marketing

Dear Sir/Madam,

I am undertaking a research into the building up of a decision support system for wool marketers in New Zealand. The study is divided into several phases. The objective of this phase of the study is to build a subsystem that helps managers to make better price decisions.

Enclosed together with this letter is a questionnaire that collects information about pricing methods in the industry as well as data about a number of parameters for building up the system. The questionnaire is intended to be filled in by the marketing manager of this company who is responsible for making price decisions. The information collected will be kept strictly confidential.

I would be grateful if you would take the time to complete the questionnaire and return it to me in the postage-free envelope enclosed. As your opinions will be highly valuable in this research, it is very important that I get completed questionnaire from you. Your earliest response will be greatly appreciated.

Yours faithfully,

Alan Tse

QUESTIONNAIRE FOR THE SECOND STAGE OF STUDY

DEPARTMENT OF MARKETING
MASSEY UNIVERSITY

SURVEY OF PRICE DECISION MAKING METHODS IN THE EXPORT WOOL INDUSTRY

INSTRUCTIONS: PLEASE ANSWER THE FOLLOWING QUESTIONS BY EITHER TICKING (✓) THE APPROPRIATE BOXES OR BY WRITING IN THE SPACE PROVIDED.

SECTION 1: PRICING DECISIONS UNDER DIFFERENT SCENARIOS

- 1 Assuming that in your business, nothing else changes but the market outlook, cashflow situation and competitive situation, please tell me by how many percent you would increase or decrease the price you are ready to quote your buyers for wool under each one of the following scenarios:

INSTRUCTIONS:

THE FOLLOWING TABLE CONTAINS FIVE COLUMNS. THE FIRST THREE COLUMNS REPRESENT THE THREE VARIABLES OF INTEREST IN THE DECISION MAKING ENVIRONMENT. EACH SCENARIO IS A COMBINATION OF THESE THREE VARIABLES. FOR EXAMPLE, THE FIRST SCENARIO IS MADE UP OF "Unfavorable" MARKET OUTLOOK (Column 1), "Favorable" CASHFLOW SITUATION (Column 2) AND "Very weak" COMPETITIVE SITUATION (Column 3). PLEASE PUT DOWN THE APPROPRIATE PERCENTAGE PRICE CHANGE YOU WOULD RECOMMEND IN RESPONSE TO THIS SCENARIO IN THE SPACE PROVIDED UNDER EITHER Column 4 OR Column 5.

Column 1	Column 2	Column 3	Column 4	Column 5
Market outlook	Cashflow situation	Competition	Percentage of price increase	Percentage of price decrease
Unfavorable	Favorable	Very weak	____%	____%
Very very favorable	Neither favorable nor unfavorable	Neither strong nor weak	____%	____%
Very favorable	Very favorable	Very strong	____%	____%
Very favorable	Favorable	Strong	____%	____%
Very favorable	Very favorable	Strong	____%	____%
Very favorable	Unfavorable	Weak	____%	____%

Market outlook	Cashflow situation	Competition	Percentage of price increase	Percentage of price decrease
Very favorable	Unfavorable	Very strong	___%	___%
Very favorable	Unfavorable	Strong	___%	___%
Very favorable	Very unfavorable	Very strong	___%	___%
Very very unfavorable	Neither favorable nor unfavorable	Neither strong nor weak	___%	___%
Favorable	Very favorable	Strong	___%	___%
Favorable	Very favorable	Weak	___%	___%
Favorable	Favorable	Strong	___%	___%
Favorable	Very unfavorable	Very strong	___%	___%
Favorable	Very unfavorable	Weak	___%	___%
Favorable	Neither favorable nor unfavorable	Neither strong nor weak	___%	___%
Unfavorable	Neither favorable nor unfavorable	Neither strong nor weak	___%	___%
Unfavorable	Very favorable	Very strong	___%	___%
Unfavorable	Favorable	Very strong	___%	___%
Unfavorable	Favorable	Strong	___%	___%
Unfavorable	Unfavorable	Strong	___%	___%
Neither favorable nor unfavorable	Favorable	Neither strong nor weak	___%	___%
Neither favorable nor unfavorable	Unfavorable	Neither strong nor weak	___%	___%
Unfavorable	Very unfavorable	Very strong	___%	___%
Neither favorable nor unfavorable	Neither favorable nor unfavorable	Strong	___%	___%
Neither favorable nor unfavorable	Neither favorable nor unfavorable	Weak	___%	___%
Very unfavorable	Very favorable	Very strong	___%	___%
Very unfavorable	Favorable	Strong	___%	___%
Very unfavorable	Very unfavorable	Very strong	___%	___%

Very unfavorable Very unfavorable Weak

_____ % _____ %

SECTION 2: PARAMETERS FOR DECISION SUPPORT SYSTEM

INSTRUCTIONS:

FOR EACH ONE OF THE QUESTIONS IN THIS SECTION, PLEASE FILL IN THE BLANKS USING ANY NUMBER BETWEEN 0 AND 1 (INCLUDING 0 AND 1). FOR EXAMPLE, IN QUESTION TWO, IF YOU HAVE NO DOUBT THAT A GIVEN PRICE INCREASE SHOULD BELONG TO THE CATEGORY OF SMALL POSITIVE PRICE CHANGE, PUT THE NUMBER 1 IN THE SPACE PROVIDED FOR THE GIVEN PRICE INCREASE. IF YOU HAVE NO DOUBT THAT A GIVEN PRICE INCREASE SHOULD NOT BELONG TO THE CATEGORY OF SMALL POSITIVE PRICE CHANGE, PUT THE NUMBER 0 IN THE SPACE PROVIDED FOR THAT GIVEN PRICE INCREASE. FOR OTHER CASES, PLEASE USE A NUMBER BETWEEN 0 AND 1 TO REFLECT THEIR DEGREES OF MEMBERSHIP IN THE CATEGORY OF SMALL POSITIVE PRICE INCREASE. BEFORE YOU ANSWER THE QUESTIONS, PLEASE READ THE FOLLOWING EXAMPLE:

EXAMPLE:

PERCENTAGE OF PRICE INCREASE

GRADE OF MEMBERSHIP IN THE CATEGORY OF "SLIGHT PRICE INCREASE"

1%	0.0
2%	0.5
3%	1.0
5%	0.2
7%	0.0

THE ABOVE MEANS THAT:

- i A 1% PRICE INCREASE IS DEFINITELY NOT A SLIGHT PRICE INCREASE, SO THE DEGREE OF BELIEF IS 0.0.
- ii I AM NOT SO SURE WHETHER OR NOT A 2% PRICE INCREASE SHOULD BE REGARDED AS A SLIGHT PRICE INCREASE. THE DEGREE OF BELIEF IS 0.5.
- iii I AGREE TOTALLY THAT A 3% PRICE INCREASE IS A SLIGHT PRICE INCREASE. THE DEGREE OF BELIEF IS 1.0.
- iv 5% IS A BIT TOO HIGH TO BE REGARDED AS A SLIGHT PRICE INCREASE. MY DEGREE OF BELIEF THAT IT SHOULD BE CATEGORIZED AS A SLIGHT PRICE INCREASE IS 0.2.
- v. 7% IS TOO HIGH TO BE CONSIDERED AS A SLIGHT PRICE INCREASE. MY DEGREE OF BELIEF IS 0.

OF COURSE, YOUR DEGREE OF BELIEF WOULD BE DIFFERENT FROM MINE.

NOW, PLEASE ANSWER THE FOLLOWING QUESTIONS IN LIKE MANNER AS THE ABOVE EXAMPLE:

2 Please tell me the extent to which you believe each one of the following percentage of price increases should be regarded as a small positive change in price, by writing numbers between 0 and 1 in the appropriate blanks provided:

Percentage of price increase	Grade of membership in the category of "small positive change in price"
1%	_____
2%	_____
3%	_____
4%	_____
5%	_____
6%	_____
7%	_____
8%	_____
9%	_____
10%	_____
11%	_____
12%	_____
13%	_____
14%	_____
15%	_____

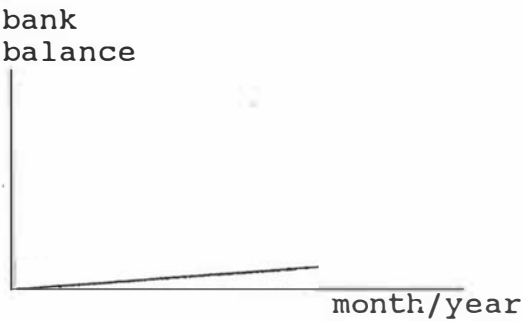
- 3 Please tell me the extent to which you believe each one of the following percentage of price decreases should be regarded as a small negative change in price by writing numbers between 0 and 1 in the appropriate space provided:

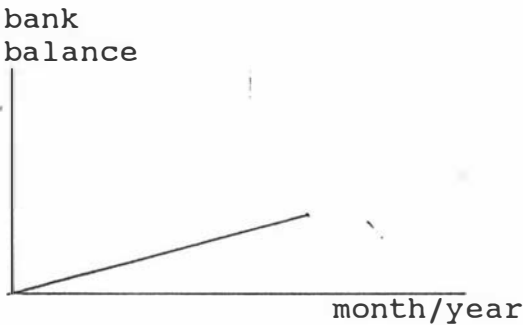
Percentage of price decrease	Grade of membership in the category of "small negative change in price"
------------------------------------	---

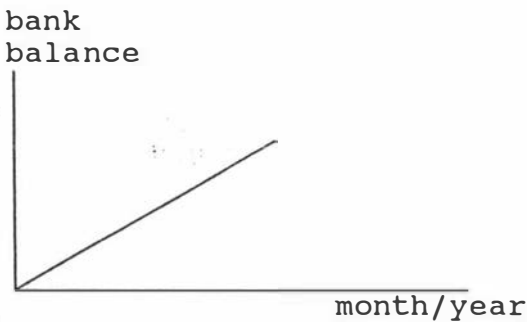
1%	_____
2%	_____
3%	_____
4%	_____
5%	_____
6%	_____
7%	_____
8%	_____
9%	_____
10%	_____
11%	_____
12%	_____
13%	_____
14%	_____
15%	_____

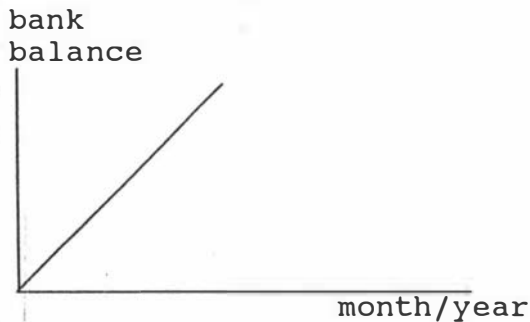
4 Each diagram below describes the movement of the cashflow position of a company in the industry in the previous twelve months. Please tell me the extent to which you believe each one of the following cashflow situations should be regarded as a favorable cashflow situation by writing a number between 0 and 1 in the appropriate space provided:

Grade of Membership
in the category of
"favorable cashflow
situation"

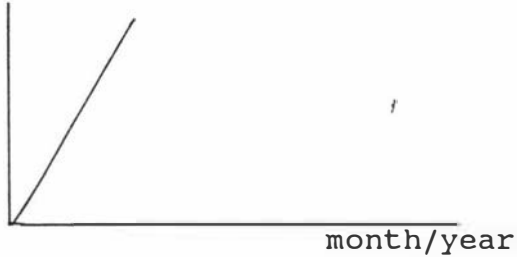








bank
balance



Grade of Membership
in the category of
"favorable cashflow
situation"

bank
balance

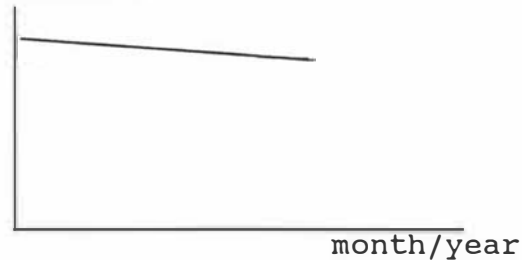


bank
balance



- 5 The diagrams below are similar to the diagrams in question 4. This time please tell me the extent to which you believe each one of the following cashflow situations should be regarded as an unfavorable cashflow situation by writing a number between 0 and 1 in the appropriate space provided:

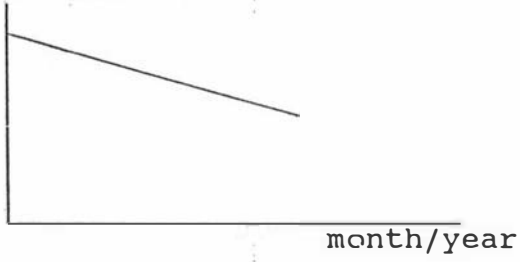
bank
balance



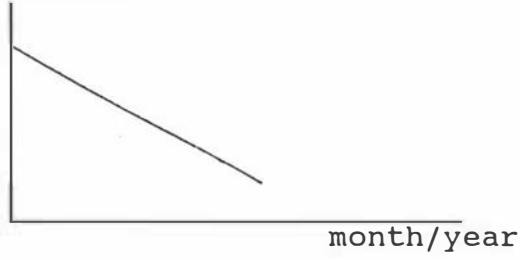
Grade of Membership
in the category of
"unfavorable cashflow
situation"

Grade of Membership
in the category of
"unfavorable cashflow
situation"

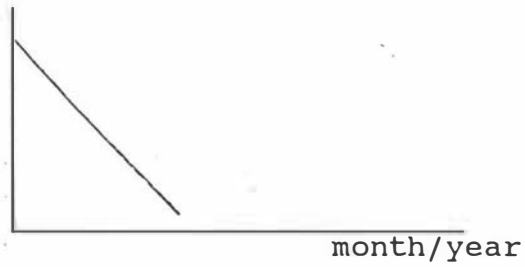
bank
balance



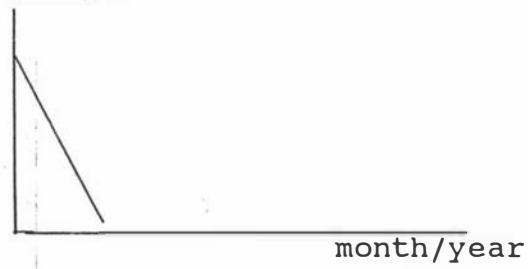
bank
balance



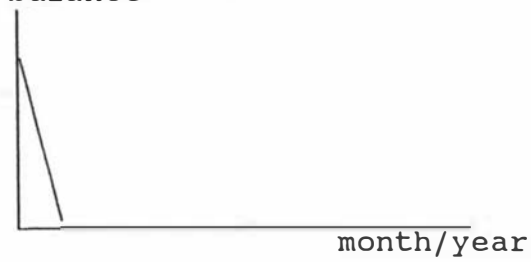
bank
balance



bank
balance



bank
balance



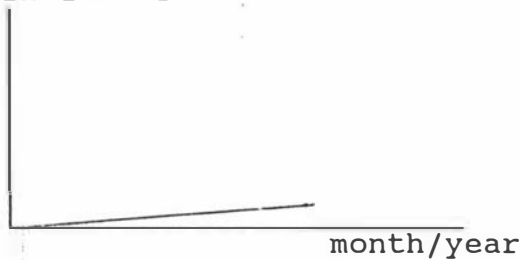
bank
balance



Grade of Membership
in the category of
"unfavorable cashflow
situation"

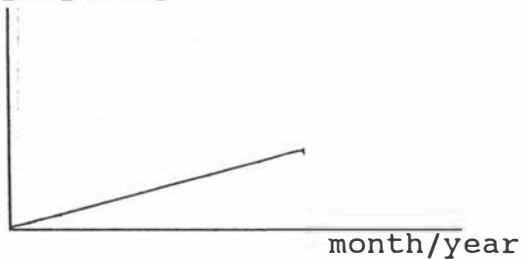
- 6 Each diagram below describes the movements of the market indicator in the previous twelve months. Please tell me the extent to which you believe each one of the following market situations described by the diagrams should be regarded as a favorable market situation by writing a number between 0 and 1 in the appropriate space provided:

market
indicator

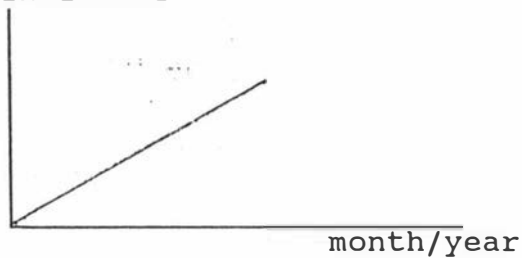


Grade of Membership
in the category of
"favourable market
situation"

market
indicator

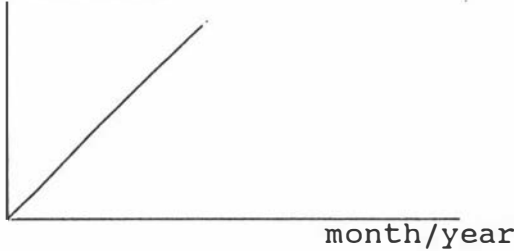


market
indicator

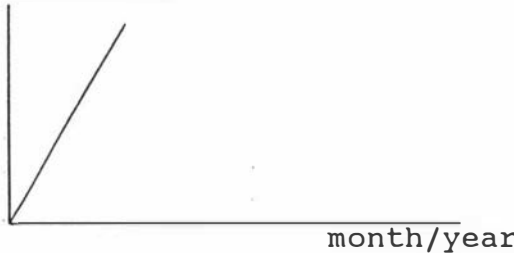


Grade of Membership
in the category of
"favourable market
situation"

market
indicator



market
indicator



market
indicator

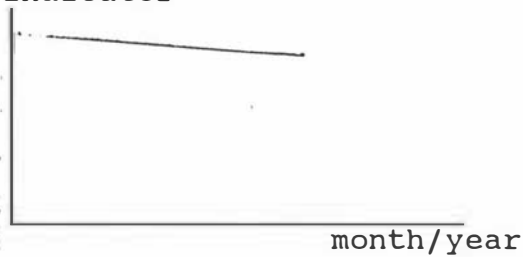


market
indicator



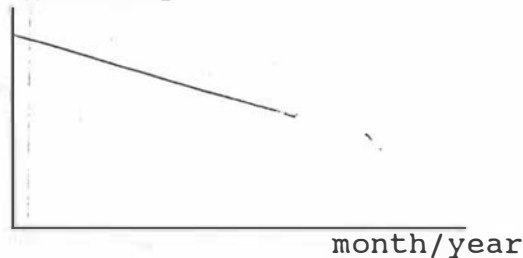
7 The diagrams below are similar to the diagrams in question 6. This time please tell me the extent to which you believe each one of the following market situations should be regarded as an unfavorable market situation by writing a number between 0 and 1 in the appropriate space provided:

market
indicator

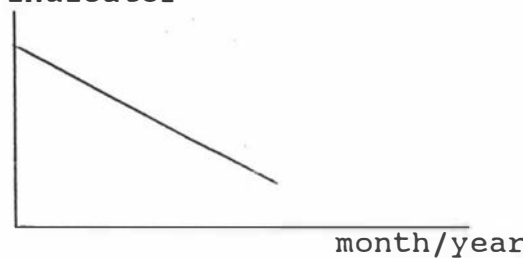


Grade of Membership
in the category of
"unfavourable market
situation"

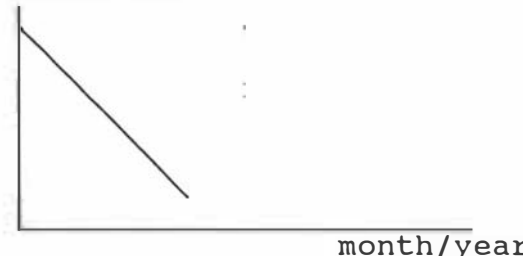
market
indicator



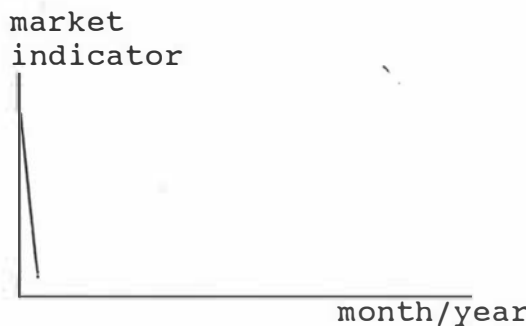
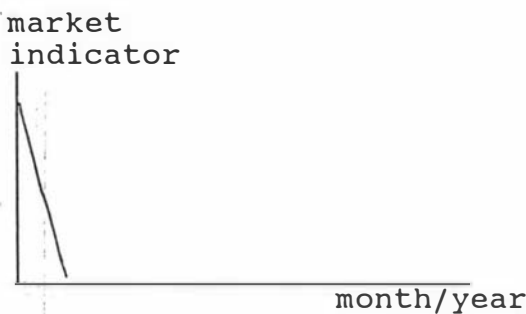
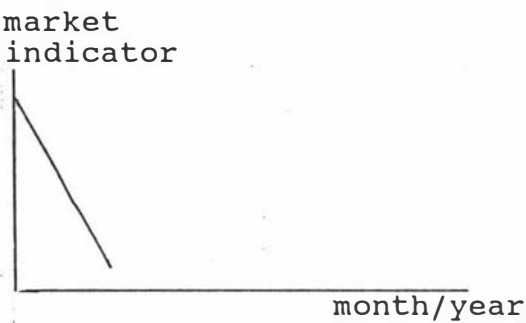
market
indicator



market
indicator



Grade of Membership
in the category of
"unfavourable market
situation"



8 For this question, I would like you to look at the following percentages. The following percentages are obtained in this manner. First, the difference between the price of your product and the price of your strongest competitor's product is obtained. Then the difference is expressed as a percentage of the price of your product. Now, would you please tell me the extent to which you believe each one of the following percentages should belong to the category of "unfavorable" competitive situation by writing a number between 0 and 1 in the appropriate space provided?

Percentage that your price is <u>above</u> that of your competitors' price	Grade of Membership in the category of "unfavourable competitive situation"
1%	_____
2%	_____
3%	_____
4%	_____
5%	_____
6%	_____
7%	_____
8%	_____
9%	_____
10%	_____
11%	_____
12%	_____
13%	_____
14%	_____
15%	_____

- 9 The following percentages are obtained in like manner as the percentages in question 8. Now, please tell me the extent to which you believe each one of the following percentages should belong to the category of "favorable" competitive situation by writing a number between 0 and 1 in the appropriate space provided.

Percentage that your price is below that of your competitors' price

Grade of Membership in the category of "favourable competitive situation"

1%	_____
2%	_____
3%	_____
4%	_____
5%	_____
6%	_____
7%	_____
8%	_____
9%	_____
10%	_____
11%	_____
12%	_____
13%	_____
14%	_____
15%	_____

SECTION 3: COMPANY PROFILE

- 10 Now I have some questions about your company and yourself, just so I can be sure that I have a good cross-section of companies and people in my sample.

Did your company hedge against exchange rate fluctuations through a financial institution?

Always

Never

Sometimes

☐☐☐

- 11 Can you please tell me which year you started working in the export wool industry?

Year: _____

THANK YOU FOR YOUR KIND COOPERATION.

APPENDIX 3

COVERING LETTER AND QUESTIONNAIRE FOR DELPHI SESSIONS

COVERING LETTER

Department of Marketing

Dear Sir,

I am writing to express my deep feelings of gratitude for your help and advice in my research on building up a price decision support system for marketers in the wool industry. The first round of the survey has been completed. A report summarizing the findings so far has been enclosed. The report summarizes the responses from ten carefully selected experts each of whom has at least ten years of marketing experience in the industry. As you are one of these ten experts selected, I am writing to you again to seek your comments on this summary report.

You will notice that the summary report is very similar to the questionnaire you responded to last time. For each question in the summary report, there are two sets of numbers. The first set of numbers were your own responses to the questions in the previous questionnaire. The second set of numbers are the average responses of the ten experts mentioned above. Under each question, spaces have been provided for you to revise your responses in case if your last responses are not the same as the averages of the group. Of course, you don't need to revise your opinions if you think it is not necessary.

Please send the summary report back to me with the stamped self-addressed envelope enclosed. After I have collected all the reports, I will compile another summary for your comment.

Please be assured that the survey will be kept strictly confidential.

Thank you for your kind cooperation in this matter.

Yours faithfully,

Alan Tse

QUESTIONNAIRE FOR DELPHI SESSIONS

DEPARTMENT OF MARKETING
MASSEY UNIVERSITY

SUMMARY REPORT OF PRICE DECISION MAKING METHODS IN THE EXPORT WOOL INDUSTRY

INSTRUCTIONS: THERE ARE THREE COLUMNS UNDER EACH ONE OF THE QUESTIONS BELOW. THE FIRST COLUMN CONTAINS YOUR RESPONSES TO THE SAME QUESTION IN THE PREVIOUS QUESTIONNAIRE. THE SECOND COLUMN CONTAINS THE AVERAGE RESPONSES OF THE TEN EXPERT MARKETERS WHOSE RESPONSES WERE USED TO COMPILE THIS REPORT. PLEASE COMPARE YOUR OWN RESPONSES TO THE AVERAGE RESPONSES OF THE GROUP AND PUT DOWN YOUR REVISED OPINIONS, IF ANY, IN THE APPROPRIATE SPACES PROVIDED IN THE THIRD COLUMN.

SECTION 1: PRICING DECISIONS UNDER DIFFERENT SCENARIOS

1 Assuming that in your business, nothing else changes but the market outlook, cashflow situation and competitive situation, please tell me by how many percent you would increase or decrease the price you are ready to quote your buyers for wool under each one of the following scenarios:

			Column1	Column2	Column3
Market outlook	Cashflow situation	Competition	Your own response	Group average	Your revised opinion (if any)
Unfavorable	Favorable	Very weak	<u>0</u> %	-0.65%	<u> </u> %
Very very favorable	Neither favorable nor unfavorable	Neither strong nor weak	<u>3</u> %	2.65%	<u> </u> %
Very favorable	Very favorable	Very strong	<u>2</u> %	2.3%	<u> </u> %
Very favorable	Favorable	Strong	<u>2</u> %	2.25%	<u> </u> %
Very favorable	Very favorable	Strong	<u>2</u> %	2.35%	<u> </u> %
Very favorable	Unfavorable	Weak	<u>3</u> %	2.15%	<u> </u> %
Very favorable	Unfavorable	Very strong	<u>3</u> %	1.5%	<u> </u> %
Very favorable	Unfavorable	Strong	<u>3</u> %	1.4%	<u> </u> %
Very favorable	Very unfavorable	Very strong	<u>3</u> %	1.1%	<u> </u> %
Very very unfavorable	Neither favorable nor unfavorable	Neither strong nor weak	<u>-3</u> %	-1.9%	<u> </u> %

Market outlook	Cashflow situation	Competition	Your own response	Group average	Your revised opinion (if any)
Favorable	Very favorable	Strong	<u>2</u> %	1.5%	<u> </u> %
Favorable	Very favorable	Weak	<u>2</u> %	1.15%	<u> </u> %
Favorable	Favorable	Strong	<u>2</u> %	0.9%	<u> </u> %
Favorable	Very unfavorable	Very strong	<u>3</u> %	0.1%	<u> </u> %
Favorable	Very unfavorable	Weak	<u>3</u> %	0.4%	<u> </u> %
Favorable	Neither favorable nor unfavorable	Neither strong nor weak	<u>2</u> %	0.5%	<u> </u> %
Unfavorable	Neither favorable nor unfavorable	Neither strong nor weak	<u>0</u> %	-1.1%	<u> </u> %
Unfavorable	Very favorable	Very strong	<u>0</u> %	-1.35%	<u> </u> %
Unfavorable	Favorable	Very strong	<u>0</u> %	-1.15%	<u> </u> %
Unfavorable	Favorable	Strong	<u>0</u> %	-0.95%	<u> </u> %
Unfavorable	Unfavorable	Strong	<u>0</u> %	-1.78%	<u> </u> %
Neither favorable nor unfavorable	Favorable	Neither strong nor weak	<u>0</u> %	0.55%	<u> </u> %
Neither favorable nor unfavorable	Unfavorable	Neither strong nor weak	<u>0</u> %	-0.2%	<u> </u> %
Unfavorable	Very unfavorable	Very strong	<u>0</u> %	-2.05%	<u> </u> %
Neither favorable nor unfavorable	Neither favorable nor unfavorable	Strong	<u>0</u> %	-0.2%	<u> </u> %
Neither favorable nor unfavorable	Neither favorable nor unfavorable	Weak	<u>1</u> %	-0.1%	<u> </u> %
Very unfavorable	Very favorable	Very strong	<u>-2</u> %	-1.75%	<u> </u> %
Very unfavorable	Favorable	Strong	<u>-2</u> %	-1.45%	<u> </u> %
Very unfavorable	Very unfavorable	Very strong	<u>-2</u> %	-3%	<u> </u> %
Very unfavorable	Very unfavorable	Weak	<u>-2</u> %	-3.03%	<u> </u> %

SECTION 2: PARAMETERS FOR DECISION SUPPORT SYSTEM


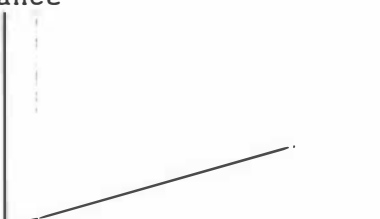
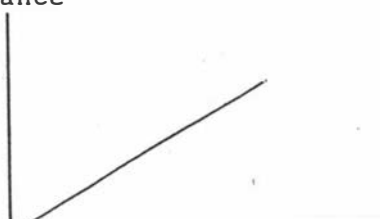
2 Please tell me the extent to which you believe each one of the following percentage of price increases should be regarded as a small positive change in price by writing numbers between 0 and 1 in the appropriate blanks provided:

Percentage of price increase	Grade of membership in the category of "small positive change in price"		
	Your own response	Group average	Your revised opinion (if any)
1%	<u>1</u>	0.95	<u> </u>
2%	<u>0.5</u>	0.89	<u> </u>
3%	<u>0</u>	0.41	<u> </u>
4%	<u>0</u>	0.35	<u> </u>
5%	<u>0</u>	0.28	<u> </u>
6%	<u>0</u>	0.28	<u> </u>
7%	<u>0</u>	0.24	<u> </u>
8%	<u>0</u>	0.22	<u> </u>
9%	<u>0</u>	0.21	<u> </u>
10%	<u>0</u>	0.20	<u> </u>
11%	<u>0</u>	0.20	<u> </u>
12%	<u>0</u>	0.20	<u> </u>
13%	<u>0</u>	0.20	<u> </u>
14%	<u>0</u>	0.20	<u> </u>
15%	<u>0</u>	0.20	<u> </u>

3 Please tell me the extent to which you believe each one of the following percentage of price decreases should be regarded as a small negative change in price by writing numbers between 0 and 1 in the appropriate space provided:

Percentage of price decrease	Grade of membership in the category of "small negative change in price"		
	Your own response	Group average	Your revised opinion (if any)
1%	<u>1</u>	0.95	<u> </u>
2%	<u>1</u>	0.81	<u> </u>
3%	<u>0.5</u>	0.60	<u> </u>
4%	<u>0</u>	0.28	<u> </u>
5%	<u>0</u>	0.25	<u> </u>
6%	<u>0</u>	0.27	<u> </u>
7%	<u>0</u>	0.27	<u> </u>
8%	<u>0</u>	0.25	<u> </u>
9%	<u>0</u>	0.24	<u> </u>
10%	<u>0</u>	0.24	<u> </u>
11%	<u>0</u>	0.23	<u> </u>
12%	<u>0</u>	0.23	<u> </u>
13%	<u>0</u>	0.22	<u> </u>
14%	<u>0</u>	0.21	<u> </u>
15%	<u>0</u>	0.20	<u> </u>

4 Each diagram below describes the movement of the cashflow position of a company in the industry in the previous twelve months. Please tell me the extent to which you believe each one of the following cashflow situations should be regarded as a favorable cashflow situation by writing a number between 0 and 1 in the appropriate space provided:

Cashflow situation	Grade of Membership in the category of "favorable cashflow situation"		
	Your own response	Group average	Your revised opinion (if any)
<div>Bank Balance</div>  <div>month/year</div>	0,5	0.32	
<div>Bank Balance</div>  <div>month/year</div>	0.6	0.72	
<div>Bank Balance</div>  <div>month/year</div>	0.7	0.87	

Cashflow
situation

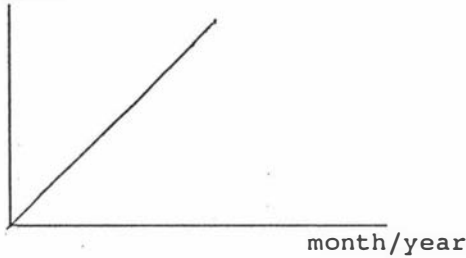
Grade of Membership
in the category of
"favorable cashflow
situation"

Your
own
response

Group
average

Your
revised
opinion
(if any)

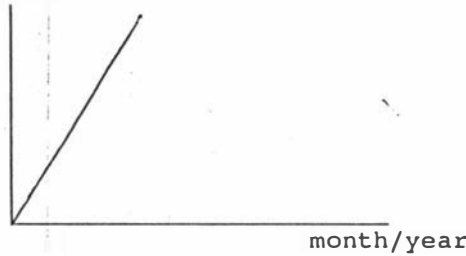
Bank
Balance



0.9

0.83

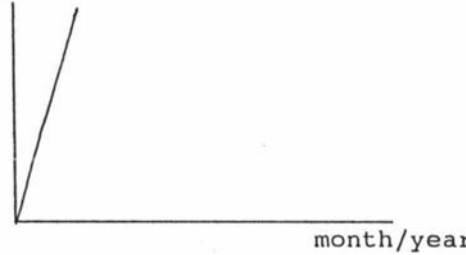
Bank
Balance



0.8

0.73

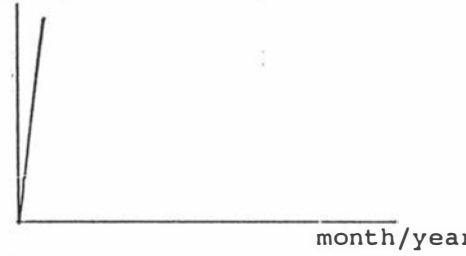
Bank
Balance



0.7

0.71

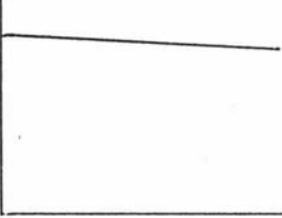
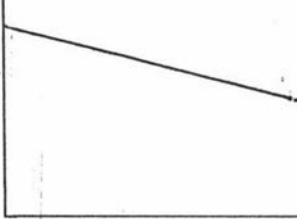
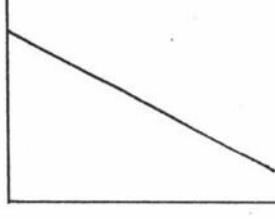
Bank
Balance



0.5

0.70

5 The diagrams below are similar to the diagrams in question 4. This time please tell me the extent to which you believe each one of the following cashflow situations should be regarded as an unfavorable cashflow situation by writing a number between 0 and 1 in the appropriate space provided:

Cashflow situation	Grade of Membership in the category of "unfavorable cashflow situation"		
	Your own response	Group average	Your revised opinion (if any)
<div>Bank Balance</div>  <div>month/year</div>	<u>0.5</u>	0.44	<u> </u>
<div>Bank Balance</div>  <div>month/year</div>	<u>0.5</u>	0.55	<u> </u>
<div>Bank Balance</div>  <div>month/year</div>	<u>0.7</u>	0.81	<u> </u>

Cashflow
situation

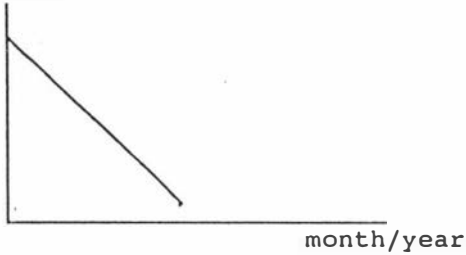
Grade of Membership
in the category of
"unfavorable cashflow
situation"

Your
own
response

Group
average

Your
revised
opinion
(if any)

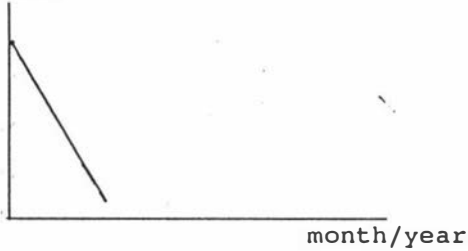
Bank
Balance



0.8

0.75

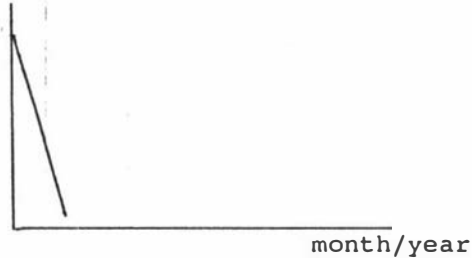
Bank
Balance



0.7

0.63

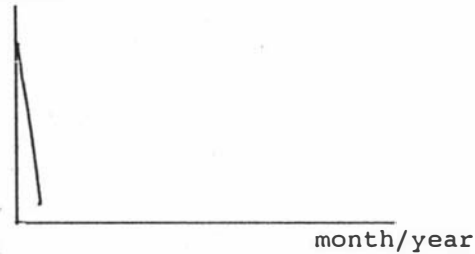
Bank
Balance



0.5

0.60

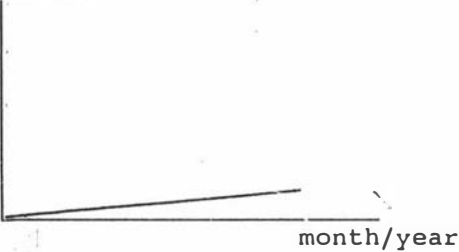
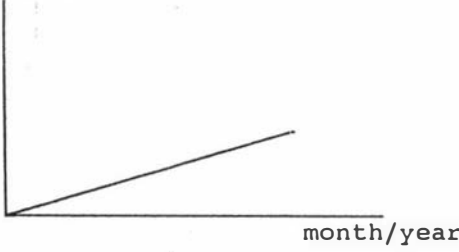
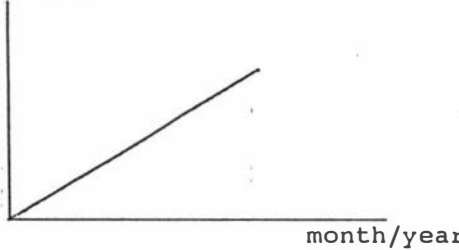
Bank
Balance



0.4

0.60

6 Each diagram below describes the movements of the market indicator in the previous twelve months. Please tell me the extent to which you believe each one of the following market situations described by the diagrams should be regarded as a favorable market situation by writing a number between 0 and 1 in the appropriate space provided:

Market situation	Grade of Membership in the category of "favorable market situation"		
	Your own response	Group average	Your revised opinion (if any)
<div>Market Indicator</div>  <div>month/year</div>	<u>0.3</u>	0.45	<u> </u>
<div>Market Indicator</div>  <div>month/year</div>	<u>0.5</u>	0.62	<u> </u>
<div>Market Indicator</div>  <div>month/year</div>	<u>0.5</u>	0.67	<u> </u>

Market
situation

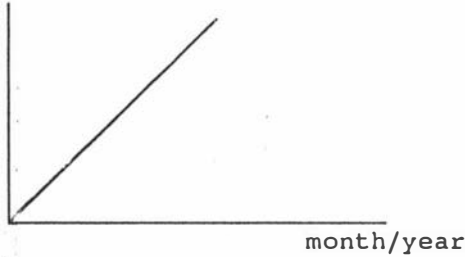
Grade of Membership
in the category of
"favorable market
situation"

Your
own
response

Group
average

Your
revised
opinion
(if any)

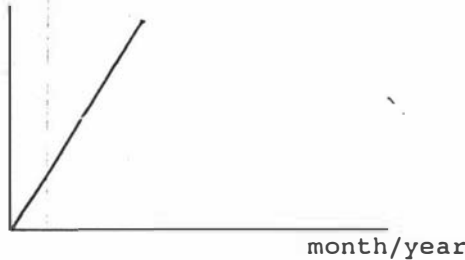
Market
Indicator



0.6

0.59

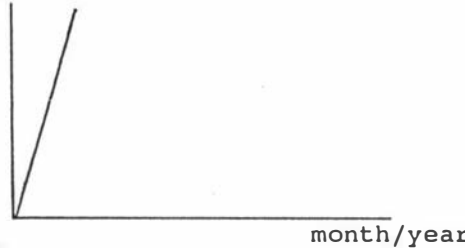
Market
Indicator



0.7

0.59

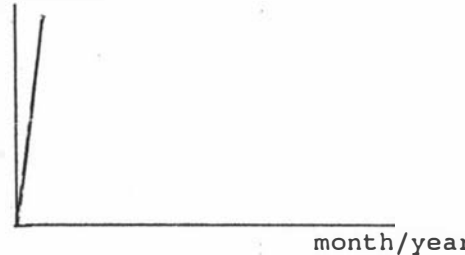
Market
Indicator



0.5

0.60

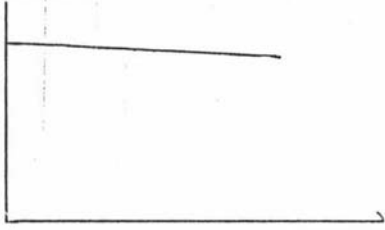
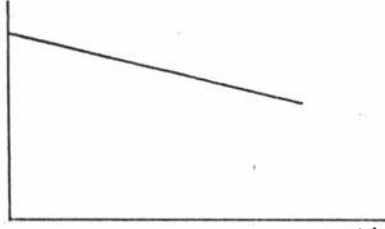
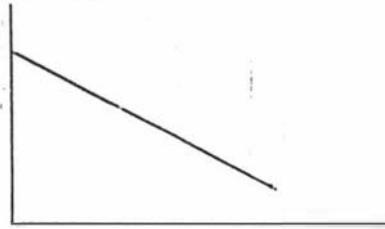
Market
Indicator



0.3

0.60

7 The diagrams below are similar to the diagrams in question 6. This time please tell me the extent to which you believe each one of the following market situations should be regarded as an unfavorable market situation by writing a number between 0 and 1 in the appropriate space provided:

Market situation	Grade of Membership in the category of "unfavorable market situation"		
	Your own response	Group average	Your revised opinion (if any)
<div>Market Indicator</div>  <div>month/year</div>	0.4	0.53	
<div>Market Indicator</div>  <div>month/year</div>	0.6	0.70	
<div>Market Indicator</div>  <div>month/year</div>	0.8	0.64	

Market
situation

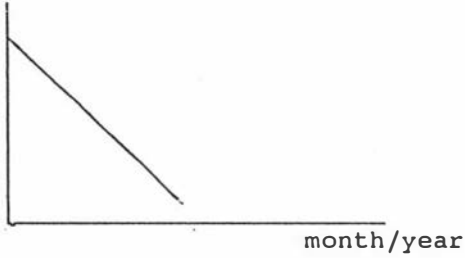
Grade of Membership
in the category of
"unfavorable market
situation"

Your
own
response

Group
average

Your
revised
opinion
(if any)

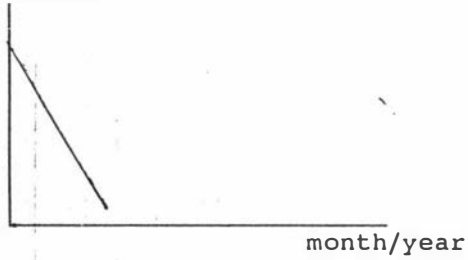
Market
Indicator



0.6

0.59

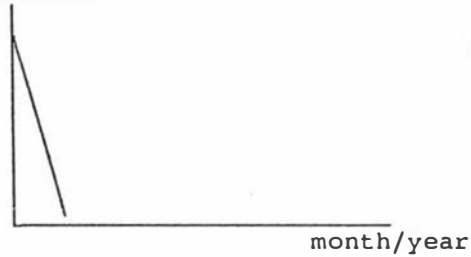
Market
Indicator



0.4

0.49

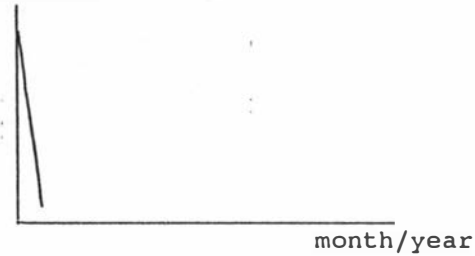
Market
Indicator



0.4

0.50

Market
Indicator



0.3

0.50

- 8 For this question, I would like you to look at the following percentages. The following percentages are obtained in this manner. First, the difference between the price of your product and the price of your strongest competitor's product is obtained. Then the difference is expressed as a percentage of the price of your product. Now, would you please tell me the extent to which you believe each one of the following percentages should belong to the category of "unfavorable" competitive situation by writing a number between 0 and 1 in the appropriate space provided?

Percentage that your price is above that of your competitors' price

Grade of Membership in the category of "unfavourable competitive situation"

	Your own response	Group average	Your revised opinion (if any)
1%	<u>0.3</u>	0.40	<u> </u>
2%	<u>0.7</u>	0.56	<u> </u>
3%	<u>0.9</u>	0.65	<u> </u>
4%	<u>1</u>	0.65	<u> </u>
5%	<u>1</u>	0.64	<u> </u>
6%	<u>1</u>	0.60	<u> </u>
7%	<u>1</u>	0.61	<u> </u>
8%	<u>1</u>	0.62	<u> </u>
9%	<u>1</u>	0.63	<u> </u>
10%	<u>1</u>	0.65	<u> </u>
11%	<u>1</u>	0.66	<u> </u>
12%	<u>1</u>	0.67	<u> </u>
13%	<u>1</u>	0.69	<u> </u>
14%	<u>1</u>	0.70	<u> </u>
15%	<u>1</u>	0.70	<u> </u>

- 9 The following percentages are obtained in like manner as the percentages in question 8. Now, please tell me the extent to which you believe each one of the following percentages should belong to the category of "favorable" competitive situation by writing a number between 0 and 1 in the appropriate space provided.

Percentage that your price is below that of your competitors' price

Grade of Membership in the category of "favourable competitive situation"

	Your own response	Group average	Your revised opinion (if any)
1%	<u>0.7</u>	0.75	<u> </u>
2%	<u>0.5</u>	0.60	<u> </u>
3%	<u>0.2</u>	0.47	<u> </u>
4%	<u>0</u>	0.30	<u> </u>
5%	<u>0</u>	0.32	<u> </u>
6%	<u>0</u>	0.36	<u> </u>
7%	<u>0</u>	0.34	<u> </u>
8%	<u>0</u>	0.34	<u> </u>
9%	<u>0</u>	0.35	<u> </u>
10%	<u>0</u>	0.34	<u> </u>
11%	<u>0</u>	0.33	<u> </u>
12%	<u>0</u>	0.32	<u> </u>
13%	<u>0</u>	0.31	<u> </u>
14%	<u>0</u>	0.31	<u> </u>
15%	<u>0</u>	0.30	<u> </u>

SECTION 3: DECISION MAKING RULES

INSTRUCTION: PLEASE READ THE FOLLOWING DECISION RULES AND INDICATE WHETHER YOU AGREE OR DISAGREE WITH THEM BY PUTTING A TICK IN THE APPROPRIATE BOX PROVIDED. IF YOU DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE PROVIDED UNDER EACH RULE.

- 10 If cashflow is favorable, then price change is small positive.

Agree ☐

Disagree ☐

IF DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE BELOW:

- 11 If cashflow is unfavorable, then price change is small negative.

Agree ☐

Disagree ☐

IF DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE BELOW:

- 12 If market outlook is favorable, then price change is small positive.

Agree ☐

Disagree ☐

IF DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE BELOW:

- 13 If market outlook is unfavorable, then price change is small negative.

Agree ☐

Disagree ☐

IF DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE BELOW:

- 14 If competition is weak , then price change is small positive.

Agree ☐

Disagree ☐

IF DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE BELOW:

- 15 If competition is strong, then price change is small negative.

Agree ☐

Disagree ☐

IF DISAGREE, PLEASE REWRITE THE RULE IN THE SPACE BELOW:

THANK YOU FOR YOUR KIND COOPERATION.

APPENDIX 4

RELATIONAL MATRICES FOR THE OTHER FIVE DECISION RULES

- 1 Relational matrix for unfavorable cashflow situation and small negative change in price:

		Unfavorable Cashflow						
		.27	.5	.87	.85	.81	.8	.77
small negative price change	.95	.95	.95	.95	.95	.95	.95	.95
	.91	.91	.91	.91	.91	.91	.91	.91
	.54	.73	.54	.54	.54	.54	.54	.54
	.31	.73	.5	.31	.31	.31	.31	.31
	.27	.73	.5	.27	.27	.27	.27	.27
	.28	.73	.5	.28	.28	.28	.28	.28
	.28	.73	.5	.28	.28	.28	.28	.28
	.25	.73	.5	.25	.25	.25	.25	.25
	.24	.73	.5	.24	.24	.24	.24	.24
	.24	.73	.5	.24	.24	.24	.24	.24
	.23	.73	.5	.23	.23	.23	.23	.23
	.23	.73	.5	.23	.23	.23	.23	.23
	.22	.73	.5	.22	.22	.22	.22	.23
	.21	.73	.5	.21	.21	.21	.21	.23
	.2	.73	.5	.2	.2	.2	.2	.23

- 2 Relational matrix for favorable market situation and small positive change in price:

		Favorable Market Situation						
		.38	.57	.74	.65	.65	.67	.67
small positive price change	.95	.95	.95	.95	.95	.95	.95	.95
	.89	.89	.89	.89	.89	.89	.89	.89
	.45	.62	.45	.45	.45	.45	.45	.45
	.38	.62	.43	.38	.38	.38	.38	.38
	.3	.62	.43	.3	.35	.35	.33	.33
	.29	.62	.43	.29	.35	.35	.33	.33
	.25	.62	.43	.26	.35	.35	.33	.33
	.22	.62	.43	.26	.35	.35	.33	.33
	.21	.62	.43	.26	.35	.35	.33	.33
	.2	.62	.43	.26	.35	.35	.33	.33
	.2	.62	.43	.26	.35	.35	.33	.33
	.2	.62	.43	.26	.35	.35	.33	.33
	.2	.62	.43	.26	.35	.35	.33	.33
	.2	.62	.43	.26	.35	.35	.33	.33

3 Relational matrix for unfavorable market situation and small negative change in price:

		Unfavorable Market Situation						
		.48	.76	.81	.75	.64	.65	.65
small negative price change	.95	.95	.95	.95	.95	.95	.95	.95
	.91	.91	.91	.91	.91	.91	.91	.91
	.54	.54	.54	.54	.54	.54	.54	.54
	.31	.52	.31	.31	.31	.36	.35	.35
	.27	.52	.27	.27	.27	.36	.35	.35
	.28	.52	.28	.28	.28	.36	.35	.35
	.28	.52	.28	.28	.28	.36	.35	.35
	.25	.52	.25	.25	.25	.36	.35	.35
	.24	.52	.24	.24	.25	.36	.35	.35
	.24	.52	.24	.24	.25	.36	.35	.35
	.23	.52	.24	.23	.25	.36	.35	.35
	.23	.52	.24	.23	.25	.36	.35	.35
	.22	.52	.24	.22	.25	.36	.35	.35
	.21	.52	.24	.21	.25	.36	.35	.35
	.2	.52	.24	.2	.25	.36	.35	.35

4 Relational matrix for weak competition and small positive change in price:

		Weak Competition						
		.75	.57	.42	.24	.36	.39	.37
small positive price change	.95	.95	.95	.95	.95	.95	.95	.95
	.89	.89	.89	.89	.89	.89	.89	.89
	.45	.45	.45	.58	.76	.64	.61	.63
	.38	.38	.43	.58	.76	.64	.61	.63
	.3	.3	.43	.58	.76	.64	.61	.63
	.29	.29	.43	.58	.76	.64	.61	.63
	.25	.25	.43	.58	.76	.64	.61	.63
	.22	.25	.43	.58	.76	.64	.61	.63
	.21	.25	.43	.58	.76	.64	.61	.63
	.2	.25	.43	.58	.76	.64	.61	.63
	.2	.25	.43	.58	.76	.64	.61	.63
	.2	.25	.43	.58	.76	.64	.61	.63
	.2	.25	.43	.58	.76	.64	.61	.63
	.2	.25	.43	.58	.76	.64	.61	.63
	.2	.25	.43	.58	.76	.64	.61	.63

Weak Competition

	.37	.38	.36	.33	.32	.31	.31	.3
small positive price change								
.95	.95	.95	.95	.95	.95	.95	.95	.95
.89	.89	.89	.89	.89	.89	.89	.89	.89
.45	.63	.62	.64	.67	.68	.69	.69	.7
.38	.63	.62	.64	.67	.68	.69	.69	.7
.3	.63	.62	.64	.67	.68	.69	.69	.7
.29	.63	.62	.64	.67	.68	.69	.69	.7
.25	.63	.62	.64	.67	.68	.69	.69	.7
.22	.63	.62	.64	.67	.68	.69	.69	.7
.21	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7
.2	.63	.62	.64	.67	.68	.69	.69	.7

5 Relational matrix for strong competition and small negative change in price:

Strong Competition

	.43	.51	.6	.61	.6	.67	.68
small negative price change							
.95	.95	.95	.95	.95	.95	.95	.95
.91	.91	.91	.91	.91	.91	.91	.91
.54	.57	.54	.54	.54	.54	.54	.54
.31	.57	.49	.4	.39	.4	.33	.32
.27	.57	.49	.4	.39	.4	.33	.32
.28	.57	.49	.4	.39	.4	.33	.32
.28	.57	.49	.4	.39	.4	.33	.32
.25	.57	.49	.4	.39	.4	.33	.32
.24	.57	.49	.4	.39	.4	.33	.32
.24	.57	.49	.4	.39	.4	.33	.32
.23	.57	.49	.4	.39	.4	.33	.32
.23	.57	.49	.4	.39	.4	.33	.32
.22	.57	.49	.4	.39	.4	.33	.32
.21	.57	.49	.4	.39	.4	.33	.32
.2	.57	.49	.4	.39	.4	.33	.32

Strong Competition

		.7	.71	.73	.75	.77	.79	.8	.8
small negative price change	.95	.95	.95	.95	.95	.95	.95	.95	.95
	.91	.91	.91	.91	.91	.91	.91	.91	.91
	.54	.54	.54	.54	.54	.54	.54	.54	.54
	.31	.31	.31	.31	.31	.31	.31	.31	.31
	.27	.3	.29	.27	.27	.27	.27	.27	.27
	.28	.3	.29	.28	.28	.28	.28	.28	.28
	.28	.3	.29	.28	.28	.28	.28	.28	.28
	.25	.3	.29	.27	.25	.25	.25	.25	.25
	.24	.3	.29	.27	.25	.24	.24	.24	.24
	.24	.3	.29	.27	.25	.24	.24	.24	.24
	.23	.3	.29	.27	.25	.23	.23	.23	.23
	.23	.3	.29	.27	.25	.23	.23	.23	.23
	.22	.3	.29	.27	.25	.23	.22	.22	.22
	.21	.3	.29	.27	.25	.23	.21	.21	.21
	.2	.3	.29	.27	.25	.23	.21	.2	.2

APPENDIX 5

/* LISTING PROGRAM FOR TZ.PRO */

```

/*****
*
* This is the top level program that drives the
* various subsystems.
*
*****/

```

predicates

```

getchoice(string)
start
choice(integer)
check(string)

```

goal

```

system("welcome.exe"),      % The program first invokes the
                             % welcome menu.
start,nl,nl,
write("\tThanks for using the system!"),nl,nl,
write("\tPlease press any key to exit."), readchar(_).

```

clauses

```

/*****
*
* The start predicate opens a window to allow a user
* to input his/her choices.
*
*****/

```

start:-

```

makewindow(1,7,7,"TZ      -      Marketing      Knowledge-based
System",0,0,25,80),nl,
write("\nPlease press 'H' for help."),nl,nl,
getchoice(Choice),
str_int(Choice,Choicei),
choice(Choicei),nl,nl,!,
clearwindow,nl,nl,
write("\tDo you wish to continue using the marketing
knowledge-based system?"),nl,
write("\t(please answer y/n)" ),readln(Respond),
Respond = "y",start.

```

start.

```

/*****
*
* The getchoice predicate prompts the user for the
* choice and provide help if the help key is pressed.
*
*****/

getchoice(Choice):-
    write("\tPlease select one of the following two
    facilities available in"),
    write("\n\tthis system, then press return."),
    write("\n\n\t1. Accounting System"),
    write("\n\n\t2. Marketing Knowledge-based System"),nl,nl,
    write("\tMy choice is: "),readln(Choice),
    check(Choice),
    Choice <> "H",nl,
    Choice <> "h",
    Choice <> "help",
    Choice <> "Help",!.

/*****
*
* The second getchoice predicate provides help
* to the user if the help key is pressed.
*
*****/

getchoice(Choice):-
    write("\n\tPlease enter either '1' or '2'."),nl,nl,
    getchoice(Choice),nl,nl.

/*****
*
* The check predicates check if the correct number
* has been keyed in.
*
*****/

check("1").
check("2").

```

```

/*****
*
*   The choice predicates invoke the accounting
*   subsystem and the marketing knowledge-based
*   system depending on the choice of the user.
*
*****/

choice(1):-
    system("acct.exe").

choice(2):-
    system("pmain.exe").

/*****
*
*   This is the end of "tz.pro".
*
*****/

```

/* PROGRAM LISTING FOR WELCOME.PRO */

```

/*****
*
* This subsystem welcomes the user and provides
* some help facilities.
*
*****/

```

```

predicates
    start
    getname(string)
    getdate(string)

```

```

goal
    start.

```

```

clauses

```

```

/*****
*
* The "start" clause produces the first window that
* welcomes the user and explains the nature of TZ.
*
*****/

```

```

start:-

```

```

    makewindow(1,7,7,"Welcome",0,0,25,80),nl,
    write("Key in 'H' for Help"),nl,
    getname(Name),
    write("\n\tWelcome    to    TZ,    the    Twilight    Zone,
",Name,"."),nl,
    write("\n\tI am a Knowledge-based System to help you
make \n\tbetter price decisions."),
    nl,
    write("\n\tI am created by Alan Tse at the Department of
Marketing, \n\tMassey Univesity."),nl,nl,
    getdate(Date),
    frontstr(2,Date,Day,Rest),
    frontstr(1,Rest,_,Rest1),
    frontstr(2,Rest1,Month,Rest2),
    frontstr(1,Rest2,_,Year),
    str_int(Day,D),
    str_int(Month,M),
    str_int(Year,Yr),
    date(Yr,M,D),
    system("music.exe"),      % "music.exe" is a subsystem that
                             % produces some music before the
                             % user goes into the main menu.

    write("\n\tPlease press any key to continue."),
    readln(_).

```

```

/*****
*
* The "getname" clause asks the user to input his/her
* name into the system so that TZ can address him/her
* in person. The clause also solicits if help is
* required by the user.
*
*****/

```

```

getname(Name):-
    write("\n\tPlease type your name then \n\t Enter.  "),
    readln(Name), Name <> "H",nl,
    Name <> "h",
    Name <> "help",
    Name <> "Help".

```

```

/*****
*
* The second "getname" clause provides help messages
* for the user if help is required.
*
*****/

```

```

getname(Name):-
    write("\n\tWhat you need to do now is to key in your
    name."),nl,
    getname(Name).

```

```

/*****
*
* The "getdate" clause asks the user to input today's
* date into the system so that TZ can update the
* accounting subsystem.
*
*****/

```

```

getdate(Date):-
    write("\n\tPlease key in today's date in the following
    format: "),nl,
    write("\n\t\tddd/mm/yyyy.  "),
    readln(Date), Date <> "H",
    Date <> "h",
    Date <> "help",
    Date <> "Help".

```

```

/*****
*
* The second "getdate" clause provides help messages
* for the user if help is required.
*
*****/

```

```

getdate(Date):-

```

```
        write("\n\tPlease key in today's date in the format
specified:"),nl,
write("\n\tFor example, if today is 14 June 1991, then
key in:"),nl,
write("\n\t\t14/06/1991.  "),nl,
getdate(Date),nl.
```

```
/*****
*
* This is the end of "welcome.pro".
*
*****/
```

```
/* LISTING PROGRAM OF MUSIC.PRO */
```

```

/*****
*
* This program produces some music (jack and jill)
* after the welcome menu and before the user goes
* into the main menu.
*
*****/

```

```
domains
```

```
direction=up;down
```

```
predicates
```

```
jack_and_jill(direction,integer)
```

```
goal
```

```
jack_and_jill(up,500).
```

```
clauses
```

```

jack_and_jill(up,F) :-
    F<5000,! ,sound(12,F),F1=F+200,jack_and_jill(up,F1).
jack_and_jill(up,F) :-
    jack_and_jill(down,F).
jack_and_jill(down,F) :-
    F>500,! ,sound(12,F),F1=F-200,jack_and_jill(down,F1).
jack_and_jill(down,_).

```

```

/*****
*
* This is the end of "music.pro".
*
*****/

```



```
/* LISTING PROGRAM FOR ACCT.PRO */
```

```

/*****
*
* This program controls the execution of the
* transaction processing subsystem and the financial
* reporting system. It allows the user to make a
* choice between the use of the two subsystems.
*
*****/

```

domains

```
x = integer
```

predicates

```
choice(x)
```

```
start
```

```
getchoice(string)
```

goal

```
start,nl,nl,
```

```
write("\tThanks for using the accounting system."),nl,nl,
```

```
write("\tPlease press any key to continue."), readchar(_).
```

clauses

```

/*****
*
* The start predicate opens a window to allow a user
* to input his/her choices.
*
*****/

```

start:-

```
makewindow(1,7,7,"Accounting System",0,0,25,80),nl,
```

```
write("\nPlease press 'H' for help."),nl,nl,
```

```
getchoice(Choice),
```

```
str_int(Choice,Choicei),
```

```
choice(Choicei),nl,nl,
```

```
write("\tDo you wish to continue? "),nl,nl,
```

```
write("\t(please answer y to continue to use the
accounting system,") ,nl,
```

```
write("\t please type n to go back to the marketing
knowledge-based system.)"),nl,
```

```
readln(Response),
```

```
Response = "y", start.
```

start.

```

/*****
*
* The getchoice predicate prompts the user for the
* choice and provide help if the help key is pressed.
*
*****/

getchoice(Choice):-
    write("\tPlease select one of the following two
    facilities available in"),
    write("\n\tthis accounting system, then press return."),
    write("\n\n\t1. Input Accounting Data"),
    write("\n\n\t2. Output Financial Statements"),nl,nl,
    write("\tMy choice is: "),readln(Choice),
    Choice <> "H",nl,
    Choice <> "h",
    Choice <> "help",
    Choice <> "Help",!.

/*****
*
* The second getchoice predicate provides help
* to the user if the help key is pressed.
*
*****/

getchoice(Choice):-
    write("\n\tPlease enter either '1' or '2'."),nl,nl,
    getchoice(Choice),nl,nl.

/*****
*
* The choice predicates invoke the transaction
* processing system and the financial reporting
* subsystem depending on the choice of the user.
*
*****/

choice(1):-
    system("1acct.exe").

choice(2):-
    system("2acct.exe").

/*****
*
* This is the end of "acct.pro".
*
*****/

```

/* PROGRAM LISTING FOR 1ACCT.PRO */

```
/* *****  
*  
* This subsystem accepts input transaction data  
* from the user and updates the accounting records.  
*  
* *****
```

domains

```
act = integer  
amount = real  
file = acctdata; input1; output1; input2; output2
```

database

```
subtotal(act,amount)  
mbalance(integer,integer,integer,real)
```

predicates

```
process  
update_mbalance(string,integer,integer,real)  
balance  
monthly_balance  
readbal  
readmbal  
repfile(file)  
getdebit(string)  
getcredit(string)  
quit(string)  
check(string)  
match(string,integer)
```

goal

```
openread(input1,"balance.dat"),  
readdevice(input1),  
readbal,  
closefile(input1),  
openread(input2,"mbalance.dat"),  
readdevice(input2),  
readmbal,  
closefile(input2),  
process,  
openwrite(output1,"balance.dat"),  
writedevic(output1),  
balance,  
closefile(output1),  
openwrite(output2,"mbalance.dat"),  
writedevic(output2),  
monthly_balance,  
closefile(output2).
```

clauses

```

/*****
*
* The process predicate allows the user to input
* details of a transaction.
*
*****/

```

process:-

```

makewindow(1,7,7,"Input Accounting Data",0,0,25,80),
write("Please type 'H' for help."),nl,nl,
getdebit(Dbacno),nl,      % Dbacno is the account number to be
                           % debited.

str_int(Dbacno,Dbacno1),
write("\tPlease input the date of the transaction
(DD/MM/YYYY): "),
readln(Date),nl,
write("\tThe amount to be debited is: $"),readreal(Debit),
getcredit(Cracno),nl,     % Cracno is the account number to be
                           % credited.

str_int(Cracno, Cracno1),
subtotal(Dbacno1,Balance),
Nbalance = Balance + Debit, % Nbalance means new balance.
assertz(subtotal(Dbacno1,Nbalance)),
retract(subtotal(Dbacno1,Balance)),
subtotal(Cracno1,Balance1),
Nbalance1 = Balance1 - Debit,
assertz(subtotal(Cracno1,Nbalance1)),
retract(subtotal(Cracno1,Balance1)),
update_mbalance(Date,Dbacno1,Cracno1,Debit),
openappend(acctdata,"acct.dat"),
writedevise(acctdata),
                           % the new transaction is stored
                           % in the "acct.dat" data file.

write(Date,Dbacno1,Cracno1,Debit,"\n"),
closefile(acctdata),!,process.

```

process.

```

/*****
*
* The getdebit predicate gets the account number
* of the account to be debited.
*
*****/

```

getdebit(Dbacno):-

```

write("\n\tPlease key in the number corresponding to the
account you"),nl,
write("\twant to debit:"),nl,nl,
write("\t100.  Cash account"),nl,
write("\t105.  Stock account"),nl,
write("\t110.  Capital account"),nl,
write("\t150.  Current asset accounts"),nl,
write("\t200.  Fixed asset accounts"),nl,

```

```

write("\t250.    Current liabilities account"),nl,
write("\t300.    Fixed liabilities account"),nl,
write("\t350.    Expenses accounts"),nl,
write("\t400.    Sales account"),nl,
write("\t700.    Purchases account"),nl,
write("\t999.    Quit"),nl,
write("\n\tAccount No: "),readln(Dbacno),
check(Dbacno),
Dbacno <> "H",nl,
Dbacno <> "h",
Dbacno <> "help",
Dbacno <> "Help",!,
quit(Dbacno).

/*****
*
* The second getdebit predicate provides some help
* to the user if the help key is pressed.
*
*****/

getdebit(Dbacno):-
    write("\n\tWhat you need to do now is to key in the
    number"),nl,
    write("\n\tcorresponding to the account you want to
    debit."),nl,
    getdebit(Dbacno),nl.

/*****
*
* The getcredit predicate gets the account number
* of the account to be credited.
*
*****/

getcredit(Cracno):-
    write("\n\tPlease key in the corresponding account to be
    credited:"),nl,
    write("\t100.    Cash account"),nl,
    write("\t105.    Stock account"),nl,
    write("\t110.    Capital account"),nl,
    write("\t150.    Current asset accounts"),nl,
    write("\t200.    Fixed asset accounts"),nl,
    write("\t250.    Current liabilities account"),nl,
    write("\t300.    Fixed liabilities account"),nl,
    write("\t350.    Expenses accounts"),nl,
    write("\t400.    Sales account"),nl,
    write("\t700.    Purchases account"),nl,
    write("\n\tAccount No: "),readln(Cracno),nl,
    check(Cracno),
    Cracno <> "H",nl,
    Cracno <> "h",
    Cracno <> "help",
    Cracno <> "Help",!.

```

```

/*****
*
* The second getcredit predicate provides some help
* to the user if the help key is pressed.
*
*****/

getcredit(Crarno):-
    write("\n\tWhat you need to do now is to key in the
    number"),nl,
    write("\n\tcorresponding to the account you want to
    crebit."),nl,
    getcredit(Crarno),nl.

/*****
*
* The check predicates are used to check if the
* correct account number has been pressed.
*
*****/

check("100").
check("105").
check("110").
check("150").
check("200").
check("250").
check("300").
check("350").
check("400").
check("700").
check("999").

/*****
*
* The quits predicate ars to check if the user really
* wants to quit the accounting subsystem.
*
*****/

quit("999"):-
    write("\n\tDo you really want to quit? (please answer
    y/n): "),
    readln(Y), Y = "y",fail.

quit(Dbacno):-
    Dbacno <> "999".

```

```

/*****
*
* The readbal predicate reads the records of the
* "balance.dat" file and store in the internal
* database of lacct.pro for later use. The
* "balance.dat" file contains balances of the
* the accounts in books.
*
*****/

readbal:-
    repfile(input1),
    readln(LN),
    frontstr(3, LN, Act, Balance),
    str_int(Act, Act1),
    str_real(Balance, Balance1),
    assertz(subtotal(Act1, Balance1)), % subtotal is the name of
                                     % the internal database.
    fail.

readbal.

/*****
*
* The readmbal predicate reads the records of the
* "mbalance.dat" file and store in the internal
* database of lacct.pro for later use. The
* "mbalance.dat" file contains monthly balances of
* the accounts in the books.
*
*****/

readmbal:-
    repfile(input2),
    readln(LN),
    frontstr(4, LN, Year, Rest),
    frontstr(3, Rest, Month, Rest1),
    frontstr(3, Rest1, Act, Balance),
    str_int(Year, Year1),
    match(Month, Month1),
    str_int(Act, Act1),
    str_real(Balance, Balance1),
    assertz(mbalance(Year1, Month1, Act1, Balance1)),
                                     % mbalance is the name of the
                                     % second internal database.
    fail.

readmbal.

```

```

/*****
*
* The match predicates are used to transform the
* month strings in the "mbalance.dat" file into month
* integers.
*
*****/

match("jan",1).
match("feb",2).
match("mar",3).
match("apr",4).
match("may",5).
match("jun",6).
match("jul",7).
match("aug",8).
match("sep",9).
match("oct",10).
match("nov",11).
match("dec",12).

/*****
*
* The repfile predicate enables the records in a
* file to be read one by one until the end of file
* is reached.
*
*****/

repfile(_).

repfile(F):-
    not(eof(F)),repfile(F).

/*****
*
* The update_mbalance predicate updates the monthly
* balances in the "mbalance.dat" file.
*
*****/

update_mbalance(Date,Dbacno1,Cracno1,Debit):-
    frontstr(3,Date,_,Rest),
    frontstr(2,Rest,Month,Rest1),
    frontstr(1,Rest1,_,Year),
    str_int(Month,Month1),
    str_int(Year,Year1),
    mbalance(Year1,Month1,Dbacno1,Balance1),
    Nbalance1 = Balance1 + Debit,
    assertz(mbalance(Year1,Month1,Dbacno1,Nbalance1)),
    retract(mbalance(Year1,Month1,Dbacno1,Balance1)),
    mbalance(Year1,Month1,Cracno1,Balance2),
    Nbalance2 = Balance2 - Debit,
    assertz(mbalance(Year1,Month1,Cracno1,Nbalance2)),
    retract(mbalance(Year1,Month1,Cracno1,Balance2)).

```



```

/*****
*
* The second update_mbalance predicate adds a monthly
* balance into the "mbalance.dat" file if the record
* is a new one and if the account to be debited is
* not the same as the account to be credited.
*
*****/

```

```

update_mbalance(Date,Dbacno1,Cracno1,Debit):-
    Dbacno1 <> Cracno1,!,
    frontstr(3,Date,_,Rest),
    frontstr(2,Rest,Month,Rest1),
    frontstr(1,Rest1,_,Year),
    str_int(Month,Month1),
    str_int(Year,Year1),
    Credit = -Debit,
    assertz(mbalance(Year1,Month1,Dbacno1,Debit)),
    assertz(mbalance(Year1,Month1,Cracno1,Credit)).

```

```

/*****
*
* The third update_mbalance predicate adds a monthly
* balance into the "mbalance.dat" file if the record
* is a new one and if the account to be debited is
* the same as the account to be credited.
*
*****/

```

```

update_mbalance(Date,Dbacno1,Cracno1,_):-
    Dbacno1 = Cracno1,!,
    frontstr(3,Date,_,Rest),
    frontstr(2,Rest,Month,Rest1),
    frontstr(1,Rest1,_,Year),
    str_int(Month,Month1),
    str_int(Year,Year1),
    assertz(mbalance(Year1,Month1,Dbacno1,0)).

```

```

/*****
*
* The balance predicate writes the balances into the
* "balance.dat" file.
*
*****/

```

```

balance:-
    subtotal(Act,Balance),
    write(Act,Balance),nl,
    fail.

```

```

balance.

```

```

/*****
*
*   The monthly_balance predicate writes the monthly
*   balances into the "mbalance.dat" file.
*
*****/

```

```

monthly_balance:-
    mbalance(Year,Month,Act,Balance),
    match(Month1,Month),
    write(Year,Month1,Act,Balance),nl,
    fail.

```

```

monthly_balance.

```

```

/*****
*
*   This is the end of "lacct.pro".
*
*****/

```

/* LISTING PROGRAM FOR 2ACCT.PRO */

```

/*****
*
* This subsystem is responsible for outputting
* financial statements.
*
*****/

```

domains

```

    file = input
    act, amt = integer
    ln = string

```

database

```

    subtotal(act,amt)

```

predicates

```

    repfile(file)
    start
    final1(amt,amt,amt,amt,amt,amt,amt,amt,amt)
    final2(amt,amt,amt,amt,amt,amt,amt)
    get_close_stock(string)

```

goal

```

    openread(input,"balance.dat"),
    readdevice(input),
    start,
    closefile(input),
    makewindow(1,7,7,"Financial Statements",0,0,25,80),nl,
    write("\nPlease press 'H' for help."),nl,nl,
    subtotal(100,Acct100), subtotal(105,Acct105),
    subtotal(110,Acct110), subtotal(150,Acct150),
    subtotal(200,Acct200), subtotal(250,Acct250),
    subtotal(300,Acct300), subtotal(350,Acct350),
    subtotal(400,Acct400), subtotal(700,Acct700),
    get_close_stock(Stock),
    str_int(Stock, Temp2),
    Temp1 = Acct105 + Acct700,
    Temp3 = Temp1 - Temp2,
    Temp4 = Acct400 - Temp3,
    Temp5 = Temp4 - Acct350,
    final1(Acct105,Acct400,Acct700,Temp1,Temp2,Temp3,Temp4,
           Acct350,Temp5),
    write("Press any key for the balance
    sheet."),readln(_),nl,nl,
    final2(Acct110,Temp5,Acct300,Acct250,Acct150,Acct100,Acct200).

```

clauses

```

/*****
*
* The predicate "start" reads the account balances
* in the "balance.dat" file and stores all the
* balances in the internal database "subtotal"
*
*****/

start:-
    repfile(input),
    readln(LN),
    frontstr(3, LN, Act, Balance),
    str_int(Act, Act1),
    str_real(Balance, Balance1),
    assertz(subtotal(Act1, Balance1)),
    fail.

start.

/*****
*
* The repfile predicate enables the records in a
* file to be read one by one until the end of file
* is reached.
*
*****/

repfile(_).

repfile(F):-
    not(eof(F)), repfile(F).

/*****
*
* The repfile predicate gets the closing stock from
* user.
*
*****/

get_close_stock(Stock):-
    write("\n\tTo produce the financial statements, you need
    to tell"),
    write("\n\tme the amount of the closing stock."), nl,
    write("\n\tWhat is the amount of the closing stock? $"),
    readln(Stock), nl, nl,
    Stock <> "H", nl,
    Stock <> "h",
    Stock <> "help",
    Stock <> "Help", !.

```

```

/*****
*
* The second get_close_stock predicate provides help
* to the user if the help key is pressed.
*
*****/

get_close_stock(Stock):-
    write("\n\tWhat you need to do now is to take your stock
    and"),
    write("\n\tkey in the amount of closing stock."),nl,
    get_close_stock(Stock),nl.

/*****
*
* The final1 predicate outputs the trading and profit
* and loss account.
*
*****/

final1(Acct105,Acct400,Acct700,Temp1,Temp2,Temp3,Temp4,
    Acct350,Temp5):-
    date(YR,MN,DD),
    write("\n\tThe following is the profit and loss
    account:"),nl,nl,
    write("\t
                                Alan Tse Co. Ltd.
    "),nl,
    write("\t Profit and Loss Account for the Year Ended
    ",DD,"-",MN,"-",YR," ").nl,
    write("\t-----
    -----"),nl,nl,
    write("\tSales
    $",Acct400),nl,
    write("\tLess
    $",Acct105),nl,
    write("\t
    $",Acct700),nl,
    write("\t
    -----
    -----"),nl,
    write("\t
    $",Temp1),nl,
    write("\t
    $",Temp2),nl,
    write("\t
    -----
    -----"),nl,
    write("\t
    Cost of Goods Sold
    $",Temp3),nl,
    write("\t
    -----
    -----"),nl,
    write("\tProfit
    $",Temp4),nl,
    write("\tLess
    $",Acct350),nl,
    write("\t
    -----
    -----"),nl,
    write("\tNet
    Profit

```

```

/*****
*
*   The final predicate outputs the balance sheet
*
*****/

```

180

```

Temp10 = Temp9 + Acct200,
write("\t
$,Temp10),nl,
write("\t
====="),nl,nl,
write("\tPress any key to continue."),readchar(_).

/*****
*
* This is the end of "2acct.pro".
*
*****/

```

```

/* LISTING PROGRAM FOR PRICE.PRO */

/*****
*
* This subsystem draws the price movement diagram.
*
*****/

constants
    bgi_Path = "c:\\prolog\\bgi"    % This constant specifies the
                                     % path for the driver.

domains
    file = input
    act = integer
    y,price = real
    pricelist = price*
    month = string
    monlist = month*

database
    data(integer,price)

predicates
    readfile(file)
    drawcurve(y,price)
    repfile(file)
    scale(y,price)
    max(price)
    min(price)
    member(price,pricelist)
    match(integer,monlist)
    labely(price,price,price,price,price,price,price,price,price,price,
           price,price,price)
    get12

goal
    openread(input,"price.dat"),
    readdevice(input),
    readfile(input),
    closefile(input),
    get12,
    scale(Scale,MinPrice),
    drawcurve(Scale,MinPrice).

```


clauses

```

/*****
*
* The readfile predicate reads the market indicates
* from the "price.dat" file and select the most
* recent twelve observations to draw the curve.
*
*****/

readfile(input):-
    repfile(input),
    readln(LN),
    frontstr(2, LN, Month, Rest),
    frontstr(4, Rest, Year, Price),
    str_int(Month, Month1),
    str_int(Year, Year1),
    date(Yr, Mn, _),
    (Yr-Year1)*12 + Mn -Month1 <13,
    Count = 13 - ((Yr-Year1)*12 + Mn -Month1),
    str_real(Price, Price1),
    assertz(data(Count, Price1)),
    fail.

readfile(_).

/*****
*
* The repfile predicate enables the records in a
* file to be read one by one until the end of file
* is reached.
*
*****/

repfile(_).

repfile(F):-
    not(eof(F)), repfile(F).

/*****
*
* The get12 predicate checks if there are twelve
* observations.
*
*****/

get12:-
    data(12, _).
```

```

/*****
*
* The second get12 predicate outputs an error message
* if there are not enough observations.
*
*****/

get12:-
    write("\n\tThere is something wrong with your data.
    Check your"),nl,
    write("\n\tdata first before drawing the curve."),nl,fail.

/*****
*
* The predicate scale is used to determine the scale
* for drawing the curve.
*
*****/

scale(Scale,MinPrice):-
    max(MaxPrice),
    min(MinPrice),
    Scale = (MaxPrice - MinPrice)/300.

/*****
*
* The max predicate is used to find out which
* observation is the largest one.
*
*****/

max(MaxPrice):-
    data(1,Price1),data(2,Price2),
    data(3,Price3),data(4,Price4),
    data(5,Price5),data(6,Price6),
    data(7,Price7),data(8,Price8),
    data(9,Price9),data(10,Price10),
    data(11,Price11),data(12,Price12),
    member(MaxPrice,[Price1,Price2,Price3,Price4,Price5,Price6,
    Price7,Price8,Price9,Price10,Price11,Price12]),
    MaxPrice >= Price1, MaxPrice >= Price2, MaxPrice >=
    Price3, MaxPrice >= Price4, MaxPrice >= Price5,
    MaxPrice >= Price6, MaxPrice >= Price7, MaxPrice >=
    Price8, MaxPrice >= Price9,
    MaxPrice >= Price10, MaxPrice >= Price11, MaxPrice >=
    Price12,!.
```

```

/*****
*
* The max predicate is used to find out which
* observation is the smallest one.
*
*****/

min(MinPrice):-
    data(1,Price1),data(2,Price2),
    data(3,Price3),data(4,Price4),
    data(5,Price5),data(6,Price6),
    data(7,Price7),data(8,Price8),
    data(9,Price9),data(10,Price10),
    data(11,Price11),data(12,Price12),
    member(MinPrice,[Price1,Price2,Price3,Price4,Price5,
        Price6,Price7,Price8,Price9,Price10,Price11,Price12]),
    MinPrice <= Price1, MinPrice <= Price2, MinPrice <=
    Price3, MinPrice <= Price4, MinPrice <= Price5,
    MinPrice <= Price6, MinPrice <= Price7, MinPrice <=
    Price8, MinPrice <= Price9,
    MinPrice <= Price10, MinPrice <= Price11, MinPrice <=
    Price12,! .

/*****
*
* The member predicate is to determine if a given
* element is the member of a list.
*
*****/

member(Y1,[Y1|_]).

member(Y1,[_|Tail]):-
    member(Y1,Tail).

/*****
*
* The drawcurve predicate draws the price curve.
*
*****/

drawcurve(Scale,MinPrice):-
    data(1,Price1),data(2,Price2),
    data(3,Price3),data(4,Price4),
    data(5,Price5),data(6,Price6),
    data(7,Price7),data(8,Price8),
    data(9,Price9),data(10,Price10),
    data(11,Price11),data(12,Price12),

    % The Sprices' are the scaled price data.

    SPrice1 = 300 - (Price1-MinPrice)/Scale, SPrice2 = 300 -
    (Price2-MinPrice)/Scale,
    SPrice3 = 300 - (Price3-MinPrice)/Scale, SPrice4 = 300 -
    (Price4-MinPrice)/Scale,
    SPrice5 = 300 - (Price5-MinPrice)/Scale, SPrice6 = 300 -

```

```

(PPrice6-MinPrice)/Scale,
SPrice7 = 300 - (Price7-MinPrice)/Scale, SPrice8 = 300 -
(PPrice8-MinPrice)/Scale,
SPrice9 = 300 - (Price9-MinPrice)/Scale, SPrice10 = 300 -
(PPrice10-MinPrice)/Scale,
SPrice11 = 300 - (Price11-MinPrice)/Scale, SPrice12 = 300
- (Price12-MinPrice)/Scale,!,
date(_,MM,_),
    match(MM,[M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12]),
                                % The match predicate
                                % determines how the x-axis
                                % should be labelled.
detectgraph(Gdriver, Gmode),    % Initialize the graph mode
initgraph(Gdriver, Gmode,_,_, bgi_Path),
line(20,150,719,150),          % Draw the X-axis.
line(20,20,20,300),            % Draw the Y-axis.
line(70,SPrice1,120,SPrice2),   % The drawing of the curve
                                % starts here.

line(120,SPrice2,170,SPrice3),
line(170,SPrice3,220,SPrice4),
line(220,SPrice4,270,SPrice5),
line(270,SPrice5,320,SPrice6),
line(320,SPrice6,370,SPrice7),
line(370,SPrice7,420,SPrice8),
line(420,SPrice8,470,SPrice9),
line(470,SPrice9,520,SPrice10),
line(520,SPrice10,570,SPrice11),
line(570,SPrice11,620,SPrice12),!,
moveto(100,1),                  % output the title of graph
outtext("This graph shows the price movements for the
past year."),
moveto(100,10),
outtext("The y-axis is the market indicator."),
% The labelling of the x-axis starts here.

moveto(70,150),
outtext(M1),
moveto(120,150),
outtext(M2),
moveto(170,150),
outtext(M3),
moveto(220,150),
outtext(M4),
moveto(270,150),
outtext(M5),
moveto(320,150),
outtext(M6),
moveto(370,150),
outtext(M7),
moveto(420,150),
outtext(M8),
moveto(470,150),
outtext(M9),
moveto(520,150),
outtext(M10),
moveto(570,150),

```

```

    outtext(M11),
    moveto(620,150),
    outtext(M12),!,
    labely(SPrice1,SPrice2,SPrice3,SPrice4,SPrice5,SPrice6,
           SPrice7,SPrice8,SPrice9,SPrice10,SPrice11,SPrice12),
    moveto(300,337),
    outtext("Please press any key to continue."),readchar(_),
    closegraph.

/*****
*
* The match predicates are used to determine the way
* the x-axis should be labelled.
*
*****/

match(1,["jan","feb","mar","apr","may","jun","jul","aug",
        "sep","oct","nov","dec"]).
match(2,["feb","mar","apr","may","jun","jul","aug","sep",
        "oct","nov","dec","jan"]).
match(3,["mar","apr","may","jun","jul","aug","sep","oct",
        "nov","dec","jan","feb"]).
match(4,["apr","may","jun","jul","aug","sep","oct","nov",
        "dec","jan","feb","mar"]).
match(5,["may","jun","jul","aug","sep","oct","nov","dec",
        "jan","feb","mar","apr"]).
match(6,["jun","jul","aug","sep","oct","nov","dec","jan",
        "feb","mar","apr","may"]).
match(7,["jul","aug","sep","oct","nov","dec","jan","feb",
        "mar","apr","may","jun"]).
match(8,["aug","sep","oct","nov","dec","jan","feb","mar",
        "apr","may","jun","jul"]).
match(9,["sep","oct","nov","dec","jan","feb","mar","apr",
        "may","jun","jul","aug"]).
match(10,["oct","nov","dec","jan","feb","mar","apr","may",
        "jun","jul","aug","sep"]).
match(11,["nov","dec","jan","feb","mar","apr","may","jun",
        "jul","aug","sep","oct"]).
match(12,["dec","jan","feb","mar","apr","may","jun","jul",
        "aug","sep","oct","nov"]).

/*****
*
* The labely predicate is used to label the y-axis.
*
*****/

labely(SPrice1,SPrice2,SPrice3,SPrice4,SPrice5,SPrice6,
       SPrice7,SPrice8,SPrice9,SPrice10,SPrice11,SPrice12):-
    data(1,Price1),data(2,Price2),
    data(3,Price3),data(4,Price4),
    data(5,Price5),data(6,Price6),
    data(7,Price7),data(8,Price8),
    data(9,Price9),data(10,Price10),
    data(11,Price11),data(12,Price12),
    moveto(5,SPrice1),
    % label the Y-axis

```

```

str_int(Prices1,Price1),
outtext(Prices1),
moveto(5,SPrice2),
str_int(Prices2,Price2),
outtext(Prices2),
moveto(5,SPrice3),
str_int(Prices3,Price3),
outtext(Prices3),
moveto(5,SPrice4),
str_int(Prices4,Price4),
outtext(Prices4),
moveto(5,SPrice5),
str_int(Prices5,Price5),
outtext(Prices5),
moveto(5,SPrice6),
str_int(Prices6,Price6),
outtext(Prices6),
moveto(5,SPrice7),
str_int(Prices7,Price7),
outtext(Prices7),
moveto(5,SPrice8),
str_int(Prices8,Price8),
outtext(Prices8),
moveto(5,SPrice9),
str_int(Prices9,Price9),
outtext(Prices9),
moveto(5,SPrice10),
str_int(Prices10,Price10),
outtext(Prices10),
moveto(5,SPrice11),
str_int(Prices11,Price11),
outtext(Prices11),
moveto(5,SPrice12),
str_int(Prices12,Price12),
outtext(Prices12).

```

```

/*****
*
*   This is the end of "price.pro".
*
*****/

```

```
/* LISTING PROGRAM FOR CASH.PRO */
```

```

/*****
*
*   This subsystem draws the cashflow movement diagram.
*
*****/

```

```
constants
```

```

    bgi_Path = "c:\\prolog\\bgi"    % This constant specifies the
                                     % path for the driver.

```

```
domains
```

```

    file = input
    act = integer
    y,amt = real
    amtlist = amt*
    month = string
    monlist = month*

```

```
database
```

```
    subtotal(integer,amt)
```

```
predicates
```

```

    readfile(file)
    drawcurve(y,amt)
    repfile(file)
    scale(y,amt)
    max(amt)
    min(amt)
    member(amt,amtlist)
    match(integer,monlist)
    labely(amt,amt,amt,amt,amt,amt,amt,amt,amt,amt,amt,amt)
    get12
    match1(string,integer)

```

```
goal
```

```

    openread(input,"mbalance.dat"),
    readdevice(input),
    readfile(input),
    closefile(input),
    get12,
    scale(Scale,MinAmt),
    drawcurve(Scale,MinAmt).

```

clauses

```

/*****
*
* The readfile predicate reads the bank balance data
* from the "mbalance.dat" file and select the most
* recent twelve observations to draw the curve.
*
*****/

readfile(input):-
    repfile(input),
    readln(LN),
    frontstr(4, LN, Year, Rest),
    frontstr(3, Rest, Month, Rest1),
    frontstr(3, Rest1, Act, Amt),
    str_int(Year, Year1),
    match1(Month, Month1),
    str_int(Act, Act1),
    str_real(Amt, Amt1),
    Act1 = 100,      % "100" is the account number assigned to
                    % bank balances.
    date(Yr, Mn, _),
    (Yr-Year1)*12 + Mn -Month1 <13,
    Count = 13 - ((Yr-Year1)*12 + Mn -Month1),
    assertz(subtotal(Count, Amt1)),
    fail.

readfile(_).

/*****
*
* The repfile predicate enables the records in a
* file to be read one by one until the end of file
* is reached.
*
*****/

repfile(_).

repfile(F):-
    not(eof(F)), repfile(F).

/*****
*
* The match1 predicates are used to transform the
* month strings in the "mbalance.dat" file into month
* integers.
*
*****/

match1("jan", 1).
match1("feb", 2).
match1("mar", 3).
match1("apr", 4).
match1("may", 5).

```



```

match1("jun",6).
match1("jul",7).
match1("aug",8).
match1("sep",9).
match1("oct",10).
match1("nov",11).
match1("dec",12).

/*****
*
*   The get12 predicate checks if there are twelve
*   observations.
*
*****/

get12:-
    subtotal(12,_).

/*****
*
*   The second get12 predicate outputs an error message
*   if there are not enough observations.
*
*****/

get12:-
    write("\n\tThere is something wrong with your data.
    Check your"),nl,
    write("\n\tcurve first before drawing the
    curve."),nl,fail.

/*****
*
*   The predicate scale is used to determine the scale
*   for drawing the curve.
*
*****/

scale(Scale,MinAmt):-
    max(MaxAmt),
    min(MinAmt),
    Scale = (MaxAmt - MinAmt)/300.

/*****
*
*   The max predicate is used to find out which
*   observation is the largest one.
*
*****/

max(MaxAmt):-
    subtotal(1,Amt1),subtotal(2,Amt2),
    subtotal(3,Amt3),subtotal(4,Amt4),
    subtotal(5,Amt5),subtotal(6,Amt6),
    subtotal(7,Amt7),subtotal(8,Amt8),
    subtotal(9,Amt9),subtotal(10,Amt10),

```

```

    subtotal(11,Amt11),subtotal(12,Amt12),
    member(MaxAmt,[Amt1,Amt2,Amt3,Amt4,Amt5,Amt6,Amt7,Amt8,
    Amt9,Amt10,Amt11,Amt12]),
    MaxAmt >= Amt1, MaxAmt >= Amt2, MaxAmt >= Amt3, MaxAmt >=
    Amt4, MaxAmt >= Amt5,
    MaxAmt >= Amt6, MaxAmt >= Amt7, MaxAmt >= Amt8, MaxAmt >=
    Amt9,
    MaxAmt >= Amt10, MaxAmt >= Amt11, MaxAmt >= Amt12,!.
```

```

/*****
*
* The max predicate is used to find out which
* observation is the smallest one.
*
*****/
```

```

min(MinAmt):-
    subtotal(1,Amt1),subtotal(2,Amt2),
    subtotal(3,Amt3),subtotal(4,Amt4),
    subtotal(5,Amt5),subtotal(6,Amt6),
    subtotal(7,Amt7),subtotal(8,Amt8),
    subtotal(9,Amt9),subtotal(10,Amt10),
    subtotal(11,Amt11),subtotal(12,Amt12),
    member(MinAmt,[Amt1,Amt2,Amt3,Amt4,Amt5,Amt6,Amt7,Amt8,
    Amt9,Amt10,Amt11,Amt12]),
    MinAmt <= Amt1, MinAmt <= Amt2, MinAmt <= Amt3, MinAmt <=
    Amt4, MinAmt <= Amt5,
    MinAmt <= Amt6, MinAmt <= Amt7, MinAmt <= Amt8, MinAmt <=
    Amt9,
    MinAmt <= Amt10, MinAmt <= Amt11, MinAmt <= Amt12,!.
```

```

/*****
*
* The member predicate is to determine if a given
* element is the member of a list.
*
*****/
```

```

member(Y1,[Y1|_]).
```

```

member(Y1,[_|Tail]):-
    member(Y1,Tail).
```

```

/*****
*
* The drawcurve predicate draws the cashflow curve.
*
*****/
```

```

drawcurve(Scale,MinAmt):-
    subtotal(1,Amt1),subtotal(2,Amt2),
    subtotal(3,Amt3),subtotal(4,Amt4),
    subtotal(5,Amt5),subtotal(6,Amt6),
    subtotal(7,Amt7),subtotal(8,Amt8),
    subtotal(9,Amt9),subtotal(10,Amt10),
    subtotal(11,Amt11),subtotal(12,Amt12),
```

```

% The Sprices' are the scaled price subtotal.

SAmt1 = 300 - (Amt1-MinAmt)/Scale, SAmt2 = 300 - (Amt2-
MinAmt)/Scale,
SAmt3 = 300 - (Amt3-MinAmt)/Scale, SAmt4 = 300 - (Amt4-
MinAmt)/Scale,
SAmt5 = 300 - (Amt5-MinAmt)/Scale, SAmt6 = 300 - (Amt6-
MinAmt)/Scale,
SAmt7 = 300 - (Amt7-MinAmt)/Scale, SAmt8 = 300 - (Amt8-
MinAmt)/Scale,
SAmt9 = 300 - (Amt9-MinAmt)/Scale, SAmt10 = 300 - (Amt10-
MinAmt)/Scale,
SAmt11 = 300 - (Amt11-MinAmt)/Scale, SAmt12 = 300 -
(Amt12-MinAmt)/Scale,!,
date(_ ,MM, _),
    match(MM,[M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12]),
                                % The match predicate
                                % determines how the x-axis
                                % should be labelled.
detectgraph(Gdriver, Gmode),    % Initialize the graph mode
initgraph(Gdriver, Gmode,_,_, bgi_Path),
line(20,150,719,150),          % Draw the X-axis.
line(20,20,20,300),            % Draw the Y-axis.
line(70,SAmt1,120,SAmt2),      % The drawing of the curve
                                % starts here.

line(120,SAmt2,170,SAmt3),
line(170,SAmt3,220,SAmt4),
line(220,SAmt4,270,SAmt5),
line(270,SAmt5,320,SAmt6),
line(320,SAmt6,370,SAmt7),
line(370,SAmt7,420,SAmt8),
line(420,SAmt8,470,SAmt9),
line(470,SAmt9,520,SAmt10),
line(520,SAmt10,570,SAmt11),
line(570,SAmt11,620,SAmt12),!,
moveto(100,1),                  % output the title of graph
outtext("This graph shows the cashflow movements for the
past year."),
moveto(100,10),
outtext("The y-axis is the bank balances in $millions."),
% The labelling of the x-axis starts here.

moveto(70,150),
outtext(M1),
moveto(120,150),
outtext(M2),
moveto(170,150),
outtext(M3),
moveto(220,150),
outtext(M4),
moveto(270,150),
outtext(M5),
moveto(320,150),
outtext(M6),
moveto(370,150),

```

```

    outtext(M7),
    moveto(420,150),
    outtext(M8),
    moveto(470,150),
    outtext(M9),
    moveto(520,150),
    outtext(M10),
    moveto(570,150),
    outtext(M11),
    moveto(620,150),
    outtext(M12),!,
    labely(SAmt1,SAmt2,SAmt3,SAmt4,SAmt5,SAmt6,SAmt7,SAmt8,
    SAMt9,SAmt10,SAmt11,SAmt12),
    moveto(300,337),
    outtext("Please press any key to continue."),readchar(_),
    closegraph.

/*****
*
* The match predicates are used to determine the way
* the x-axis should be labelled.
*
*****/

match(1,["jan","feb","mar","apr","may","jun","jul","aug",
"sep","oct","nov","dec"]).
match(2,["feb","mar","apr","may","jun","jul","aug","sep",
"oct","nov","dec","jan"]).
match(3,["mar","apr","may","jun","jul","aug","sep","oct",
"nov","dec","jan","feb"]).
match(4,["apr","may","jun","jul","aug","sep","oct","nov",
"dec","jan","feb","mar"]).
match(5,["may","jun","jul","aug","sep","oct","nov","dec",
"jan","feb","mar","apr"]).
match(6,["jun","jul","aug","sep","oct","nov","dec","jan",
"feb","mar","apr","may"]).
match(7,["jul","aug","sep","oct","nov","dec","jan","feb",
"mar","apr","may","jun"]).
match(8,["aug","sep","oct","nov","dec","jan","feb","mar",
"apr","may","jun","jul"]).
match(9,["sep","oct","nov","dec","jan","feb","mar","apr",
"may","jun","jul","aug"]).
match(10,["oct","nov","dec","jan","feb","mar","apr","may",
"jun","jul","aug","sep"]).
match(11,["nov","dec","jan","feb","mar","apr","may","jun",
"jul","aug","sep","oct"]).
match(12,["dec","jan","feb","mar","apr","may","jun","jul",
"aug","sep","oct","nov"]).

```

```

/*****
*
*   The labely predicate is used to label the y-axis.
*
*****/

labely(SAmt1,SAmt2,SAmt3,SAmt4,SAmt5,SAmt6,SAmt7,SAmt8,
      SAmt9,SAmt10,SAmt11,SAmt12):-
    subtotal(1,Amt1),subtotal(2,Amt2),
    subtotal(3,Amt3),subtotal(4,Amt4),
    subtotal(5,Amt5),subtotal(6,Amt6),
    subtotal(7,Amt7),subtotal(8,Amt8),
    subtotal(9,Amt9),subtotal(10,Amt10),
    subtotal(11,Amt11),subtotal(12,Amt12),
    moveto(5,SAmt1),                % label the Y-axis
    str_int(Amts1,Amt1),
    outtext(Amts1),
    moveto(5,SAmt2),
    str_int(Amts2,Amt2),
    outtext(Amts2),
    moveto(5,SAmt3),
    str_int(Amts3,Amt3),
    outtext(Amts3),
    moveto(5,SAmt4),
    str_int(Amts4,Amt4),
    outtext(Amts4),
    moveto(5,SAmt5),
    str_int(Amts5,Amt5),
    outtext(Amts5),
    moveto(5,SAmt6),
    str_int(Amts6,Amt6),
    outtext(Amts6),
    moveto(5,SAmt7),
    str_int(Amts7,Amt7),
    outtext(Amts7),
    moveto(5,SAmt8),
    str_int(Amts8,Amt8),
    outtext(Amts8),
    moveto(5,SAmt9),
    str_int(Amts9,Amt9),
    outtext(Amts9),
    moveto(5,SAmt10),str_int(Amts10,Amt10),
    outtext(Amts10),
    moveto(5,SAmt11),
    str_int(Amts11,Amt11),
    outtext(Amts11),
    moveto(5,SAmt12),
    str_int(Amts12,Amt12),
    outtext(Amts12).

/*****
*
*   This is the end of "cash.pro".
*
*****/

```

```
/* LISTING PROGRAM FOR PMAIN.PRO */
```

```

/*****
*
* This subsystem draws inferences and recommend a
* price to the user based on the user's input of his
* evaluations of the various fuzzy factors in the
* decision making environment.
*
*****/

```

domains

```

rlist1 = fig1*
rlist2 = fig2*
rlist3 = fig3*
n1,x,fig1,fig2,fig3,fig4 = real
z = real
type = string
file = pricedata; input
mlist = real*

```

predicates

```

rprice1(z)
rprice2(z)
rprice3(z)
posit1(fig2,rlist2,real)
getprlist(rlist2,rlist1,rlist3,type)
rmatlist1(rlist3,string,type)
rmatlist2(rlist3,string,type)
rmatlist3(rlist3,string,type)
rmatlist4(rlist3,string,type)
rmatlist5(rlist3,string,type)
rmatlist6(rlist3,string,type)
rmatlist7(rlist3,string,type)
rmatlist8(rlist3,string,type)
rmatlist9(rlist3,string,type)
rmatlist10(rlist3,string,type)
rmatlist11(rlist3,string,type)
rmatlist12(rlist3,string,type)
rmatlist13(rlist3,string,type)
rmatlist14(rlist3,string,type)
rmatlist15(rlist3,string,type)
min(fig2,fig1,fig3)
hedgedmf(rlist1,mlist,x,type)
final(z,x,type)
cost(real,real)
get_other_cost(real)
get_sales_balance(integer,real)
get_exp_balance(integer,real)
repfile(file)
mf(mlist,string,type)
sort(rlist2,rlist2)
split(real,rlist2,rlist2,rlist2)
append(rlist2,rlist2,rlist2)
option1(string)

```

```
option2(string)
option3(string)
check(string)
```

goal

```
makewindow(1,7,7,"Marketing Knowledge-based
System",0,0,25,80),nl,
write("Please press 'H' for help."),nl,
write("\n\tI am going to ask you to input your evaluation
of"),
write("\n\tthe current market situation."),nl,
write("\n\tBut before you input your judgement, I am going
to"),
write("\n\tshow you a graphical display of the movements"),
write("\n\tof the prices based on the market indicators of
the last"),
write("\n\ttwelve months."),nl,
write("\n\tIt is hoped that the diagram will be able to"),
write("\n\tto help you to identify whether the wool market
is"),
write("\n\tgoing up or coming down."),nl,
write("\n\tPlease press any key to continue."),
readchar(_),
system("price.exe"), % The "price.exe" subsystem is invoked
% here to draw the price moveme
rprice1(Z1),
gotowindow(1),
clearwindow,
write("\n\tNow I am going to show you the cashflow situation
of"),
write("\n\tyour company in the previous twelve months."),nl,
write("\n\tPlease press any key to continue."),
readchar(_),
system("cash.exe"), % The "cash.exe" subsystem is invoked
% here to draw the cashflow movement curve.
rprice2(Z2),
rprice3(Z3),
write("What is the total amount of wool in kilogram you are
about to sell"),nl,
write("to your customer?"),nl,
readreal(Amt),nl,
cost(Cost,Amt),
Z = Cost*1.021*(1 + (0.665*Z1 + 0.253*Z2 + 0.082*Z3)/100),
write("The recommended price for the wool you are going to
sell to"),
write("your customer is $",Z,"."),nl,
write("Please press any key to continue."),readchar(_).
```

clauses

```

/*****
*
* The cash predicate determines the total cost of
* of the wool to be sold to the customer.
*
*****/

cost(Cost,Amt):-
    openread(pricedata,"price.dat"),
                                % The most current market price
                                % is read from the price data
                                % file.
    readdevice(pricedata),
    filepos(pricedata,-14,2),
    readln(LN),
    closefile(pricedata),
    frontstr(6, LN,_,MI),
    str_real(MI,Price),
                                % Price contains the most current
                                % market price of the wool to be
                                % sold to the customer.
    get_other_cost(Othercost),
    Cost = Amt*Price*(1+abs(Othercost)).
                                % abs(Othercost) expresses the
                                % other cost items in absolute
                                % values. Othercost is obtained
                                % from the following predicate.

/*****
*
* The get_other_cost predicate gets the other sundry
* cost items for the wool to be sold. The total of
* these sundry cost items is expressed as a percentage
* of the total sales figure.
*
*****/

get_other_cost(Othercost):-
    get_sales_balance(400,Sbalance),
    get_exp_balance(350,Ebalance),
    Othercost = Ebalance/Sbalance.

```



```

/*****
*
* The get_sales_balance predicate gets the total sales
* figure from the sales account in the "balance.dat"
* file.
*
*****/

get_sales_balance(400,Sbalance):-
    % "400" is the account number for
    % the sales account in "balance.dat".
    openread(input,"balance.dat"),
    readdevice(input),
    repfile(input),
    readln(LN),
    frontstr(3, LN, Act, Balance),
    str_int(Act, 400),
    str_real(Balance, Sbalance),
    closefile(input).

/*****
*
* The second get_sales_balance outputs an error
* message if the total sales figure is zero.
*
*****/

get_sales_balance(400,Sbalance):-
    Sbalance = 0,
    write("The sales account has a balance of zero!!"),nl,nl,
    write("Please check the balance of the account. ").

/*****
*
* The get_exp_balance predicate gets the total
* expenses figure from the sales account in the
* "balance.dat" file. For simplicity, it is assumed
* that the accounting system has only one expense
* account with the account number "350".
*
*****/

get_exp_balance(350,Ebalance):-
    openread(input,"balance.dat"),
    readdevice(input),
    repfile(input),
    readln(LN),
    frontstr(3, LN, Act, Balance),
    str_int(Act, 350),
    str_real(Balance, Ebalance),
    closefile(input).

```

```

/*****
*
* The second get_exp_balance outputs an error
* message if the total expenses figure is zero.
*
*****/

get_exp_balance(350,Ebalance):-
    Ebalance = 0,
    write("The expense account has a balance of
    zero!!"),nl,nl,
    write("Please check the balance of the account. ").

/*****
*
* The repfile predicate enables the records in a
* file to be read one by one until the end of file
* is reached.
*
*****/

repfile(_).

repfile(F):-
    not(eof(F)),repfile(F).

/*****
*
* The mf predicates contain the fuzzy membership
* functions of the primary fuzzy variables used in
* the system.
*
*****/

mf([0.25,0.67,0.85,0.8,0.7,0.67,0.66],favorable,cashflow).
mf([0.27,0.5,0.87,0.85,0.81,0.8,0.77],unfavorable,cashflow).
mf([0.38,0.57,0.74,0.65,0.65,0.67,0.67],favorable,market).
mf([0.48,0.76,0.81,0.75,0.64,0.65,0.65],unfavorable,market).
mf([0.75,0.57,0.42,0.24,0.36,0.39,0.37,0.37,0.38,0.36,0.33,
    0.32,0.31,0.31,0.3],favorable,competition).
mf([0.43,0.51,0.6,0.61,0.6,0.67,0.68,0.7,0.71,0.73,0.75,0.77,
    0.79,0.8,0.8],unfavorable,competition).

/*****
*
* The different rmatlist predicates are the different
* columns of the six relational matrices. There are
* fifteen columns for each matrix.
*
*****/

rmatlist1([0.95,0.95,0.95,0.95,0.95,0.95,0.95],
    favorable,cashflow).
rmatlist1([0.95,0.95,0.95,0.95,0.95,0.95,0.95],
    favorable,market).
rmatlist1([0.95,0.95,0.95,0.95,0.95,0.95,0.95,0.95,0.95,0.95,

```

```

    0.95,0.95,0.95,0.95,0.95],favorable,competition).
rmatlist1([0.95,0.95,0.95,0.95,0.95,0.95,0.95],
    unfavorable,cashflow).
rmatlist1([0.95,0.95,0.95,0.95,0.95,0.95,0.95],
    unfavorable,market).
rmatlist1([0.95,0.95,0.95,0.95,0.95,0.95,0.95,0.95,0.95,
    0.95,0.95,0.95,0.95,0.95,0.95],unfavorable,competition).

rmatlist2([0.89,0.89,0.89,0.89,0.89,0.89,0.89],
    favorable, cashflow).
rmatlist2([0.89,0.89,0.89,0.89,0.89,0.89,0.89],
    favorable, market).
rmatlist2([0.89,0.89,0.89,0.89,0.89,0.89,0.89,0.89,0.89,0.89,
    0.89,0.89,0.89,0.89,0.89],favorable,competition).
rmatlist2([0.91,0.91,0.91,0.91,0.91,0.91,0.91],
    unfavorable, cashflow).
rmatlist2([0.91,0.91,0.91,0.91,0.91,0.91,0.91],
    unfavorable, market).
rmatlist2([0.91,0.91,0.91,0.91,0.91,0.91,0.91,0.91,0.91,0.91,
    0.91,0.91,0.91,0.91,0.91],unfavorable, competition).

rmatlist3([0.75,0.45,0.45,0.45,0.45,0.45,0.45],
    favorable,cashflow).
rmatlist3([0.62,0.45,0.45,0.45,0.45,0.45,0.45],
    favorable,market).
rmatlist3([0.45,0.45,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
    0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist3([0.73,0.54,0.54,0.54,0.54,0.54,0.54],
    unfavorable,cashflow).
rmatlist3([0.54,0.54,0.54,0.54,0.54,0.54,0.54],
    unfavorable,market).
rmatlist3([0.57,0.54,0.54,0.54,0.54,0.54,0.54,0.54,0.54,0.54,
    0.54,0.54,0.54,0.54,0.54],unfavorable,competition).

rmatlist4([0.75,0.38,0.38,0.38,0.38,0.38,0.38],
    favorable,cashflow).
rmatlist4([0.62,0.43,0.38,0.38,0.38,0.38,0.38],
    favorable,market).
rmatlist4([0.38,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,0.64,
    0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist4([0.73,0.5,0.31,0.31,0.31,0.31,0.31],
    unfavorable,cashflow).
rmatlist4([0.52,0.31,0.31,0.31,0.36,0.35,0.35],
    unfavorable,market).
rmatlist4([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.31,0.31,0.31,
    0.31,0.31,0.31,0.31,0.31],unfavorable,competition).

rmatlist5([0.75,0.33,0.3,0.3,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist5([0.62,0.43,0.3,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist5([0.3,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,0.64,
    0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist5([0.73,0.5,0.27,0.27,0.27,0.27,0.27],
    unfavorable,cashflow).
rmatlist5([0.52,0.27,0.27,0.27,0.36,0.35,0.35],

```

```

    unfavorable,market).
rmatlist5([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
    0.27,0.27,0.27,0.27,0.27],unfavorable,competition).

rmatlist6([0.75,0.33,0.29,0.29,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist6([0.62,0.43,0.29,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist6([0.29,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,0.64,
    0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist6([0.73,0.5,0.28,0.28,0.28,0.28,0.28],
    unfavorable,cashflow).
rmatlist6([0.52,0.28,0.28,0.28,0.36,0.35,0.35],
    unfavorable,market).
rmatlist6([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.28,
    0.28,0.28,0.28,0.28,0.28],unfavorable,competition).

rmatlist7([0.75,0.33,0.25,0.25,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist7([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist7([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,0.64,
    0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist7([0.73,0.5,0.28,0.28,0.28,0.28,0.28],
    unfavorable,cashflow).
rmatlist7([0.52,0.28,0.28,0.28,0.36,0.35,0.35],
    unfavorable,market).
rmatlist7([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.28,0.28,
    0.28,0.28,0.28,0.28],unfavorable,competition).

rmatlist8([0.75,0.33,0.22,0.22,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist8([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist8([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,0.64,
    0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist8([0.73,0.5,0.25,0.25,0.25,0.25,0.25],
    unfavorable,cashflow).
rmatlist8([0.52,0.25,0.25,0.25,0.36,0.35,0.35],
    unfavorable,market).
rmatlist8([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,0.25,
    0.25,0.25,0.25,0.25],unfavorable,competition).

rmatlist9([0.75,0.33,0.21,0.21,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist9([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist9([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,0.64,
    0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist9([0.73,0.5,0.24,0.24,0.24,0.24,0.24],
    unfavorable,cashflow).
rmatlist9([0.52,0.24,0.24,0.25,0.36,0.35,0.35],
    unfavorable,market).
rmatlist9([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,0.25,
    0.24,0.24,0.24,0.24],unfavorable,competition).

```

```

rmatlist10([0.75,0.33,0.2,0.2,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist10([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist10([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
    0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist10([0.73,0.5,0.24,0.24,0.24,0.24,0.24],
    unfavorable,cashflow).
rmatlist10([0.52,0.24,0.24,0.25,0.36,0.35,0.35],
    unfavorable,market).
rmatlist10([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
    0.25,0.24,0.24,0.24,0.24],unfavorable,competition).

rmatlist11([0.75,0.33,0.2,0.2,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist11([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist11([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
    0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist11([0.73,0.5,0.23,0.23,0.23,0.23,0.23],
    unfavorable,cashflow).
rmatlist11([0.52,0.24,0.23,0.25,0.36,0.35,0.35],
    unfavorable,market).
rmatlist11([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
    0.25,0.23,0.23,0.23,0.23],unfavorable,competition).

rmatlist12([0.75,0.33,0.2,0.2,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist12([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist12([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
    0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist12([0.73,0.5,0.23,0.23,0.23,0.23,0.23],
    unfavorable,cashflow).
rmatlist12([0.52,0.24,0.23,0.25,0.36,0.35,0.35],
    unfavorable,market).
rmatlist12([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
    0.25,0.23,0.23,0.23,0.23],unfavorable,competition).

rmatlist13([0.75,0.33,0.2,0.2,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist13([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).
rmatlist13([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
    0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist13([0.73,0.5,0.22,0.22,0.22,0.22,0.23],
    unfavorable,cashflow).
rmatlist13([0.52,0.24,0.22,0.25,0.36,0.35,0.35],
    unfavorable,market).
rmatlist13([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
    0.25,0.23,0.22,0.22,0.22],unfavorable,competition).

rmatlist14([0.75,0.33,0.2,0.2,0.3,0.33,0.34],
    favorable,cashflow).
rmatlist14([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
    favorable,market).

```

```

rmatlist14([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist14([0.73,0.5,0.21,0.21,0.21,0.21,0.23],
unfavorable,cashflow).
rmatlist14([0.52,0.24,0.21,0.25,0.36,0.35,0.35],
unfavorable,market).
rmatlist14([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
0.25,0.23,0.21,0.21,0.21],unfavorable,competition).

rmatlist15([0.75,0.33,0.2,0.2,0.3,0.33,0.34],
favorable,cashflow).
rmatlist15([0.62,0.43,0.26,0.35,0.35,0.33,0.33],
favorable,market).
rmatlist15([0.25,0.43,0.58,0.76,0.64,0.61,0.63,0.63,0.62,
0.64,0.67,0.68,0.69,0.69,0.7],favorable,competition).
rmatlist15([0.73,0.5,0.2,0.2,0.2,0.2,0.23],
unfavorable,cashflow).
rmatlist15([0.52,0.24,0.2,0.25,0.36,0.35,0.35],
unfavorable,market).
rmatlist15([0.57,0.49,0.4,0.39,0.4,0.33,0.32,0.3,0.29,0.27,
0.25,0.23,0.21,0.2,0.2],unfavorable,competition).

/*****
*
* rpricel obtains the percentage price change due to
* the market situation.
*
*****/

rpricel(Z1):-
makewindow(2,7,7,"Please tell me your market
situation",2,5,20,70),nl,
option1(P),
str_real(P,P1),
final(Z1,P1,market),!.

/*****
*
* option1 obtains the user's judgment of the market
* situation.
*
*****/

option1(P):-
write("Please select one of the following options that
best describes"),nl,
write("your perception of the current market
situation:"),nl,nl,
write("\t1. Very very favourable"),nl,
write("\t2. Very favoruable"),nl,
write("\t3. Favourable"),nl,
write("\t4. Slightly favourable"),nl,
write("\t5. Neither favorable nor unfavorable"),nl,
write("\t6. Slightly unfavourable"),nl,
write("\t7. Unfavourable"),nl,
write("\t8. Very unfavourable"),nl,

```

```

write("\t9. Very very unfavourable"),nl,nl,
write("Please indicate your choice by keying in the
appropriate number:  "),
readln(P), % P contains the option chosen by
            % the user.

check(P),
P <> "H",
P <> "h".

/*****
*
* The second option1 predicate provides some help to
* the user if the help key is pressed.
*
*****/

option1(P):-
    nl,write("Please input a number from 1 to 9 to indicate
your "),
write("choice."),nl,nl,
option1(P).

/*****
*
* rprice2 obtains the percentage price change due to
* the cashflow situation.
*
*****/

rprice2(Z2):-
    makewindow(3,7,7,"Please tell me your cashflow
situation",2,5,20,70),nl,
    option2(P),
    str_real(P,P2),
    final(Z2,P2,cashflow),!.

/*****
*
* option2 obtains the user's judgment of the cashflow
* situation.
*
*****/

option2(P):-
    write("Please select one of the following options that
best describes your current cashflow situation:"),nl,nl,
    write("\t1. Very very favourable"),nl,
    write("\t2. Very favoruable"),nl,
    write("\t3. Favourable"),nl,
    write("\t4. Slightly favourable"),nl,
    write("\t5. Neither favorable nor unfavorable"),nl,
    write("\t6. Slightly unfavourable"),nl,
    write("\t7. Unfavourable"),nl,
    write("\t8. Very unfavourable"),nl,
    write("\t9. Very very unfavourable"),nl,nl,
    write("Please indicate your choice by keying in the

```

```

appropriate number:  "),
readln(P),           % P contains the option chosen by
                    % the user.

check(P),
P <> "H",
P <> "h".

/*****
*
* The second option2 predicate provides some help to
* the user if the help key is pressed.
*
*****/

option2(P):-
    nl,write("Please input a number from 1 to 9 to indicate
your "),
write("choice."),nl,nl,
option2(P).

/*****
*
* rprice3 obtains the percentage price change due to
* the competitive situation.
*
*****/

rprice3(Z3):-
    clearwindow,
    makewindow(4,7,7,"Please tell me your competitive
situation",2,5,20,70),nl,
option3(P),
str_real(P,P3),
final(Z3,P3,competition).

/*****
*
* option3 obtains the user's judgment of the
* competitive situation.
*
*****/

option3(P):-
    write("Please select one of the following options that
best describes your current competitive
situation:"),nl,nl,
write("\t1. Very very weak"),nl,
write("\t2. Very weak"),nl,
write("\t3. Weak"),nl,
write("\t4. Slightly weak"),nl,
write("\t5. Neither weak nor strong"),nl,
write("\t6. Slightly strong"),nl,
write("\t7. Strong"),nl,
write("\t8. Very strong"),nl,
write("\t9. Very very strong"),nl,nl,
write("Please indicate your choice by keying in the

```



```

appropriate number:  "),
readln(P),           % P contains the option chosen by
                     % the user.

check(P),
P <> "H",
P <> "h".

/*****
*
*   The second option3 predicate provides some help to
*   the user if the help key is pressed.
*
*****/

option3(P):-
    nl,write("Please  input a number from 1 to 9 to  indicate
your "),
    write("choice."),nl,nl,
    option3(P).

/*****
*
*   The check predicates check if the correct option
*   has been entered.
*
*****/

check("1").
check("2").
check("3").
check("4").
check("5").
check("6").
check("7").
check("8").
check("9").

/*****
*
*   If the user chooses option 5, that is the "neither
*   ... nor" option, the price change due to the fuzzy
*   factor is zero.
*
*****/

final(ZZ,5,_):-
    ZZ = 0.

```

```

/*****
*
* The final predicate gets the percentage change
* due to each fuzzy factor. PP is the option chosen
* by the user. Type is the fuzzy factor in question.
*
*****/

```

```

final(ZZ,PP,Type):-

```

```

    PP < 5, !,      % If PP > 5, the percentage price change
                   % will be positive.

```

```

    mf(Glist,favorable,Type),
    hedgedmf(Mlist,Glist,PP,Type),!,
    rmatlist1(Rlist1,favorable,Type),
    getprlist(Prlist1,Mlist,Rlist1,Type),!,sort(Prlist1,[S1|_]),!,
    rmatlist2(Rlist2,favorable,Type),
    getprlist(Prlist2,Mlist,Rlist2,Type),!,sort(Prlist2,[S2|_]),!,
    rmatlist3(Rlist3,favorable,Type),
    getprlist(Prlist3,Mlist,Rlist3,Type),!,sort(Prlist3,[S3|_]),!,
    rmatlist4(Rlist4,favorable,Type),
    getprlist(Prlist4,Mlist,Rlist4,Type),!,sort(Prlist4,[S4|_]),!,
    rmatlist5(Rlist5,favorable,Type),
    getprlist(Prlist5,Mlist,Rlist5,Type),!,sort(Prlist5,[S5|_]),!,
    rmatlist6(Rlist6,favorable,Type),
    getprlist(Prlist6,Mlist,Rlist6,Type),!,sort(Prlist6,[S6|_]),!,
    rmatlist7(Rlist7,favorable,Type),
    getprlist(Prlist7,Mlist,Rlist7,Type),!,sort(Prlist7,[S7|_]),!,
    rmatlist8(Rlist8,favorable,Type),
    getprlist(Prlist8,Mlist,Rlist8,Type),!,sort(Prlist8,[S8|_]),!,
    rmatlist9(Rlist9,favorable,Type),
    getprlist(Prlist9,Mlist,Rlist9,Type),!,sort(Prlist9,[S9|_]),!,
    rmatlist10(Rlist10,favorable,Type),
    getprlist(Prlist10,Mlist,Rlist10,Type),!,
    sort(Prlist10,[S10|_]),!,
    rmatlist11(Rlist11,favorable,Type),
    getprlist(Prlist11,Mlist,Rlist11,Type),!,
    sort(Prlist11,[S11|_]),!,
    rmatlist12(Rlist12,favorable,Type),
    getprlist(Prlist12,Mlist,Rlist12,Type),!,
    sort(Prlist12,[S12|_]),!,
    rmatlist13(Rlist13,favorable,Type),
    getprlist(Prlist13,Mlist,Rlist13,Type),!,
    sort(Prlist13,[S13|_]),!,
    rmatlist14(Rlist14,favorable,Type),
    getprlist(Prlist14,Mlist,Rlist14,Type),!,
    sort(Prlist14,[S14|_]),!,
    rmatlist15(Rlist15,favorable,Type),
    getprlist(Prlist15,Mlist,Rlist15,Type),!,
    sort(Prlist15,[S15|_]),!,
    sort([S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,S15],
    [Y1|_]),!,
    posit1(Y1,[S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,
    S15],ZZ).

```

```

final(ZZ,PP,Type):-
    PP > 5, P = 10 - PP,      % The price change is negative if
                                % PP > 5.
    mf(Glist,unfavorable,Type),
    hedgedmf(Mlist,Glist,P,Type),!,
    rmatlist1(Rlist1,unfavorable,Type),
    getprlist(Prlist1,Mlist,Rlist1,Type),!,sort(Prlist1,[S1|_]),!,
    rmatlist2(Rlist2,unfavorable,Type),
    getprlist(Prlist2,Mlist,Rlist2,Type),!,sort(Prlist2,[S2|_]),!,
    rmatlist3(Rlist3,unfavorable,Type),
    getprlist(Prlist3,Mlist,Rlist3,Type),!,sort(Prlist3,[S3|_]),!,
    rmatlist4(Rlist4,unfavorable,Type),
    getprlist(Prlist4,Mlist,Rlist4,Type),!,sort(Prlist4,[S4|_]),!,
    rmatlist5(Rlist5,unfavorable,Type),
    getprlist(Prlist5,Mlist,Rlist5,Type),!,sort(Prlist5,[S5|_]),!,
    rmatlist6(Rlist6,unfavorable,Type),
    getprlist(Prlist6,Mlist,Rlist6,Type),!,sort(Prlist6,[S6|_]),!,
    rmatlist7(Rlist7,unfavorable,Type),
    getprlist(Prlist7,Mlist,Rlist7,Type),!,sort(Prlist7,[S7|_]),!,
    rmatlist8(Rlist8,unfavorable,Type),
    getprlist(Prlist8,Mlist,Rlist8,Type),!,sort(Prlist8,[S8|_]),!,
    rmatlist9(Rlist9,unfavorable,Type),
    getprlist(Prlist9,Mlist,Rlist9,Type),!,sort(Prlist9,[S9|_]),!,
    rmatlist10(Rlist10,unfavorable,Type),
    getprlist(Prlist10,Mlist,Rlist10,Type),!,
    sort(Prlist10,[S10|_]),!,
    rmatlist11(Rlist11,unfavorable,Type),
    getprlist(Prlist11,Mlist,Rlist11,Type),!,
    sort(Prlist11,[S11|_]),!,
    rmatlist12(Rlist12,unfavorable,Type),
    getprlist(Prlist12,Mlist,Rlist12,Type),!,
    sort(Prlist12,[S12|_]),!,
    rmatlist13(Rlist13,unfavorable,Type),
    getprlist(Prlist13,Mlist,Rlist13,Type),!,
    sort(Prlist13,[S13|_]),!,
    rmatlist14(Rlist14,unfavorable,Type),
    getprlist(Prlist14,Mlist,Rlist14,Type),!,
    sort(Prlist14,[S14|_]),!,
    rmatlist15(Rlist15,unfavorable,Type),
    getprlist(Prlist15,Mlist,Rlist15,Type),!,
    sort(Prlist15,[S15|_]),!,
    sort([S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,S15],
    [Y1|_]),!,
    posit1(Y1,[S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,
    S15],N1),
    ZZ = - N1.

```

```

/*****
*
* The hedgedmf predicates are used to get the hedged
* membership functions depending on the option
* chosen by the user.
*
*****/

hedgedmf([],[],_,_).

hedgedmf([Cf110|ATail],[Cf10|BTail],1,Type):-
    Cf110 = Cf10*Cf10*Cf10*Cf10,
    hedgedmf(ATail,BTail,1,Type).

hedgedmf([Cf110|ATail],[Cf10|BTail],2,Type):-
    Cf110 = Cf10*Cf10,
    hedgedmf(ATail,BTail,2,Type).

hedgedmf([Cf110|ATail],[Cf10|BTail],3,Type):-
    Cf110 = Cf10,
    hedgedmf(ATail,BTail,3,Type).

hedgedmf([Cf110|ATail],[Cf10|BTail],4,Type):-
    Cf110 = sqrt(sqrt(Cf10*Cf10*Cf10)),
    hedgedmf(ATail,BTail,4,Type).

/*****
*
* The getprlist predicate is used to perform the
* min operations in the max-min compositional rule
* of inference.
*
*****/

getprlist([],[],[],_).

getprlist([AHead|ATail],[BHead|BTail],[CHead|CTail],Type):-
    min(AHead,BHead,CHead),
    getprlist(ATail,BTail,CTail,Type).

/*****
*
* The sort, split and append predicates are used to
* sort a given list.
*
*****/

sort([],[]).
sort([H|T],S) if
    split(H,T,U1,U2),
    sort(U1,V1),sort(U2,V2),
    append(V1,[H|V2],S),!.

split(M,[H|T],[H|U1],U2) if
    H >= M, split(M,T,U1,U2).
split(M,[H|T],U1,[H|U2]) if

```

```

    split(M,T,U1,U2).
split(_,[],[],[]).

append([],L,L).
append([H|T],L2,[H|L3]):--append(T,L2,L3).

/*****
*
*   The posit predicates are used to determine the
*   percentage price change for each fuzzy factor input
*   given by the user.
*
*****/

posit1(Y1,[R1|_],1):-
    Y1 = R1.
posit1(Y1,[_,R2|_],2):-
    Y1 = R2.
posit1(Y1,[_,_,R3|_],3):-
    Y1 = R3.
posit1(Y1,[_,_,_,R4|_],4):-
    Y1 = R4.
posit1(Y1,[_,_,_,_,R5|_],5):-
    Y1 = R5.
posit1(Y1,[_,_,_,_,_,R6|_],6):-
    Y1 = R6.
posit1(Y1,[_,_,_,_,_,_,R7|_],7):-
    Y1 = R7.
posit1(Y1,[_,_,_,_,_,_,_,R8|_],8):-
    Y1 = R8.
posit1(Y1,[_,_,_,_,_,_,_,_,R9|_],9):-
    Y1 = R9.
posit1(Y1,[_,_,_,_,_,_,_,_,_,R10|_],10):-
    Y1 = R10.
posit1(Y1,[_,_,_,_,_,_,_,_,_,_,R11|_],11):-
    Y1 = R11.
posit1(Y1,[_,_,_,_,_,_,_,_,_,_,_,R12|_],12):-
    Y1 = R12.
posit1(Y1,[_,_,_,_,_,_,_,_,_,_,_,_,R13|_],13):-
    Y1 = R13.
posit1(Y1,[_,_,_,_,_,_,_,_,_,_,_,_,_,R14|_],14):-
    Y1 = R14.
posit1(Y1,[_,_,_,_,_,_,_,_,_,_,_,_,_,_,R15],15):-
    Y1 = R15.

```

```

/*****
*
*   The min predicate is used to determine the minimum
*   of three given numbers.
*
*****/

min(Cf,Rm,Pr) if
    Pr >=Rm, Cf = Rm.
min(Cf,_,Pr) if
    Pr =Cf.

/*****
*
*   This is the end of "pmain.pro".
*
*****/

```