USING A MATHEMATICAL MODEL IN CONJUNCTION WITH EXPERT SYSTEMS TO ASSIST INVESTMENT DECISIONS

By

MIN ZHANG

Bachelor of Science

Shanghai Second Polytechnic University

Shanghai, P. R. China

1988

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 1996

USING A MATHEMATICAL MODEL IN CONJUNCTION

WITH EXPERT SYSTEMS TO ASSIST

INVESTMENT DECISIONS

Thesis Approved:

Thesis Adviser

Blayne E. Mayfield

Thomas C. Collins

Dean of the Graduate College

ACKNOWLEDGMENTS

I wish to express my sincere appreciation to my thesis advisor, Dr. George E. Hedrick for his intelligent supervision, constructive guidance, and inspiration. I also wish to express my gratitude to Dr. Jacques E. LaFrance for his guidance, encouragement, and friendship. My sincere appreciation extends to my thesis committee chairman Dr. Blayne E. Mayfield, whose guidance, assistance, and encouragement are also invaluable.

I would like to give my special appreciation to my wife, Weili, for her precious suggestions to my research, her strong encouragement at times of difficulty, love and understanding throughout this whole process. Thanks also go to my parents for their support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Expert systems are important applications of Artificial Intelligence. An expert system is a computer program that emulates the behavior of human experts in a narrowly but well-defined domain of knowledge. It can mimic the problem solving abilities of experts in particular areas.

The development of expert systems can be traced back to the 1950s. In 1957, MaCarthy invented LISP (List Processing), a programming language for handling symbolic processing for Artificial Intelligence and expert systems application [24]. That same year, the Rand-Carnegie team of Newell, Shaw, and Simon developed the General Problem Solver (GPS) to solve problems of elementary logic, chess, high school algebra, and word-problems [10, 32]. During the 1960s, a number of early expert systems had been designed, such as DENDRAL for mass spectroscopy [22], SAINT for symbolic integration, STUDENT for solving high school algebra and word-problems [28 p. 33]. However, expert systems were not used practically until 1972 when MYCIN was developed at Stanford University for diagnosing bacterial infections in blood [5, 37, 28 p. 4]. In the 1970s, following the work on MYCIN, the technology for developing expert systems advanced. Colemerauer invented PROLOG (Programming in Logic) in 1970 [8, 28 p. 4]. The LISP machine, the first specialized AI computer, was invented at MIT in 1977 [28 p. 5]. During that period, industrial interest in developing expert systems also increased. Expert systems were developed in application areas, such as diagnosis, perception, instruction, learning, game playing, programming, and theorem proving [28 p. 33].

Entering into the 1980s, expert systems became commercially available. The first commercial expert system DEC's XCON was put into use in 1981 [25, 26, 15 p. 4]. Expert systems building tools such as OPS5 [12], KEE and S.1 [35]were first offered for sale around 1983 [15 p. 4]. During the same period, researchers also started their work in the commercial area. Bouwman studied and summarized human financial decision making procedure [3]. 1985 and 1986 saw the first wave of expert systems driven by research and development groups that wanted to learn more about new technology [15 p. 4]. That wave died down in 1987 and 1988, as companies tried to absorb and evaluate what they had acquired in previous years [15 p.4]. The years 1989 and 1990 marked the turning point of expert systems. Since then, wide acceptance of expert systems occurred [23, 11]. The usage of expert systems has transformed the ways companies develop software, use computers, and do business [15 pp. 7-11].

With the widespread use of computers, and with the advances in microcomputer technology, business expert systems in the areas of accounting, taxation, banking, and planning started to develop in the 1980s [36, 29, 1, 42]. Since 1980, much research and development have been undertaken by both the computer industry and academic researchers. Some companies, such as the big 6 accounting firms (Arthur Anderson, Price Waterhouse, Deloitte & Touche, Coopers & Lybrand, Ernest & Young, and KPMG), have adopted expert systems to accomplish complex tasks, enhance efficiency, and reduce costs. The trend is that expert systems will be integrated into the daily operations of those companies that desire to maintain a competitive advantage in the 21st century [6].

In recent years, more expert systems in business applications have been developed and used. For example, EXPERTAX developed by Coopers & Lybrand for tax planning identifies tax versus book difference and explains the difference between statutory and effective tax rates [11]. LESSEE developed at the University of Tennessee is used in financial statement preparation and is able to determine classification and amortization schedules for leases [11]. EDP-XPERT developed by Hansen and Messier can be used to evaluate the reliability of controls in advanced computer systems [11]. AIG/DIC developed by American International is used by insurance companies to underwrite complex insurance policies[11]. CREDIT AUTHORIZER developed by American Express is used for risk analysis to determine credit authorization and credit limits [11].

Meanwhile, researchers also have devoted resources to research business expert systems. Their studies covered different areas of business. A. A. Sindi studied the role of user acceptance in determining the success or failure of expert systems [41]. A model was developed to understand and predict user acceptance of expert systems. Sindi concluded that users' intentions of using expert systems were the only determinant of the frequency of system use.

Chui-Yu Chiu, in his research, focused on the problem of capital budgeting which involve uncertainties about future cash flow [7]. He proposed using fuzzy set theory to deal with the uncertainties of cash flow and the discount rate.

Man-Ho Han proposed a comprehensive conceptual framework for corporate portfolio analysis and exploited the ability of a knowledge-based decision support system (KBDSS) to accommodate the complex dimensions of portfolio analysis in a vigorous way [14]. Based on the Structured Factors Analysis Model (SFAM), a prototype of KBDSS called the Corporate Portfolio Analysis Support System (COPASS) was constructed and tested to demonstrate the potential of the framework.

Laurie Swinney researched the extent of auditors' reliance on expert system judgments [43]. Although the expert system was capable of making judgments, the results of the research indicated excessive reliance on the expert system only when the expert system's judgment is negative.

Walter Hamscher from the Price Waterhouse Technology Center undertook a research on Business Understander, a second generation knowledge-based facility for supporting the understanding of clients' business [13]. The key component was the business analyzer finding anomalies in financial results and computing explanation for them. In this system, causal knowledge was represented in the form of constraints among financial variables while empirical knowledge was represented as probability distribution over alternative assumptions.

McEacharn's research effort was directed toward the development of an expert system that incorporated fuzzy logic into the planning-stage materiality judgments [27]. Planning-stage audit materiality judgments typically involve considerations of large amounts of subjective and quantitative factors. Fuzzy logic permitted explicit consideration of these subjective and quantitative factors. The research result illustrated the applicability of fuzzy logic to ambiguous accounting decision situations.

Eugene Krushelnycky developed a model of merging rule-based expertise from two experts [19]. He believed that two-expert expert systems were more flexible and offered a wider selection of both inputs and possible solution sets for users. To validate his theory, eight such systems were constructed to compare a prior crisp and fuzzy system with the posterior system and empirical results. A classification scheme was also introduced in his systems to identify the critical variables of inter-expert solution. He stated that the result proved that the model that incorporated the classification scheme of interaction using critical variables was superior to the models that utilized a simple union scheme for merging rules.

Thiruvengadam Ravi combined knowledge-based expert systems and a simulation model to design and evaluate Flexible Manufacturing Systems (FMS) which was a new generation manufacturing systems [34]. The system included an expert system synthesizer to generate initial system designs, a simulation model developer to convert designs into graphical simulation models automatically, and an expert analyzer to analyze simulation outputs, identify design deficiencies, and recommend changes.

Mary Jane Lenard developed a hybrid expert system to make going concern assessments [20]. Her system included two components. One was the statistical model that was used to predict bankruptcy. The other was an expert system. She integrated the statistical model and the expert system to form the going concern hybrid expert system that can recommend whether auditors should modify the audit report with a going concern uncertainty.

In the past several years, researchers began to incorporate multiple problem-solving methods to construct expert systems to solve complex problems. One common approach is to combine associational (pattern-matching) and causal (model-based) reasoning [9, 38]. The others include combining causal and case-based reasoning [18]; causal and probabilistic reasoning [13]; qualitative and quantitative methods [33].

This thesis addresses the possibility of combining a mathematical model with expert systems to make financial decisions. In today's business world, more complex problems are being encountered than ever before, such as joint venture investment decisions. These kinds of problems involve knowledge from different segments. Normally, there are no clear cut answers to these problems due to the uncertainties involved in solving these kinds of problems. However, these kinds of complex problems can be solved by using multiple methods. First, the whole problem can be broken into separate sub-problems. Then different methods can be employed to deal with each sub-problem. Finally, the whole investment problem can be resolved by integrating these solved sub-problems.

This research paper proposes a system design for financial feasibility study that includes a mathematical model, two kinds of expert system designs, and methods of dealing with uncertainty. This paper introduces an experiment to demonstrate the possibility of integrating different methods in solving complex financial problems.

CHAPTER II

THE NATURE OF A FINANCIAL FEASIBILITY STUDY AND METHODS OF EXPERT SYSTEM DESIGN

The Nature of a Financial Feasibility Study

A joint venture investment requires a long term commitment from investors. It is often risky and involves large amount of capital. Thus, it needs to be researched thoroughly before a final decision can be made. The feasibility study of a joint venture investment is a comprehensive analysis of the project. It helps investors to have a better understanding of the project before reaching their final decision.

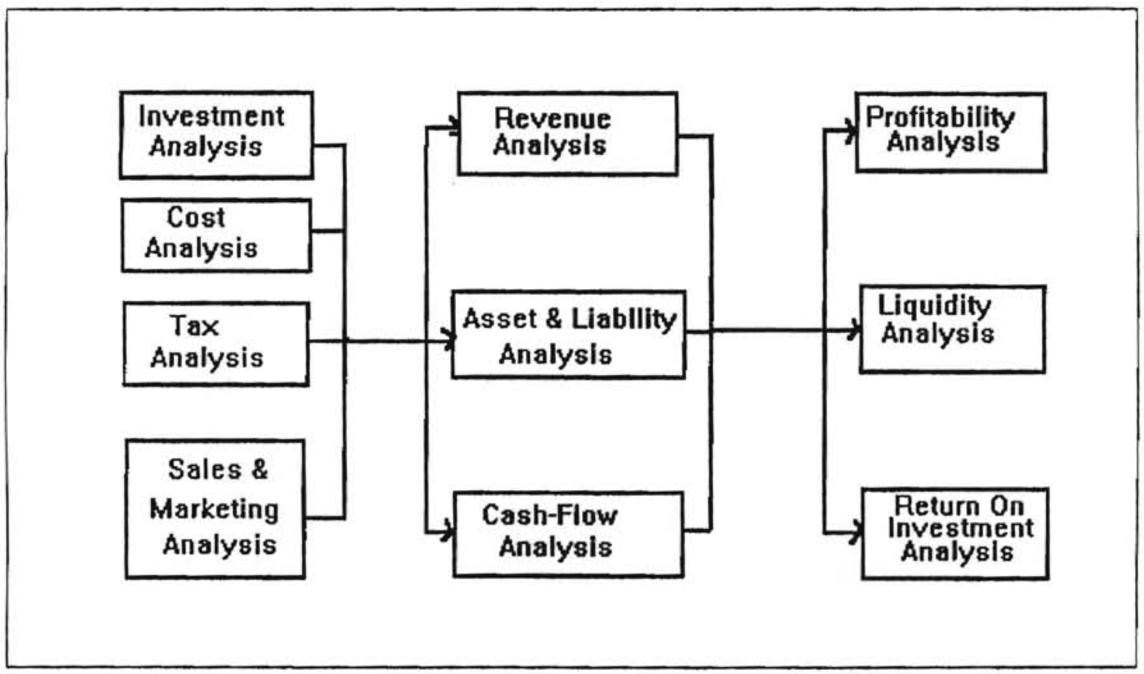

Figure 1 The Process of a Financial Feasibility Study

A feasibility study usually contains ten parts as shown in Figure 1. These analyses are individually independent, because each has a set of rules of analysis and each has its own characteristics. Yet they are also closely related, because the result of one analysis might be the input of another analysis. For example, the results of sales and marketing forecasts are the bases of revenue analysis. The result of tax analysis is an important input of profit analysis. The result of depreciation analysis affects gross revenue.

There are several different evaluation methods, such as NPV (Net Present Value), IRR (Internal Rate of Return), and BCR (Benefit-Cost Rate). These methods enable us to analyze the soundness of the investment from different perspectives. Meanwhile, sensitivity, uncertainty, and risk analysis of the financial indicators (such as IRR, NPV, break even point, profit margin) need to be conducted as well. These analyses identify and analyze the factors that are most critical to the success of the project, because the accuracy of these financial indicators depends on the reliability of different forecasts.

Methods of Expert System Design

Major Components of Expert System

An expert system usually contains four major parts: a knowledge base, an inference engine, a user interface, and an explanation facility as shown in Figure 2 [23, 44 p. 473]. The knowledge base is a collection of facts and rules pertinent to the application area. It contains all the information needed to solve a problem in a specific application area. The inference Engine is the "brain" of expert system, also known as its control structure. It enables user to use the captured knowledge in the knowledge base. User Interface provides communication between the user and the inference engine. It also presents backup from the explanation facility. The explanation facility explains the system's reasoning and justifies its conclusion [11].

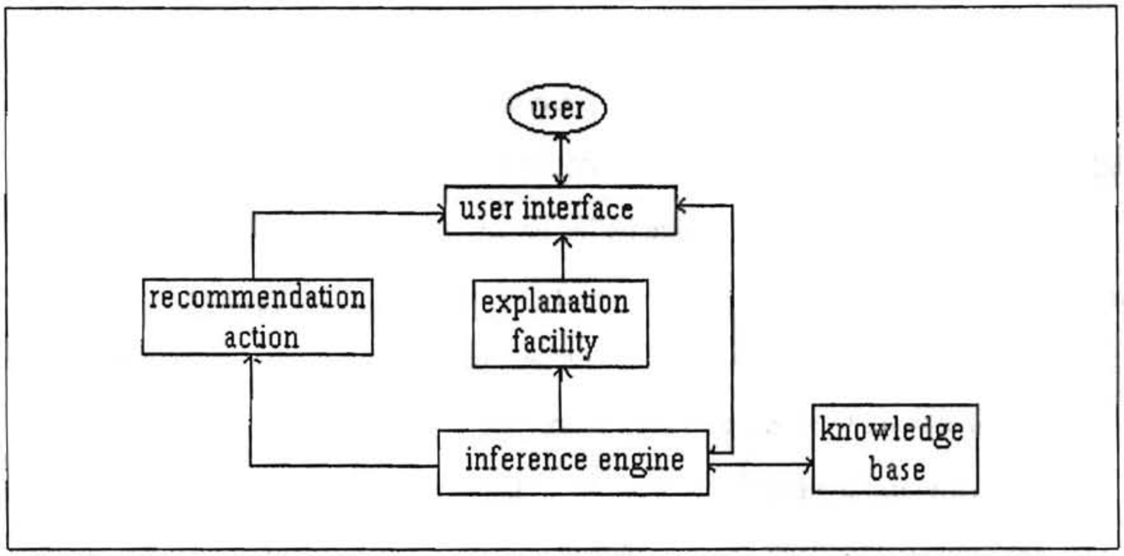

Figure 2 Structure of an Expert System

Methods

There are several commonly used inference methods in building expert system -- inferencing with rules, inferencing with frame, model-based reasoning, case based reasoning, and inferencing with uncertainty.

Inferencing with rules involves implementation of the decision process based on IF-THEN rules, which is reflected in the search mechanism with the rule interpreter [44 pp. 589-596]. There are two ways to control inference in rule-based expert system: forward chaining and backward chaining. Forward chaining is a data-driven approach. This approach starts from available information, then seeks conclusions by looking for the facts that match the IF portion of its IF-THEN rules. In contrast to forward chaining, backward chaining is a goal-driven approach in which it starts from an expectation of what is to happen and then seeks evidence that supports (or contradicts) the expectation.

Inference with frames is much more complicated than inferencing with rules [44 p. 596]. A frame is a data structure that includes all the knowledge about a particular object. It usually includes two basic elements: slot and facet. A slot is a set of attributes that describe the object represented by the frame. Each slot contains one or more facets that describe knowledge or procedures about the attributes in the slot. The slot provides a mechanism for inferencing called expectation-driven processing. The inference processing that takes place with frame is essentially the seeking of confirmation of various expectations. By using frames, it is easier to make inferences about new objects, events, or situations because the frames provide a base of knowledge drawn from previous experiences. There are two different ways to implement inferencing with frames. The first is using rules in frames. A rule can reason about the characteristics of a frame by referring to its slot values. The second is using hierarchical reasoning. According to hierarchical reasoning, certain alternatives, objects, or events can be eliminated at various levels of the search hierarchy [44 pp. 596-599].

Model based systems are especially useful in diagnosing equipment problems. Model-based reasoning is based on knowledge of the structure and behavior of the equipment that the system is designed to understand. The models used in this type of reasoning can be either mathematical models or component models. A mathematical model can simulate real situations, while component models contain functional descriptions of all components and their interactions [44 pp. 599-601].

Case based reasoning adapts solutions that were used to solve previous problems and uses them to solve new problems. It finds cases in the data base that solved problems similar to the current one and adapts the previous solutions to fit the current problem. Differences between the current and previous situations are taken into consideration during the process [44 pp. 601-603].

Inferencing with uncertainty is a three-step process as shown in Figure 3 produced by Turban [44 p. 610]. In step one, an expert provides inexact knowledge in terms of rules with likelihood values. In step two, the inexact knowledge of the basic set of events can be used directly to draw inference in simple cases (Step 3). However, since events are frequently interrelated, it is necessary to combine the information provided in step one to create a global value for the system. Methods that can be used for such integration are Bayesian Probabilities, theory of evidence, certainty factors, and fuzzy sets. Step three is to draw inferences. Working with an inference engine, the input given in step one can be adjusted after viewing the results from step two and step three [44 pp. 610-611].

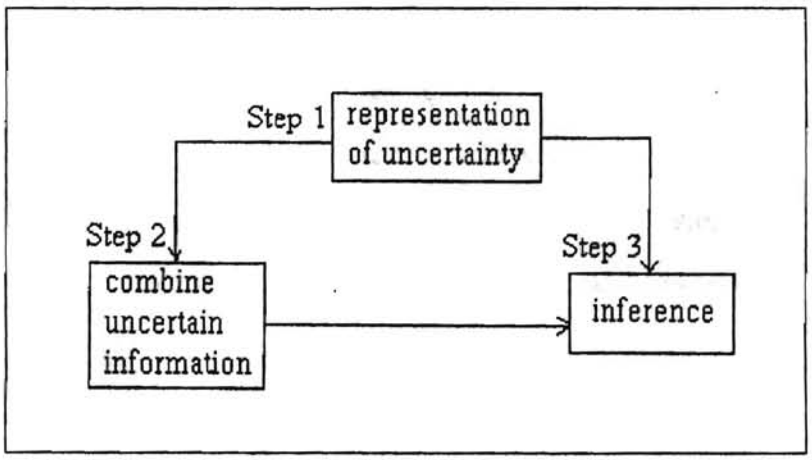

Figure 3 Dealing with Uncertainty in an Expert System

CHAPTER III

SYSTEM DESIGN FOR A FINANCIAL FEASIBILITY STUDY

Theoretical Framework for the System Design

Human information-processing theorists hypothesize that decision making can be divided into several phases [39]. The first phase includes cognitive activities concerned with recognizing relevant information in decision makers' environment. This phase is referred to as information acquisition. Although this step can include some preliminary interpretive data processing, its main purpose is to establish the stage for later problem-solving activities that process first interpretations into a final decision. This initial exploratory activity was further studied by Ericson, Bouwman and Biggs in a series of process tracing experiments [40, 3, 2]. Each of these experiments asked experienced subjects to talk about their thinking processes while solving a significant financial problem. Protocols transcribed from these recorded sessions showed extensive usage of financial statement data. Figure 4 produced by Mui shows Bouwman's explanation of financial decision making process [3, 30 p. 129].

Bouwman divided financial decision making into two phases: familiarization and reasoning. His experimental subjects included two groups of experts making two types of decisions. The first group was financial analysts evaluating a stock for possible investment. The second group was loan officers evaluating a multi-million-dollar participation loan. According to Bouwman, as Figure 4 illustrates, the process of financial decision making consists of four steps.

1) Scan the environment and background values to identify key items such

as "sales" or "net income."

2) Evoke financial templates for companies or industries from long term memory, such as "high-tech company" or "late recessionary industry."

3) Search for instantiations of these templates with specific information (i.e. a more directed reading of initial data).

4) Evaluate, or decide overall.

The first step corresponds to the phase of familiarizing, while the second, third and fourth steps correspond to the phase of reasoning [30 pp. 129-130].

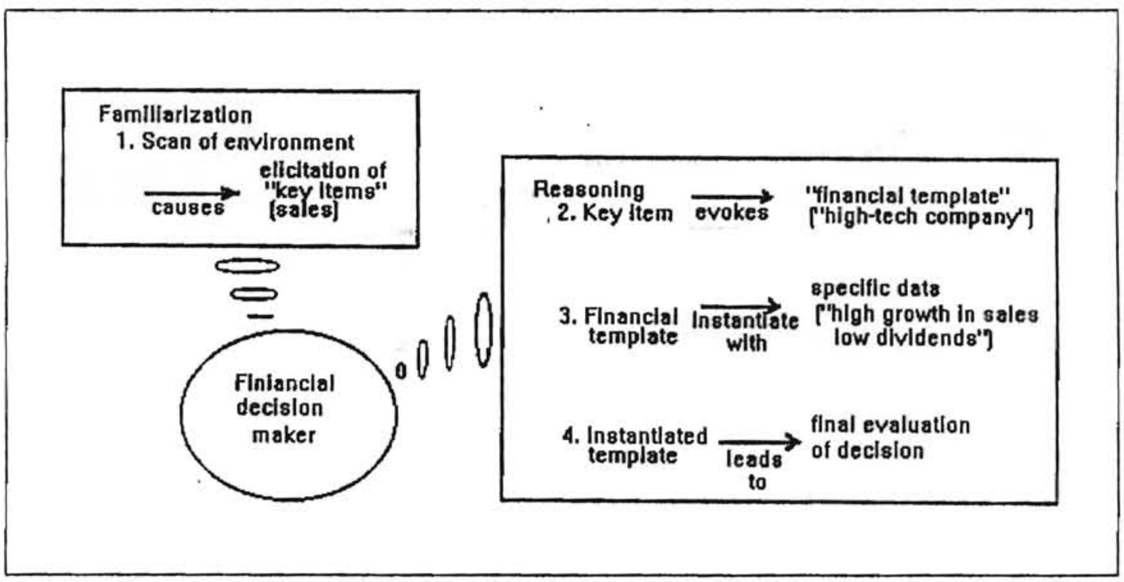

Figure 4 The Financial Decision Making Process

Bouwman's description of the financial decision process was developed based on several experimental studies of the process of financial decision making. Although this two-phase description probably should not be generalized, it can serve as a framework for analyzing issues involved in providing automated support for financial problem solving. For example, by using this framework, the SEC (the US Securities and Exchange Commission) developed the Financial Statement Analyzer (FSA) to analyze financial information contained in EDGAR's filings (Electronic Data Gathering, Analysis and Retrieval) [31]. Bouwman's financial decision process is also used as framework for the system design in this paper.

Based on Bouwman's description of the financial decision making process, this system design consists of mainly a mathematical model and several support expert systems. As shown in Figure 5, it is divided into four major components -- DATA, ANALYSIS, REASONING, and RECOMMENDATIONS.

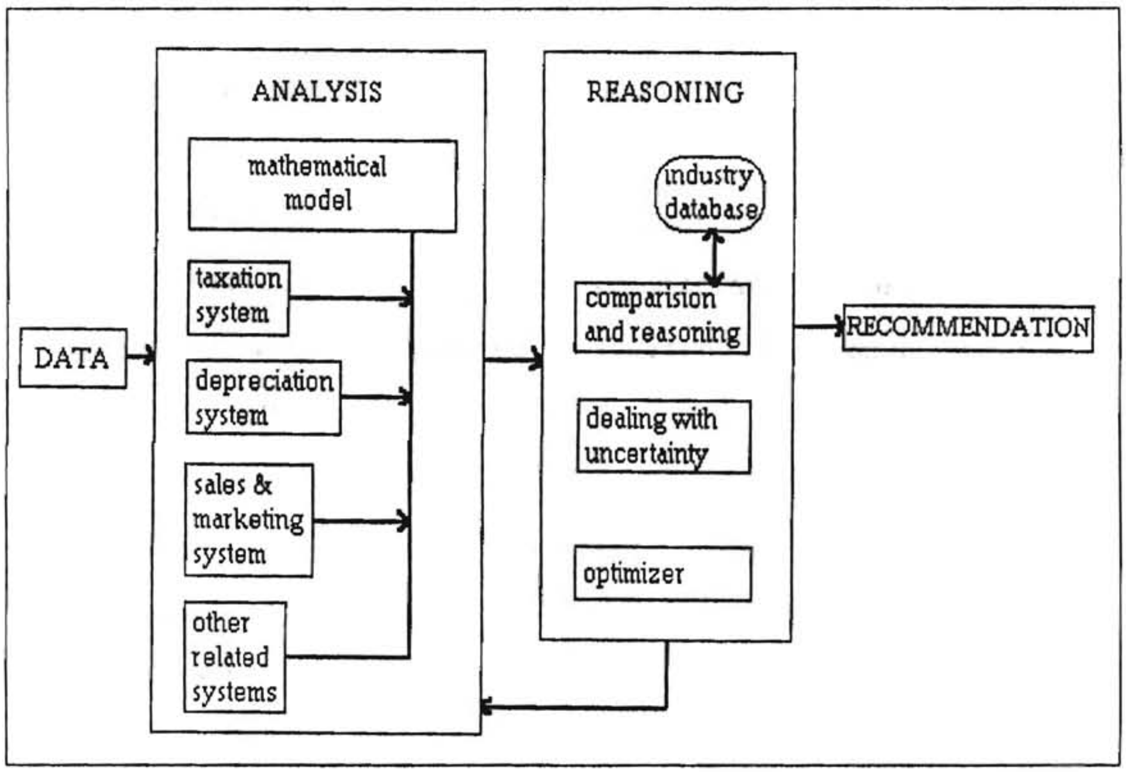

Figure 5 The System Design for an Making Investment Decision

DATA is used to collect data needed to make a financial judgment. The data includes items, such as cost, marketing and the format of the joint venture. All these related data serve as input to the next phase, ANALYSIS.

The ANALYSIS part of the system contains a mathematical model and support expert systems such as depreciation, taxation, and sales. Some accounting data from

DATA enters directly into the mathematical model. Others, such as the format of an organization and the characteristics of certain equipment, trigger the support expert systems. The results of reasoning the support expert systems provide the mathematical model with corresponding data. By using the support expert systems, default data or methods can also be provided. The outputs of ANALYSIS are a series of ratios and trends that reflect financial results from different aspects. This part of the system corresponds roughly to the familiarization phase of Bouwman's financial decision process.

At the stage of REASONING, the ratios and trends from ANALYSIS are first compared with those of major competitors and those of companies in the same industry. These data are available at the SEC. The comparison unveils any weaknesses of the adapted strategy that needs to be improved. Together with the goals of joint venture investment, these weaknesses serve as the starting point and trigger the support expert system to optimize the strategy. Through the backward chaining reasoning, improved strategy is sent back to ANALYSIS to reprocess. Since all the ratios and trends from ANALYSIS are based on estimates and forecasts, they may very well change if any of those estimates and forecasts changes. Thus, at the REASONING stage, the uncertainties of those ratios and trends are examined.

RECOMMENDATION is the final stage of the system. The system provides an optimized solution to the user and explains in detail the decision making process and submits disclosures with charts and graphs.

The Mathematical Model

Profitability Analysis

Four major methods are currently used to analyze the profitability of a project: payback period, net present value (NPV), internal rate of return (IRR), and profitability index (PI) [4, 16, 46].

Payback Period   Payback period is the first formal method used to evaluate capital projects. It is defined as the expected number of years required to recover the original investment. It is calculated based on the projected cash flow statement. It can be expressed as follows:

$$P_t = \text{Years before the year of complete recovery} + \frac{\text{Unrecovered Investment}}{\text{Cash flow during the year of complete recovery}} \qquad (3.2.1)$$

where P_t is the payback period in years; years before the year of complete recovery is calculated by taking the year's number (when accumulated cash flow becomes positive) minus 1; unrecovered investment is the absolute value of the accumulated net cash flow before the year of complete recovery; cash flow during the year of complete recovery is the projected cash flow from the projected cash flow statement.

Payback is a type of break-even calculation in the sense that, if cash flows come in as projected until the payback year, then the project will break-even. However, the regular payback method does not account for the cost of capital. No cost for the debt or equity used to undertake the project is reflected either in the cash flows or in the calculation. Its variant, the discounted payback period, does account for the cost of capital. It shows the break-even year after covering debt and equity cost. The discounted payback period is similar to the regular payback period, except that the expected cash flows are discounted by the project's cost of capital. Thus the discounted payback period is defined as the number of years required to recover the investment from discounted net cash flow.

Even though the discounted payback period solves the problem of cost of capital, still both payback methods have serious deficiencies. Both ignore the cash flows after payback. However, the payback method provides information on how long funds will be committed to a project. Thus, if other things hold constant, the shorter the payback period, the greater is the project's liquidity. Also since cash flows in the distant future are considered as being riskier than near-future cash flows, payback period is often used as a rough measure of the riskiness of a project. It is believed that the longer the payback period, the riskier the investment [4 p. 264, 16 p. 231].

Net Present Value (NPV) NPV is an improved method of evaluation that takes account of the cost of capital and the cash flows after payback. NPV is expressed as follows:

$$NPV = \Sigma^{n}_{t=1} CF_t / (1+k)^t = \Sigma^{n}_{t=1} (CI_t - CO_t) / (1+k)^t \qquad (3.2.2)$$

where CF_t is cash flow in year t, CI_t is cash inflow in year t, CO_t is cash outflow in year t, and k is the cost of capital.

To calculate the NPV of a project, the present value of each year's cash flow should be calculated first by discounting both cash inflows and outflows at the project's cost of capital. The summation of the discounted cash flows is defined as the project's NPV.

The rationale for the NPV method is straight forward. If a project has a positive NPV, then its cash flows are not only sufficient to repay the invested capital and provide the required rate of return on the capital, but also are generating excess return for the stockholders. NPV is the best method when comparing two mutually exclusive projects [4 p.274]. In situations when total investment is limited, the following rules should be followed (Table 1).

Situation	Criteria
(1) Equal amount of investment	Choose the project that provides the highest NPV of cash inflows
(2) Equal amount of cash inflows	Choose the project that requires the smallest NPV of cash outflows
(3) Unequal amount of cash inflows and outflows	Choose the project that has the highest net NPV

Table 1 The Criteria for Choosing a Project

The above criteria can be demonstrated by using the following example. Suppose there are two projects, project 1 and project 2. The NPVs of the two projects are as follows:

Project 1: $NPV_1 = \Sigma^{n}_{t=1}(CI_1 - CO_1)^t / (1+k)^t$

Project 2: $NPV_2 = \Sigma^{n}_{t=1}(CI_2 - CO_2)^t / (1+k)^t$

Situation 1: $\Sigma^{n}_{t=1}CO_{1t} / (1+k)^t = \Sigma^{n}_{t=1}CO_{2t} / (1+k)^t$

If $\Sigma^{n}_{t=1}CI_{1t} / (1+k)^t > \Sigma^{n}_{t=1}CI_{2t} / (1+k)^t$

Then $NPV_1 > NPV_2$

Project 1 is better then Project 2. Project 1 should be selected.

Situation 2: $\Sigma^{n}_{t=1}CI_{1t} / (1+k)^t = \Sigma^{n}_{t=1}CI_{2t} / (1+k)^t$

If $\Sigma^{n}_{t=1}CO_{1t} / (1+k)^{t} < \Sigma^{n}_{t=1}CO_{2t} / (1+k)^{t}$

Then $NPV_1 > NPV_2$

or $\Sigma^{n}_{t=1}(CI_1-CO_1) / (1+k)^{t} > \Sigma^{n}_{t=1}(CI_2-CO_2) / (1+k)^{t}$

Project 1 is better then project 2. Project 1 should be selected.

Situation 3: When comparing two projects with significantly different investment amounts and different returns, the project with the highest NPV should be picked. In practice, due to the shortage of funds, NPVR (Net Present Value Rate) should also be used when comparing two projects with significantly different investment amounts. To calculate the NPVR, divide the project's NPV by the present value of the project's total investment. The project with a higher ratio should be chosen. Sometimes, two competing projects have different life spans. To solve this problem, the least common multiple of the projects' life terms should be calculated. The calculated least common multiple should then be used as the life term for both projects in calculating NPVs.

Internal Rate of Return (IRR) IRR is defined as that discount rate, r, which equates the present value of a project's expected cash inflows to the present value of the project's expected cost.

PV (Inflows) = PV (Investment Costs)

or equivalently: $\Sigma^{n}_{t=0}CF_t / (1+r)^{t} = 0$

The value of each cash flow is a known factor while the value of r is unknown. Thus, this is an equation with one unknown and it can be solved for the value of r. The solution value of r is defined as the IRR.

To calculate the IRR, either the trial-and-error method or the interpolation method can be used. To use the trial-and-error method, the project's NPV is first calculated by using the cost of capital (R). If the NPV happens to equal to zero, then the cost of capital is also the IRR. However, if the NPV is greater than zero (positive), it means that IRR is greater than R. If the NPV is less then zero (negative), it means that IRR is less than R. Based on the first attempt, the discount rate should be adjusted and the project's NPV should be recalculated. This may repeat several times, till the discount rate (IRR) which equates the project's present value to zero is found. In reality, with some experience, one will find that usually no more than two trials are necessary, because the first result will show the direction of any refinement needed. After trying for the second time (a positive NPV by using R_1, and a negative NPV by using R_2), one may switch to the interpolation method. The basis of using the interpolation method is that the ratio of the difference between IRR and the lower discount rate R_1 and the difference between the two selected discount rates equals to the ratio of the difference between the positive NPV and NPV=0 and the difference between the two calculated NPVs.

Thus, $(IRR - R_1) : (R_2 - R_1) = (NPV_1 - NPV) : (NPV_1 - NPV_2)$

where R_1 is the lower discount rate selected, R_2 is the higher discount rate selected, NPV_1 is calculated by using R_1, NPV_2 is calculated by using R_2.

Therefore, $$IRR = R_1 + (R_2 - R_1) * NPV_1 / (NPV_1 - NPV_2) \qquad (3.2.3)$$

When using IRR, the decision criterion is to compare IRR to the cost of capital. If IRR is greater than the cost of capital, the project is acceptable. Otherwise, it should be rejected. If there are more than one mutually exclusive projects, choose the one with the maximum IRR (assuming all IRRs are greater than the cost of capital). The rationale for IRR is that if IRR exceeds the cost of capital used to finance the project, a surplus remains after paying for the cost of capital [4 p. 270, 46 p. 218].

IRR works fine for normal capital projects that have one or more cash outflows followed by a series of cash inflows. However, for the abnormal projects that have large cash outflows during or at the end of their lives, the IRR method encounters the problem of multiple IRRs. To solve this problem, a modified IRR (MIRR) is developed. MIRR is defined as PV Costs = PV of Terminal Value,

or $$\Sigma^{n}_{t=0} CO_t / (1+k)^t = \Sigma^{n}_{t=0} CI_t (1+k)^{n-t} / (1+MIRR)^n \qquad (3.2.4)$$

or $$\text{PV Costs} = TV/(1+MIRR)^n. \qquad (3.2.5)$$

The left term of the equation is simply the present value of investment outlays when discounted by the cost of capital. The numerator of the right term is the future value of the inflows at the end of the project's life term, assuming that the cash inflows are reinvested at the cost of capital. The compounded sum in the numerator is also called the terminal value (TV). The discounted rate that forces the PV of the TV to equal the PV of the costs is defined as the MIRR.

MIRR has a significant advantage over the regular IRR. It solves the problem of multiple IRRs. Since it assumes that cash inflows from the project are reinvested at the cost of capital that is generally more correct, the MIRR is a better indicator of a project's true profitability [4 p. 283].

Profitability Index Another method used to evaluate projects is the profitability index (PI) or the benefit / cost ratio.

$$PI = \text{PV Benefits} / \text{PV Cost}$$

$$= \Sigma^{n}_{t=0} CIF_t * (1+k)^{-t} / \Sigma^{n}_{t=0} COF_t * (1+k)^{-t} \qquad (3.2.6)$$

where CIF is the expected cash inflows or benefits, COF is the expected cash outflows or costs.

PI shows the relative profitability of any project, or the present value of benefits per present value dollar of costs. A project is acceptable if its PI is greater than 1, and the higher the PI, the higher the project's ranking [4 p. 270, 16 p. 244].

Break-even Analysis

Break-even analysis is a study about the relationship between sales income, cost, production volume, and sales volume. It determines the production (sales) volume where total sales income exactly equals total cost. The break-even point is the level of operation where total sales income equals total cost and net profit equals zero. When calculating the break-even point, it is assumed that production volume equals sales volume, production cost is the linear function of production volume or sales volume, fixed cost is a constant, unit variable cost is in direct proportion to production volume, unit price stays unchanged during different periods and different production levels, and sales income is the linear function of unit price and total number of units sold.

The break-even point can be calculated in terms of sales (production) volume or sales income. For projects that produce a single product, it is easier to calculate the break-even point in term of sales volume.

Break-even Sales Volume = Fixed Cost/(Unit Price - Unit Variable Cost)

= Fixed Cost / Unit Profit Contribution (3.2.7a)

However, for projects that produce multiple products, the differences between unit prices and unit variable costs vary. Therefore, it is much easier to calculate the break-even point in term of sales income.

If X is production (sales) volume, Y_I is sales income, F is fixed cost, P is unit price, V is unit variable cost, and Y_2 is production cost,

Then $Y_1 = PX$

$Y_2 = VX + F$

At the break-even point, $Y_1 = Y_2$

Therefore, $PX = VX + F$

And $BEP = X = F/(P-V)$ (3.2.7b)

Based on the formula, the break-even point is determined by the fixed cost and the difference between the unit price and the unit variable cost. This analysis helps to determine the effect of changes in unit price, fixed cost, and unit variable cost on the break-even point. According to the break-even analysis, if the project has a high break-even point, it needs a high sales (production) volume to break-even. The project easily would be affected by any change in production and the market. Similarly, the higher the project's fixed cost, the higher the break-even point, a situation detrimental to the project. Also, the greater the difference between the unit price and the unit variable cost, the lower the break-even point, because the project requires a lower production volume to stay in profit.

Ratio Analysis

Ratio analysis is the most popular technique used for financial statement analysis. It helps in studying the special relationships existing among financial data. Ratio analysis can be performed either on a cross-sectional or on a time-series basis. The objectives are to develop a norm that the entity should achieve. Many ratios can be derived from the financial statements. Financial ratios usually are divided into five categories: liquidity ratios, leverage ratios, profitability ratios, turn over or asset management ratios, and market-value ratios [45].

Liquidity Ratios Liquidity refers to the ability of an entity to meet its short-term financial obligations. Liquidity ratio includes both the current ratio and the quick ratio.

Current Ratio = Current Assets / Current Liabilities (3.2.8)

Current assets generally consist of cash, short-term marketable securities, accounts receivable, inventories, and prepaid items. Current liabilities typically include most of the items due within one year. It is believed that a 2:1 ratio should be maintained to ensure the entity's liquidity. However, the ratio would be more meaningful if it is compared with other entities' within the industry. A ratio smaller than 1 indicates that the entity may possibly have liquidity problem, yet a ratio that is much higher that the industry average indicates that funds have not been fully utilized.

Quick Ratio = (Cash + Accounts Receivable + Short-Term Marketable Securities) / Current Liabilities (3.2.9)

Some analysts believe that current ratio is not a very effective way to measure liquidity. Inventories are not necessarily as liquid as cash or net receivable. Therefore, quick ratio was derived to provide a numerator that represents greater liquidity. These assets are often called quick assets.

Leverage Ratios Leverage ratios examine the extent that non-equity funding (debt) is used by a firm and the long-term ability of the firm to meet its debt-service payment.

Debt to Equity Ratio = (Long-term Debt + Current Liability) / Shareholders' Equity (3.2.10)

This ratio measures the firm's ability to pay back its debt. It shows that among all the funds used by the firm, how much come from debt owners. Debt holders would prefer a smaller ratio that indicates that the risk of not recovering the debt is small.

Times Interest Earned = Operating Income / Interest Expense (3.2.11)

Times interest earned indicates the relative ability of a firm to pay the interest on its debt. This ratio emphasizes on the relationship between income and interest expense. Times interest earned is calculated by dividing operating income (income before deducting interest expense and tax expense) by interest expense.

Profitability Ratios Profitability ratios provide a summary view of the company's relative earnings for the period in question. They typically relate a measure of earnings to a normalizing measure of size.

Gross Profit Margin on Sales = Net Income / Sales (3.2.12)

The gross profit margin on sales reflects the economic benefit from sales. The net profit margin on sales reflects the net contribution of sales after deducting interest, tax, and other operating expenses.

Return on Total Assets = Net Income / Total Assets (3.2.13)

This is a comprehensive analysis of a project's profitability (including equity, current liability, and long-term debt) after it reaches the stage of normal production. The numerator consists of net income (gross income - interest expense - tax expense) and the denominator consists of total assets.

Return on Equity = Net Income / Common Equity (3.2.14)

This ratio indicates the effective utilization of common shareholders' equity. It is calculated by dividing net income available to common shareholders by common shareholders' equity.

Turn Over Ratios Turnover ratios are measures of the dynamics of the financial operations of the entity. They relate sales to items like total assets, inventory and accounts receivable to evaluate the utilization of these items.

Total Asset Turnover = Sales / Total Assets (3.2.15)

This ratio indicates the number of times that annual sales covers total assets.

Inventory Turnover = Sales / Average Inventory (3.2.16)

This ratio shows management's ability to control inventory. It is sales divided by inventory. While sales occur over the entire year and the inventory figure is for a point in time, it is better to use an average inventory figure to reflect the actual situation properly. One can divide the number of days in a year by the turnover rate to convert into days. It indicates the average number of days it will be taken to sell the inventory.

Accounts Receivable Turnover = Sales / Average Accounts Receivable (3.2.17)

This ratio indicates whether the balance of accounts receivable is valid and whether the collection of the credit sales is effective. To convert the turnover rate into days, one can divide the number of days in a year (normally use 360) by the turnover rate. This indicates the average number of days it will take to clear the accounts receivable.

Market-Value Ratios

PE Ratio = Market Price Per Share / Earnings Per Share (3.2.18)

PE (Price-Earnings) ratio shows how much investors are willing to pay per dollar of reported profits. Other factors holding constant, companies with better growth prospects have higher PE ratio. Similarly, the lower a company's risk, the higher its PE ratio, other factors holding constant.

Market to Book Ratio = Market Value of Stock / Book Value of Stock (3.2.19)

This ratio gives another indicator of how investors regard the company. The numerator of the formula is the market value of the stock at the end of the year, and the denominator is the total shareholder's equity divided by the number of outstanding shares.

Dealing with Uncertainty in the System Design

The main purpose of financial analysis is to provide a basis for the decision of whether to accept or to reject a planned project. Therefore the result of the financial analysis not only should meet the predetermined criteria, but also should be reliable. Yet financial analysis relies on predictions and estimates of future activities and lots of factors could contribute to errors in predictions and estimates: insufficient data, limitation in calculation method, limitation caused by unknown factors, existence of unmeasurable factors, unrealistic or inaccurate assumptions, major improvement in technology, changes in economic relations and economic structures, and unexpected economic and political conditions.

In order to reduce the impact of uncertain factors on financial indicators, an uncertainty analysis should be conducted. Uncertainty analysis both tests the project's ability to undertake risks and determines the reliability of economic and financial data. It helps management make sound business decision. Although there are many ways in dealing with uncertainty, such as probability, conditional probability, and fuzzy logic, this research adopts a unique approach that includes sensitivity, statistic, and risk analyses due to the following two characteristics of financial analysis. First, different projects have different sensitive factors. For example, the factor of raw material prices may have great impact on a manufacturing project, but have little impact on a hotel project. Therefore, it is necessary to identify sensitive factors for every project. Second, the projections of many uncertain factors (such as foreign exchange rate and interest rate) are beyond the scope of probability and fuzzy logic. Using statistic methods are more feasible.

Sensitivity Analysis

There are many factors that influence the performance of a project, such as production volume, product price, prices of major raw materials and labor, investment in fixed assets, construction period, sales tax, and exchange rate. Sensitivity analysis identifies sensitive factors and analyzes the impact on the financial indicators when one of those factors changed or when several of those factors changed simultaneously. NPV and IRR are major indicators of a project's economic value. The changes in financial data are reflected in the changes of NPV and IRR.

In sensitivity analysis, a slight change in some factors causes major changes in financial indicators. These factors are called sensitive factors. Other factors, when changed, cause slight or unnoticeable changes in financial indicators. Those factors are labeled as insensitive factors. The degree of sensitivity of the project can be shown as changes in percentage in financial indicators caused by changes in percentage of the sensitive factors and can also be shown as percentage of change allowed in major factors when financial indicators equal their comparing competitors' financial indicators.

When doing sensitivity analysis, one should first identify the financial indicators that need to be studied. In practice, different types of projects have different types of major factors. It is unnecessary to evaluate every single major factor. Only those sensitive factors that have significant impacts on construction period and economic life time of the projects need to be evaluated. After a brief search for sensitive factors, the financial indicators should be tested. By changing one sensitive factor at a time, recalculating the IRR, NPV, or other indicators, one can analyze the impact of each of the factors.

Uncertainty is generally the source of risk. However, the degree of risk caused by each sensitive factor varies. The uncertainty of a highly sensitive factor is riskier than the uncertainty of a less sensitive factor. Although sensitivity analysis can study the influence of those uncertain factors, and can identify the degree of sensitivity of financial indicators to different uncertain factors, it cannot determine the expected value of financial indicators.

Statistical Analysis

Sensitivity analysis studies the impact of changing factors on NPV and IRR by isolating each of the variable factors. The levels of sensitivity of each of the variable factors differ, so the probabilities of realizing those changes also vary. For some factors, the probability of change has a direct relationship with the level of sensitivity. For others, probability of change has an inverse relationship with the level of sensitivity. Therefore, sensitivity analysis cannot capture everything, but statistic analysis is a good supplement to sensitivity analysis. Once the probabilities of change of those sensitive factors are determined, it is easier to analyze the financial indicators.

Expected Value of NPV The expected value of NPV is the weighted average return of a long-term investment after taking full consideration of all possible circumstances. It is calculated under the assumption that all the probable values of uncertain factors and the probabilities of their occurrences are known.

If the probable values of an uncertain factor A of a project are $A_1, A_2, ..., A_t$, their corresponding probabilities are $P(A_1), P(A_2), ..., P(A_t)$, then the expected value of A is $E(A) = \Sigma^n_{t=1} A_t P(A_t)$. (3.3.1)

In calculating the expected value of NPV, the most critical part is to calculate the expected values of each years' net cash flow. Once the expected values of each years' cash flow are known, expected value of NPV can be calculated by using the formula of NPV.

According to the above formula $E(A) = \Sigma^n_{t=1} A_t P(A_t)$, the expected value of each year's cash flow is $Z[Y_t] = \Sigma^m_{i=1} Y_{ti} P(Y_{ti})$, where $Z[Y_t]$ is the expected value of net cash flow in year t, Y_{ti} is the possible net cash flow in year t, $P(Y_{ti})$ is the corresponding probability of Y_{ti}, m is the number of possible situations. Based on the formula of NPV, the expected value of NPV is $E[NPV] = \Sigma^n_{t=1} Z[Y_t]*(1+i)^{-t}$ where E(NPV) is the expected net present value, and n is the term in years.

Standard Deviation Expected value is the average value of a randomly changeable variable factor. There is difference between the average value (expected value) and the actual value. Standard deviation is thus used to determine the degree of deviation of the expected value from the actual value.

$$\sigma = \pm \{E(A^2) - [E(A)]^2\}^{1/2} \quad (3.3.2)$$

where σ is the standard deviation,

$$E(A^2) = \Sigma^n_{t=1} A^2_t P(A_t).$$

The above formula can also be expressed as

$$\sigma = \pm \{\Sigma^n_{t=1} [E(A) - A_t]^2 * P(A_t)\}^{1/2}$$

Some projects have both high expected NPV and large standard deviation. So if the worst scenario came true, the project would suffer great loss. Therefore, the chosen project not only should have higher expected net present value, but also a smaller standard deviation than other projects.

Risk Analysis

Expected Return Analysis Risk can be defined as the degree of variation in the actual versus expected return of a project. The wider the possible deviation, the greater the risk [16 p. 253]. Expected return analysis evaluates the project by using expected return and standard deviation in different situations.

$$E(R) = \Sigma^{n}_{t=1} \rho_t R_t \qquad (3.3.3)$$

$$\sigma_R = \{\Sigma^{n}_{t=1} \rho_t [R_t - E(R)]^2\}^{1/2} \qquad (3.3.4)$$

where R_t is the return of the project in situation t, ρ_t is the probability in situation t, E(R) is the expected return, σ_R is the standard deviation of expected return, n is the number of possible situations.

The criterion for selecting a project is that the project with the highest expected value, or with the smallest standard deviation, should be selected. In real world, there could be contradictory results: Project A may have a higher expected value than Project B, but Project B may have a smaller standard deviation than Project A. In this situation, the expected value is not a good evaluation method. Sometimes, in order to understand the worst/best scenario better, the range of expected values is calculated.

$$R_g = R_{max} - R_{min} \qquad (3.3.5)$$

where R_g is the range of distribution of expected return, R_{max} is the probable maximum return, and R_{min} is the probable minimum return.

If the range of expected return is large, it means that the project's return is unstable, because if the worst/best scenario comes true, the actual return is well below/above the expected return. Conversely, if the range is small, it means that the project's return is pretty stable. The actual return is close to the expected return.

Adjusting Cash Flow There are differences in project risks due to differences in project types or accuracy in predicting cash flow. The differences should be identified and project risks should be adjusted by multiplying an adjusting index.

$$CE = \Sigma^{n}_{t=0} (\alpha F_t) / (1+R_f)^t \qquad (3.3.6)$$

where F_t is net income (cost) in year t, α is the adjusting index ($0 < \alpha < 1$), R_f is the risk-free discount rate, CE is the NPV after considering risks of the project.

The advantage of this method is that probability is not required in the calculation. It is critical to determine the correct adjusting index that is determined based on experience and is hard to estimate, because it can change with time.

Adjusting Discount Rate Adjusting the discount rate is another way of adjusting the differences in project risks. When calculating net present value, one should use larger discount rates for projects with higher risks and smaller discount rates for projects with lower risks, because a larger discount rate leads to a more conservative result. The adjusted discount rates are calculated by adding an adjusting rate to the risk-free discount rate.

$$R' = R_f + a \qquad (3.3.7)$$

where R' is the adjusted discount rate, R_f is the risk-free discount rate, a is the adjusting rate ($0 < a < 1$).

The above formula indicates that the adjusted discount rate is greater than the risk-free discount rate. The adjusted discount rate method uses the same adjusting rate through out the project's life. It does not change with time and it does not need to know the probability rate. The adjusting rate also is determined based on experience that is inherently inaccurate.

Expert System Designs

Making financial investment decisions usually involves knowledge in many different areas, such as tax, depreciation, cost, marketing and sales. Each of these areas individually is well defined and each has its own characteristics. To properly handle various differences, two kinds of expert systems are used in this system design. One is the forwarding chaining expert system that draws conclusion from available information. For example, a depreciation expert system can choose a suitable depreciation method for an equipment according to its characteristics. The other is the backward chaining expert system that can be used to optimize the investment strategies. This paper suggests two expert system design methods that easily can be used to construct those relatively small expert systems. These two methods can be implemented in C, as well as other languages. Together with the mathematical model, the whole system can be implemented on almost any PC or mainframe.

The Method of Forward Chaining Expert System Design

Forward chaining is also called bottom-up reasoning, because it reasons from the lower-level (evidence and facts) to the top level (conclusions that are based on the facts). Forward chaining is a data driven approach. It starts from available conditions as they come in. For each condition, the system searches the knowledge base for rules that match the condition in the IF part. Each of those rules can in turn generate new condition from the conclusions of the invoked THEN part. These new conditions are added to the existing conditions. After all the conditions have been examined, the system works its way towards a conclusion. In this paper, the forward chaining design is based on the following data structures and algorithm proposed by Levine [21].

Data Structures The forward chaining expert system contains four data structures. In order to illustrate the data structures of the forward chaining expert system design, a sample knowledge base and a number of tables derived from the sample knowledge base are introduced in Figure 6.

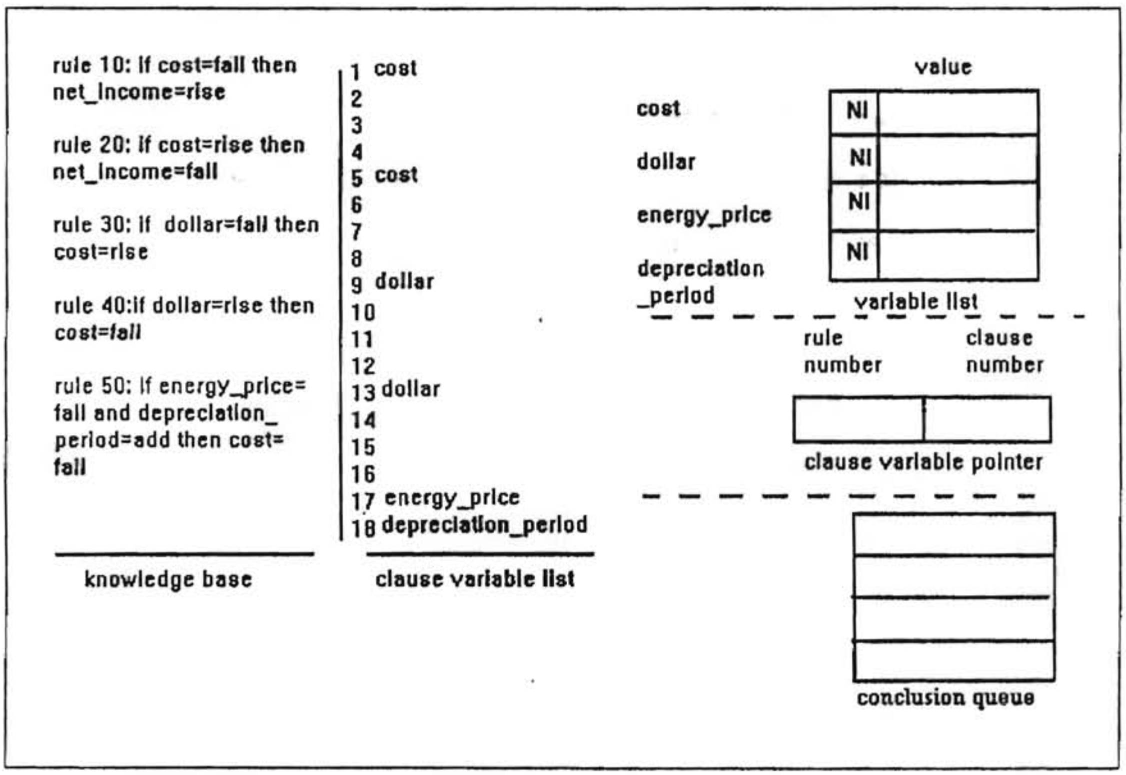

Figure 6 Data Structures and a Sample Knowledge Base for a FCESD

The sample knowledge base is a simplified model constructed to analyze the changes in net income of a joint venture company. Suppose there is an international joint venture company in the United States. It imports raw materials from Germany. Rule 10 states that if cost falls, then net income will rise. Rule 20 states that if cost rises, then net income will fall. Since the company needs to import raw materials from Germany, foreign exchange rate is crucial to its cost. Rule 30 states that if the exchange rate rises (A strong Dollar versus a weak Mark), then the cost of buying raw materials in Germany will fall. Rule 40 states that if the exchange rate falls (A weak Dollar versus a strong Mark), then the cost of buying raw materials will rise. The longer the depreciation period, the lower each year's depreciation expense. Rule 50 states that if energy price falls and depreciation period is extended, then cost will fall.

The first data structure used is the clause variable list. This list shows which variables in the problem are associated with the IF parts of specific IF-THEN statements. The numbers to the left of the variable names represent the array location in the data structure where the variable names are placed. In this design there are four array locations reserved for each rule. If a rule does not utilize all of these locations, they are left blank.

The second data structure is the conclusion variable queue. The conclusion variable queue is used to keep track of the variables that are or will be dealt with. For example, (Figure 6), the inference engine examines whether there is any statement that contains DOLLAR = FALL. If there is, the THEN part of the statement is invoked and new conditional variable COST is initiated. After processing all the rules containing DOLLAR, the effect of rising cost also is examined. To save this new variable, COST is placed in the conclusion queue after DOLLAR. After all the IF-THEN rules that contain DOLLAR in the IF part have been processed, variable DOLLAR is removed from the conclusion variable queue. Variable COST automatically moves to the front of the queue and is processed just like the variable DOLLAR was processed.

The third data structure is the variable list. Before any question is asked, none of the variables have been instantiated. The variable list is used to show whether a variable has been instantiated or not.

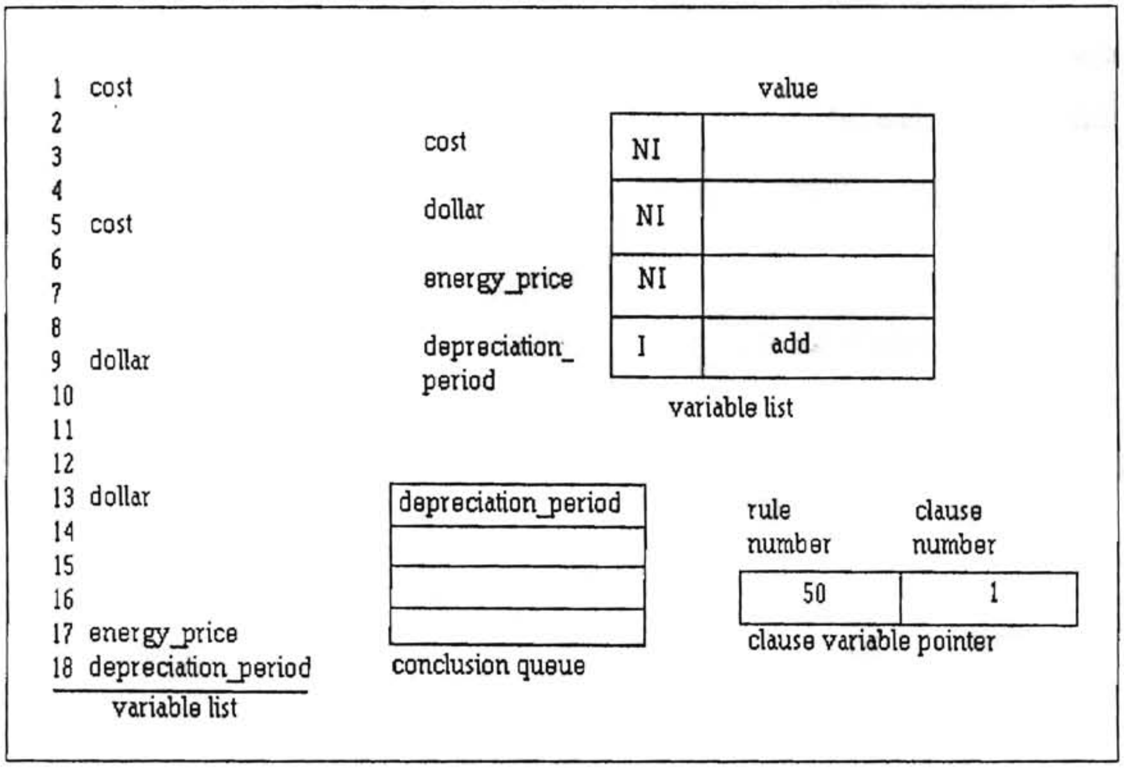

Figure 7 FCESD Illustration 1

The last data structure is the clause variable pointer. It keeps track of the clause within the rule that is being examined (Figure 7). It is made up of the rule number and the clause number. The processing of each clause is a function of the variable in the front of the conclusion variable queue. The pointer actually points to an entry in the clause variable list. That entry contains the condition variable that is being processed and that is at the front of the conclusion variable queue.

Algorithm Using the above data structures, the following is the algorithm used for forward chaining expert system design.

(1) The condition is identified.

(2) The condition variable is placed on the conclusion variable queue and its value is marked on the variable list.

(3) The clause variable list is searched for the variable whose name is the same as the one in the front of the queue. If found, the rule number and a 1 are placed into the clause variable pointer. If not found, go to step 6.

(4) Each variable in the IF clause of the rule that is not already instantiated is now instantiated. The variables are in the clause variable list. If all the clauses are true, the THEN part is invoked.

(5) The instantiated THEN part of the variable is placed in the back of the conclusion variable queue.

(6) When there are no more IF statements containing the variable that is at the front of the conclusion variable queue, that variable is removed.

(7) If there are no more variables on the conclusion variable queue, the session ends. If there are more variables, go to step 3.

To illustrate the usage of the above algorithm, the following is an example of examining a specific situation -- What is the effect of using longer depreciation period (that is DEPRECIATION_PERIOD = ADD).

After the forward chaining engine finishes reading the knowledge base, it constructs lists, pointers, and queues. The conclusion queue initially contains DEPRECIATION_PERIOD whose value is ADD. Next, the engine searches the clause variable list for the first instance of DEPRECIATION_PERIOD and finds that it is in Rule 50. The clause variable pointer is therefore set to Rule 50 Clause 1, (Figure 7). From the clause variable list, there are two variables ENERGY_PRICE and

DEPRECIATION_PERIOD. Since the original condition, DEPRECIATION_PERIOD, has already been instantiated to ADD and the engine cannot find the value of ENERGY_ PRICE, the system asks the user for the value of ENERGY_PRICE. Suppose user's response is FALL, then ENERGY_PRICE is instantiated in the variable list. The clause number of the clause variable pointer is then incremented from 1 to 2. Since both conditions (ENERGY_PRICE = FALL and DEPRECIATION_PERIOD = ADD) are true, the THEN part of Rule 50 is invoked and results in COST = FALL. Since COST represents a new condition, it is placed in the conclusion variable queue (Figure 8), and the variable list is also updated (Figure 9). Since there are no more rules left with DEPRECIATION_PERIOD, DEPRECIATION_PERIOD is removed from the conclusion queue. COST moves to the front. The engine then searches the clause variable list for COST, just as it did to find DEPRECIATION _PERIOD. The result is shown in Figure 10.

Although both IF parts of Rule 10 and Rule 20 contain variable COST, only the THEN part of Rule 10 is invoked when the value of COST = FALL. Rule 20 is never invoked. This results in NET_INCOME being placed on the conclusion queue list (Figure 11). COST is terminated from conclusion queue, leaving NET_INCOME at the front of the queue. Since NET_INCOME is not in the IF part of any rule, the session ends. The answer to the question of using a longer depreciation period is that net income will increase.

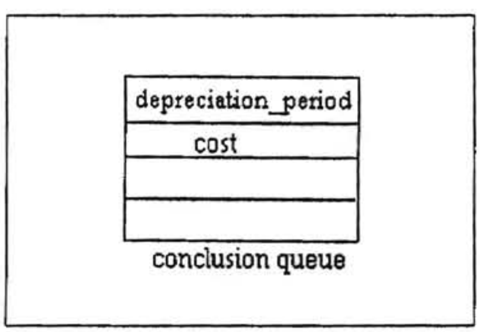

Figure 8 FCESD Illustration 2

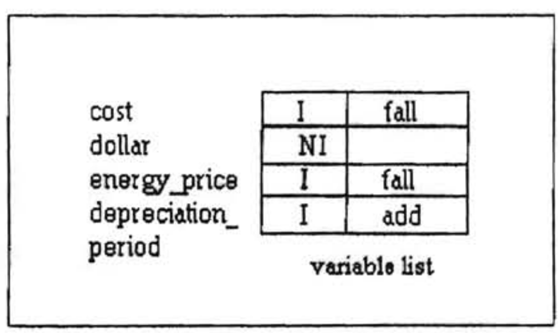

Figure 9 FCESD Illustration 3

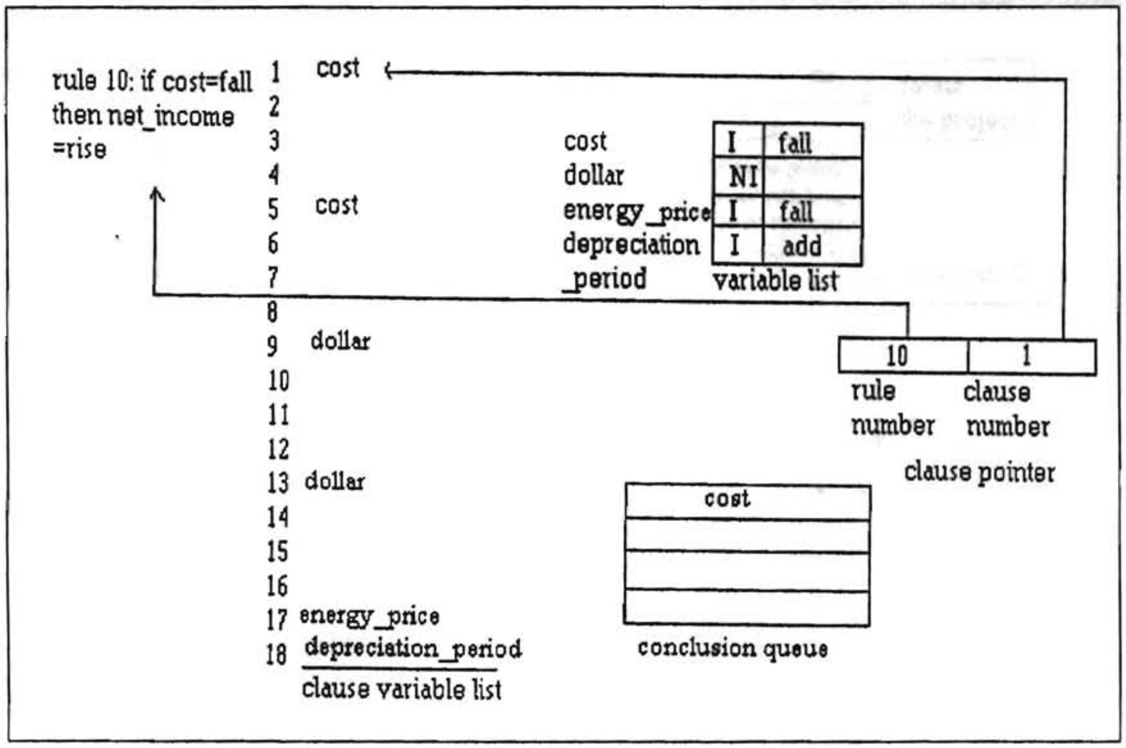

Figure 10 FCESD Illustration 4

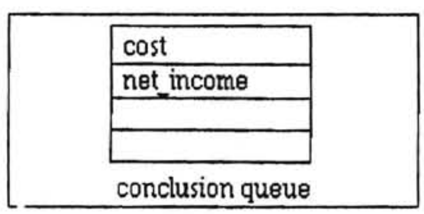

Figure 11 FCESD Illustration 5

The Method of Backward Chaining Expert System Design

Backward chaining reverses the process of forward chaining. It is a goal-driven approach that starts from an expectation of what is going to happen, then seeks evidences that support (or contradict) the expectation. Since the goal-driven approach is complex and difficult to conceptualize, decision tree is used to construct backward chaining knowledge base. Decision tree method illustrates the problem clearly,

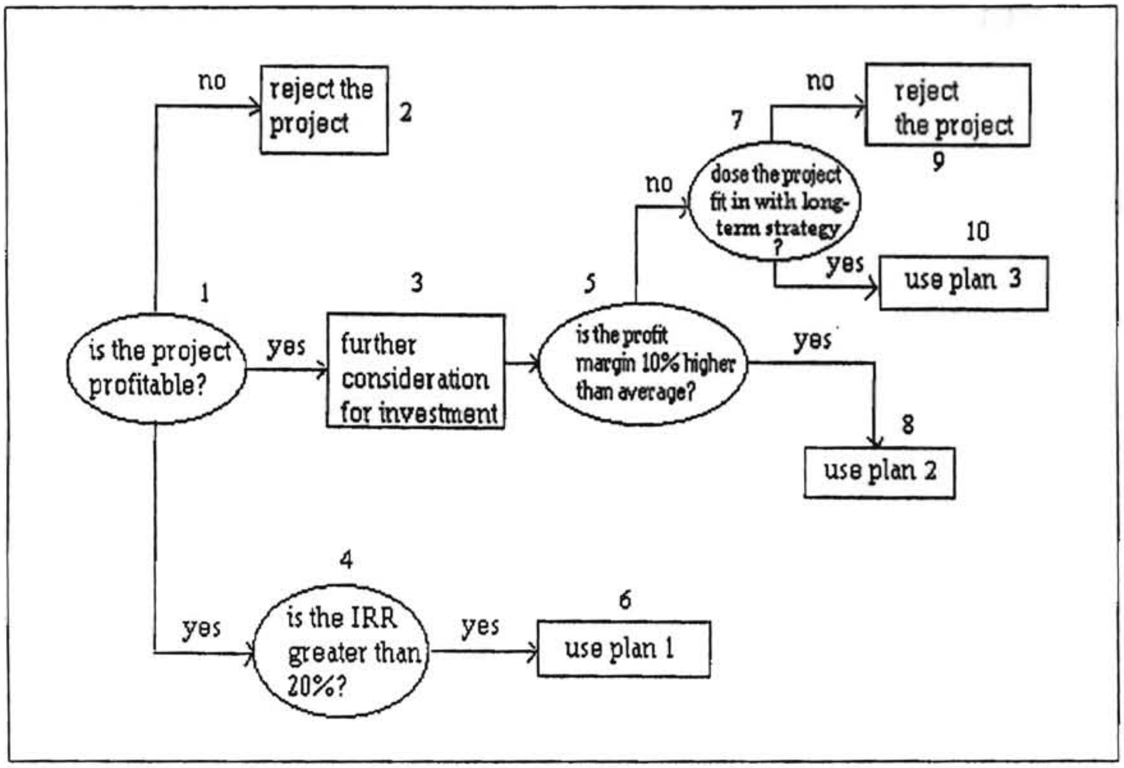

Figure 12 A Sample Decision Tree for a BCESD

because it can capture all the factors that must be considered in reaching a decision and can demonstrate how one consideration leads to another graphically. Figure 12 is a sample decision tree used to make investment decision. At the end of each branch of the decision tree is a conclusion, in this example, the decision of whether or not to invest in a project and if so, which plan should be used. The circles and rectangles are defined as nodes. The numbers in the nodes serve as references. The circle nodes are decision nodes and the rectangle nodes are used to signify conclusions or sub-conclusions. The arrow lines are called branches. They designate the direction of the diagram. As shown in Figure 12, the following six rules cover paths that lead to every goal of the investment decision tree.

10 IF PROFIT = NO 1.2

THEN INVESTMENT = NO

20 IF PROFIT = YES 1.3

THEN CONSIDERATION = YES

30 IF PROFIT = YES and IRR > 20% 1.4.6

THEN INVESTMENT = PLAN A

40 IF CONSIDERATION = YES and 3.5.7.10

MARGIN < 1.1 * AVERAGE and

STRATEGY = YES

THEN INVESTMENT = PLAN C

50 IF CONSIDERATION = YES and 3.5.7.9

MARGIN <= 1.1 * AVERAGE and

STRATEGY = NO

THEN INVESTMENT = NO

60 IF CONSIDERATION = YES and 3.5.8

MARGIN > 1.1 * AVERAGE

THEN INVESTMENT = PLAN B

Rule 10 states that if a project is not profitable, then the project will not be considered. Rule 20 states that if a project is profitable, then the project will be considered for investment. Rule 30 states that if a project is profitable and its internal rate of return is greater than 20%, then invest in the project using Plan A. Rule 40 states that if a project is considerable, its profit margin is less then 1.1 time the industry average, and the project fits in with the company's long term strategy, then invest in the project using Plan C. Rule 50 states that if a project is considerable for investment, the profit margin is less than and equal to 1.1 time the industry average, and it does not fit in with the company's long term strategy, then do not invest in the project. Rule 60 states that if a project is considerable, and the profit margin is greater than 1.1 time the industry average, then invest in the project using Plan B.

In this paper, the backward chaining design is based on the following data structures and algorithm that is also proposed by Levine[21].

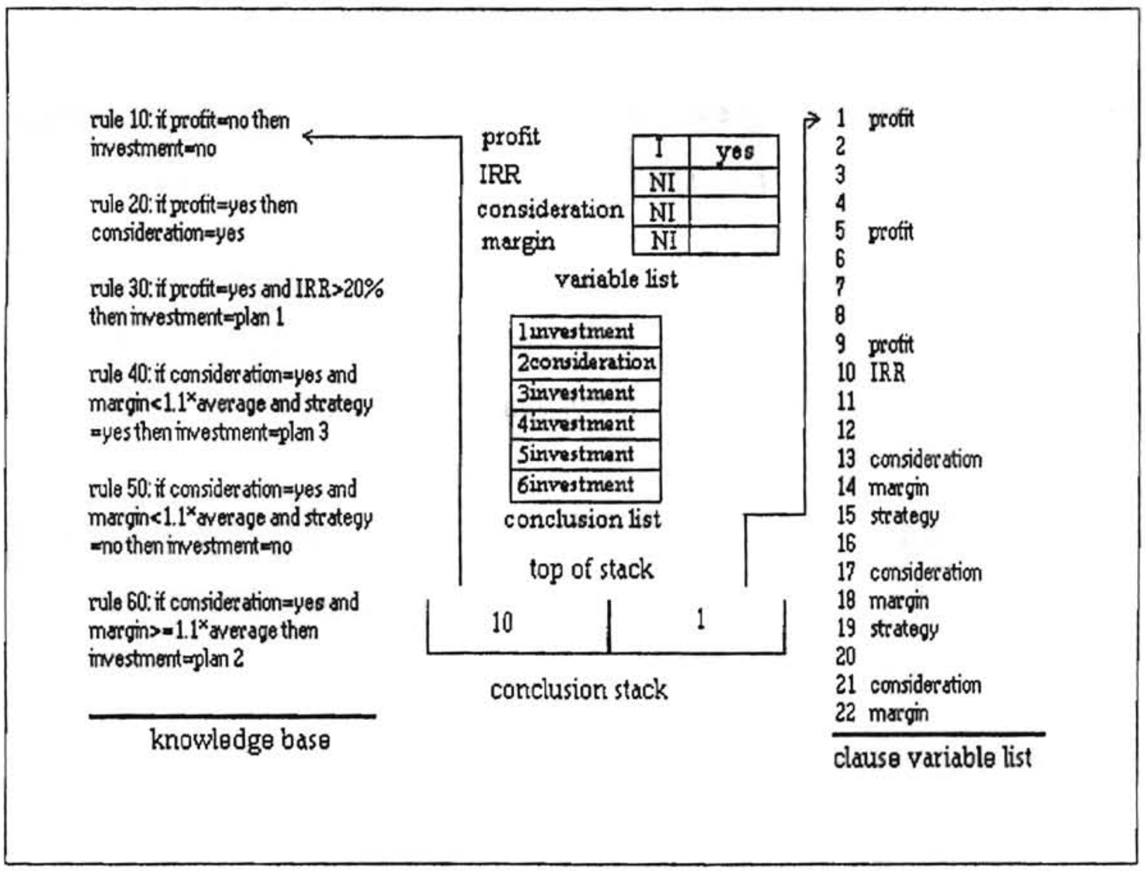

Figure 13 Data Structures and a Sample Knowledge Base for BCESD

Data Structures To illustrate the data structure of backward chaining expert system design, the sample knowledge base obtained from decision tree and a number of tables derived from the sample knowledge base are introduced in Figure 13.

The first data structure, the conclusion list, lists all the possible conclusions in sequential order. Each entry in the conclusion list contains three items: the rule number, the conclusion associated with that rule number, and a set of conditions that lead to the conclusion. This structure is used to locate a conclusion by its corresponding rule number: If the IF part of the rule is true, then the THEN part is invoked, and the conclusion is instantiated.

The second data structure is the variable list. Each entry in the structure has two parts. They are variable names that include all the variables in the IF parts of the knowledge base, and instantiation indicators that indicate whether or not a variable is instantiated. The variable list has two constraints: (1) each variable can appear at most once in the list, and (2) if the variable is included in the variable list, it can not appear in the conclusion list.

The third data structure is the clause-variable list that contains the list of variables for each IF part of IF-THEN rules. In this system design, an IF statement can contain a maximum of four variables. These variables are connected by logical operators AND, OR, or NOT. If a rule does not use all four locations, they are left blank. Although the limit of four variables can be adjusted, allocating the same number of locations to each rule reduces the complexity of programming.

The last data structure is the conclusion stack. It ties together the other three structures and indicates which IF-THEN statement contains the conclusion that the inference engine is trying to reach, and which clause in the IF portion is being examined.

Algorithm Using the above data structures, the following is the algorithm for backward chaining expert system design.

(1) Identify the conclusion.

(2) Search the conclusion list for the first instance of the conclusion's name. If found, place the rule on the conclusion stack using the rule number and a 1 to represent the clause number. If not found, notify the user that an answer cannot be found.

(3) Instantiate the IF clause (i.e., each condition variable) of the statement.

(4) If one of the IF clause variables is not instantiated, as indicated by the variable list, and is not on the conclusion list, ask the user to enter a value.

(5) If one of the clauses is a conclusion variable, place the conclusion variable's rule number on the top of the stack and go back to step 3.

(6) If the statement on top of the stack cannot be instantiated using the present IF-THEN statement, remove the statement from the top of the stack and search the conclusion list for another instance of that conclusion variable's name.

(7) If such a statement is found, go back to step 3.

(8) If there are no more conclusions left on the conclusion stack with that name, the rule for the previous conclusion is false. If there is no previous conclusion, then notify the user that an answer cannot be found. If there is a previous conclusion, go back to step 6.

(9) If the rule on top of the stack can be instantiated, remove it from the stack. If another conclusion variable is underneath, increment the clause number, and for the remaining clauses go back to step 3. If no other

conclusion variable is underneath, the question is answered. The user gets a conclusion.

To illustrate the idea of backward chaining design, the following example uses backward chaining to inference the sample knowledge base.

After receiving the initial question -- should one invest in this project, the inference engine finds the conclusion variable INVESTMENT from the conclusion list, then installs the number of the next rule that contains INVESTMENT on top of the conclusion stack. Since this is the first time INVESTMENT has been encountered, the inference engine starts from the very beginning. It finds that Rule 10 contains the conclusion variable INVESTMENT. Consequently, the conclusion stack then installs the number 10 to indicate Rule 10 Clause 1. By looking at the clause-variable list, the inference engine finds just one uninstantiated variable, PROFIT, for Rule 10. Since PROFIT has no value at this time and is not in the conclusion list, the system asks user to provide a value. Suppose user's answer is YES, the value of PROFIT is thus installed as shown in Figure 13.

Since PROFIT = NO, Rule 10 cannot be invoked, and must be removed from the conclusion stack. The inference engine continues to search for the next rule that uses INVESTMENT as a conclusion variable. This is found in Rule 30. The inference engine places Rule 30 on top of the conclusion stack (Figure 14) and tries to install all the variables of Rule 30 in the clause variable list. The engine first checks PROFIT and finds it has been instantiated, then it increments the clause number by 1 to check whether IRR has been installed. Since IRR is neither installed, nor in the conclusion list, the system asks the user to provide the value of IRR. Suppose the answer is 15%, and since one of the conditions in Rule 30 is not true, it cannot be invoked, and must be removed from conclusion stack. By continuing to search the remainder of the conclusion list, the engine finds that INVESTMENT is a conclusion variable of Rule 40. The engine places Rule 40 on top of the conclusion stack (Figure 15).

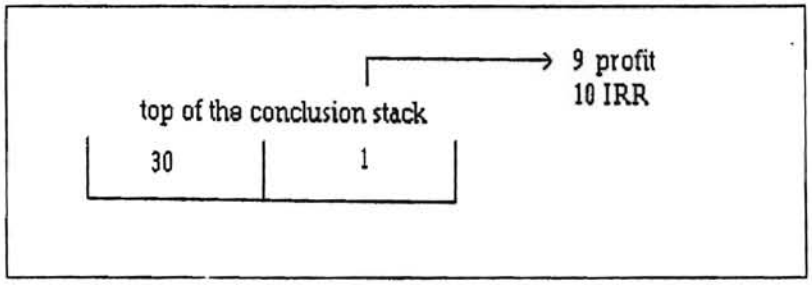

Figure 14 BCESD Illustration 1

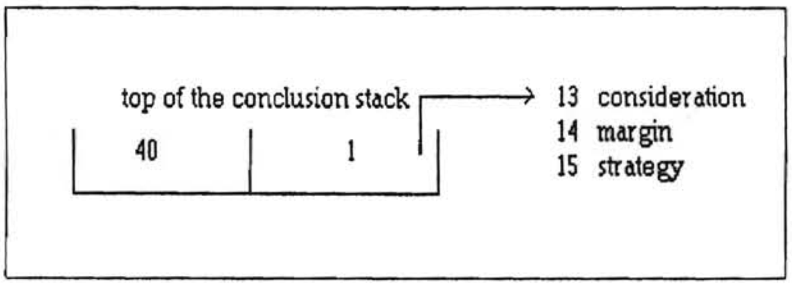

Figure 15 BCESD Illustration 2

The first clause of Rule 40 is CONSIDERATION. CONSIDERATION has not been installed in the variable list, but it is in the conclusion list. It is the conclusion variable of rule 20. Therefore, the engine places Rule 20 on top of the conclusion stack (Figure 16). Since the value of PROFIT already has been set to YES, Rule 20 can now be executed and CONSIDERATION equals YES. After Rule 20 has been executed, it is removed from the conclusion stack. Rule 40 thus gets back on top of the conclusion stack. The clause number is incremented to 2. Since MARGIN has not been instantiated, and is not in the conclusion list, the system asks the user to provide the value. The user's answer, 7%, is installed as the value of MARGIN. The clause number is now incremented to 3. The third clause, STRATEGY, has not been installed, and neither is it in the conclusion list. The system asks the user to provide the value of STRATEGY. If the answer is YES, then Rule 40 is invoked, because all the clauses in Rule 40 are true.

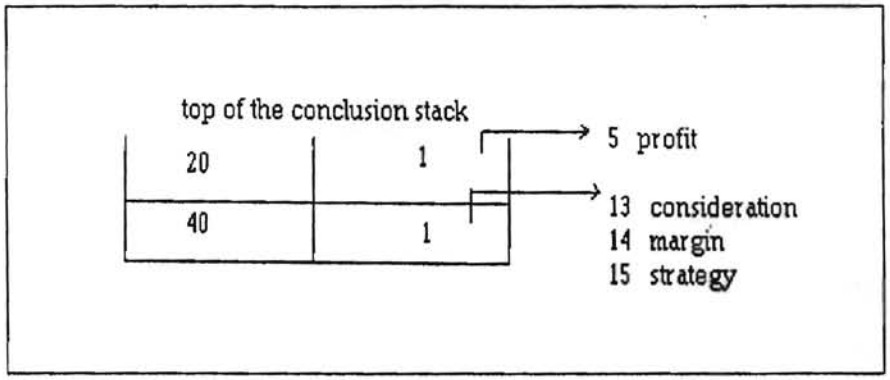

Figure 16 BCESD Illustration 3

Since there are no more rules that use INVESTMENT as a conclusion variable, INVESTMENT is instantiated to PLAN C. The goal has been reached.

CHAPTER IV

THE EXPERIMENT OF COMBINING A MATHEMATICAL MODEL WITH TWO EXPERT SYSTEMS

The purpose of this experiment is to demonstrate the possibility of combining a mathematical model with different expert systems to solve complex decision problems. The whole experiment includes a mathematical model, a forward chaining expert system and a backward chaining expert system. The input data are from the financial feasibility report of a joint venture software company. The experiment is implemented on the Sequent S81 machine of the Computer Science Department of Oklahoma State University. The programming language used is C.

Item\Year	0	1	2	3	4	5	6	7	8	9	...	...	n
1.Net Sales													
2.Depreciation Exp.													
3.Residual Value													
4.Know How													
5.Cash Inflow													
6.Investment													
7.Tax Expense													
8.Variable Cost													
9.Fixed Cost													
10.Cash Outflow													
11.Net Cash Flow													
12.Discount Rate													
13.Present Value													

Table 2 The Format of Cash Flow Statement

The mathematical model is constructed to analyze financial statements, and to calculate NPV and IRR. The calculations of NPV and IRR are based on the projected cash flow statement. In this experiment, the format of cash flow statement is shown in Table 2. In this table, net sales (row 1), know how (row 4), investment (row 6), tax expense (row 7), variable cost (row 8), and fixed cost (row 9) are from other financial statements. Since showing the formats of these statements is beyond the purpose of this experiment, these data are treated simply as inputs of the cash flow statement. Depreciation expense (Row 2) and residual value (row 3) from the depreciation schedule are the results of running the depreciation expert system. The details of the depreciation schedule and the expert system will be explained shortly. Cash inflow (Row 5) is calculated by adding net sales (Row 1), depreciation expense (Row 2), residual value (Row 3), and know how (Row 4). Cash outflow (Row 10) is calculated by adding investment (Row 6), tax expense (Row 7), variable cost (Row 8), and fixed cost (Row 9). Net cash flow (Row 11) is calculated by subtracting cash outflow (Row 10) from cash inflow (Row 5). Discount rate (Row 12) is an empirical value that is based on the interest rate and the riskiness of the project. NPV and IRR are calculated by using formula 3.2.2 and formula 3.2.3 respectively.

The forward chaining expert system is constructed to choose depreciation methods for fixed assets based on their characteristics. The knowledge base used in this forward chaining expert system contains the following rules:

Rule 10: IF the usage of an asset can be divided into units, and the usage of an asset can be measured by units consumed, THEN use the activity method.

Rule 20: IF no specific description of an asset, THEN use the straight-line method.

Rule 30: IF the productivity of an asset decreases evenly by year or by month, THEN use the straight-line method.

Rule 40: IF the productivity of an asset decreases faster during its earlier periods, THEN use the sum-of-the-years'-digits method.

Rule 50: IF the productivity of an asset decreases steadily, and maintenance cost rises in later periods, THEN use the declining-balance method.

The formulas for the four different depreciation methods are listed in Table 3 [17 pp. 543-548].

Depreciation Methods	Formulas
Activity Method	Depreciation Expense = (Cost - Residual Value) * Hours Used in This Year / Total Estimated Hours.
Straight-Line Method	Depreciation Expense = (Cost - Residual value) / Estimated Service Life (in years).
Sum-of-the-Years'-Digits Method	Depreciation expense decreases year by year based on a decreasing fraction of depreciation cost (original cost less salvage value). Each fraction uses the sum of the years as a denominator and the number of years of estimated life remaining as of the beginning of the year as a numerator. The numerator decreases year by year while the denominator remains constant.
Declining-Balance Method	Depreciation Expense = Depreciation Rate (expressed as a percentage that is some multiple of the straight-line method) * Book Value (of the asset at the beginning of each period).

Table 3 Depreciation Methods and Formulas

For each fixed asset, the depreciation expert system asks questions about its characteristics, then chooses a depreciation method for that asset. According to the principals and characteristics of different assets, the expert system constructs depreciation schedules. The format of the depreciation schedule used in this experiment is shown in Table 4.

Item\Year	0	1	2	3	4	5	6	7	8	9	......	n
Computer and Electronic Equipment												
1.Principal												
2.Depreciation Exp.												
3.Net Value												
Transportation Equipment												
4.Principal												
5.Depreciation Exp.												
6.Net Value												
7.Total Deprecation Exp.												

Table 4 The Format of Depreciation Schedule

In Table 3, principal (Row 1 and Row 4) is the original value of an asset; depreciation expense (Row 2 and Row 5) is calculated by using different depreciation methods (Table 2). Net value (Row 3 and Row 6) is calculated by subtracting current year's depreciation expense (Row 2 and Row 5) from last year's net value. Total depreciation expense (Row 7) is calculated by summing up depreciation expenses, Row 2 and Row 5.

Goals	Strategies
Minimize tax payment	Plan A: Use shorter depreciation periods, and declining-balance method.
Minimize tax payment & enhance financial statements	Plan B: Use average depreciation periods, and sum-of -the-years'-digits method.
Enhance financial statements	Plan C: Use longer depreciation periods, and straight-line method.

Table 5 Optimizing Strategies

The backward chaining expert system is established to optimize the investment strategy. This backward chaining expert system has two goals. One is to minimize tax payments in earlier years. The other is to provide enhanced financial statements (higher profits) to shareholders. According to user's request, the expert system can optimize one of the two goals, or both of them. Based on the selected goals, the expert system chooses the compatible plan listed in Table 5.

The knowledge base of the expert system consists of the following rules:

Rule 10: IF optimizing is not the goal THEN optimizing is not needed.

Rule 20: IF optimizing is the goal THEN try to optimize.

Rule 30: IF try to optimize considering tax only THEN use plan A.

Rule 40: IF try to optimize considering both tax and financial reporting THEN use plan B.

Rule 50: IF try to optimize considering neither tax nor financial reporting THEN no optimizing action is taken.

Rule 60: IF try to optimize considering financial reporting only THEN use plan C.

After starting the experimental system, it first reads data from the data files. For those fixed assets, the system obtains original values from the data file. The forward chaining expert system asks user questions about the characteristics of the assets. Based on user's answers, the expert system chooses proper depreciation methods for those assets. After the mathematical model gets all the data, it calculates the cash flow, NPV and IRR. At the last step, the backward chaining expert system asks user to provide the optimizing goal of the investment, then it provides optimizing strategies to the user. The source code of this experimental system is in APPENDIX A. One actual result is in APPENDIX B. The selected results of running the experimental system and the analyses of the experiment are as follows. The characters in italic represent the prompts on the screen. The characters in bold represent user's inputs.

The Experimental System For

Financial Feasibility Study

...... reading data from the data file.

Forward Chaining Expert System for Selecting Depreciation Method

Electronic or Computer Equipment

Usage can be divided into units (UNI).

Life term can be measured by year or month (TIM).

Productivity decreases by time (DEC).

Do not know (DEF).

-->TIM

The expert system first prints the name of the asset and the categories of possible characteristics on the screen. It then waits for user's answer. In this forward chaining expert system, there are three categories of assets. If user does not know which category the asset belongs to, user can enter DEF. The expert system defaults the asset to category TIM.

How long can the equipment be used? **5** (years)

After the user selects the category, the expert system prints the question to ask the user how long that asset can be used. Then it waits for the user's answer.

Transportation Equipment:

Usage can be divided into units (UNI).

Life term can be measured by year or month (TIM).

Productivity decreases by time (DEC).

Do not know (DEF).

-->DEC

Sum of the years' digits (SYD).

Declining balance (DBM).

-->SYD

How long can the equipment be used? **5** (years)

After all the questions have been answered, the expert system constructs the depreciation schedule.

DEPRECIATION SCHEDULE

Computer or Electronic Equipment

Iterm\Year		*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
PRIN	*149.00*								
DEPR	*26.82*	*26.82*	*26.82*	*26.82*	*26.82*				
NVAL		*92.38*	*65.56*	*11.92*	*14.90*	*14.90*	*14.90*	*14.90*	

Item\Year	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
PRIN								
DEPR								
NVAL	*14.90*	*14.90*	*14.90*	*14.90*	*14.90*	*14.90*	*14.90*	*14.90*

Transportation Equipment

Iterm\Year	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
PRIN	*20.00*							
DEPR	*6.00*	*4.80*	*3.60*	*2.40*	*1.20*			
NVAL	*14.00*	*9.20*	*5.60*	*3.20*	*2.00*	*2.00*	*2.00*	

Iterm\Year	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
PRIN								
DEPR								
NVAL	*2.00*	*2.00*	*2.00*	*2.00*	*2.00*	*2.00*	*2.00*	*2.00*

Total of Depreciation

Item\Year	*0*	*1*	*2*	*3*	*4*	*5*	...	*15*
TOTA	*32.82*	*31.62*	*30.42*	*29.22*	*28.02*	*0.00*	...	*0.00*

PRIN -- Principal
DEPR -- Depreciation Expense
NVAL -- Net Value
TOTA -- Total Depreciation Expense

......processing data in the mathematical model.

After receiving all the needed data, the mathematical model constructs the projected cash flow statement, and calculates NPV and IRR.

CASH FLOW STATEMENT

Iterm\Year	*0*	*1*	*2*	*3*	*4*
NSAL		*173.15*	*426.10*	*717.60*	*1034.60*
DEPR		*32.82*	*31.62*	*30.42*	*29.22*
RESI					
KNOW		*5.40*	*5.40*	*5.40*	*5.40*
CAIN	*0.00*	*211.37*	*463.12*	*753.42*	*1069.22*
INVE	*280.00*				
TAX					*8.67*
VCOS		*102.84*	*216.68*	*344.04*	*513.71*
FCOS		*149.80*	*205.04*	*269.33*	*416.39*
COUT	*280.00*	*252.64*	*421.72*	*640.37*	*938.78*
NECF	*-280.00*	*-41.27*	*41.40*	*113.05*	*130.44*
DRAT 12%	*1.00*	*0.89*	*0.80*	*0.71*	*0.64*
PV	*-280.00*	*-36.85*	*33.00*	*80.47*	*82.90*

NSAL -- Net Sales
RESI -- Residual Value
CAIN -- Cash Inflow
INVE -- Investment
VCOS -- Variable Cost
FCOS -- Fixed Cost
COUT -- Cash Outflow
KNOW -- Know How
NECF -- Net Cash Flow
DRAT -- Discount Rate

Item\Year	*5*	*6*	*7*	*8*	*9*
NSAL	*1417.00*	*1417.00*	*1417.00*	*1417.00*	*1417.00*
DEPR	*28.02*				
RESI					

KNOW	5.40				
CAIN	1450.42	1417.00	1417.00	1417.00	1417.00
INVE	280.00				
TAX	14.45	17.23	17.23	17.23	17.23
VCOS	707.84	707.84	707.84	707.84	707.84
FCOS	535.05	501.63	501.63	501.63	501.63
COUT	1257.34	1226.69	1226.69	1226.69	1226.69
NECF	193.08	190.31	190.31	190.31	190.31
DRAT 12%	0.57	0.51	0.45	0.40	0.36
PV	109.56	96.42	86.09	76.86	68.63

Item\Year	10	11	12	13	14
NSAL	1417.00	1417.00	1417.00	1417.00	1417.00
DEPR					
RESI					
KNOW					
CAIN	1417.00	1417.00	1417.00	1417.00	1417.00
INVE					
TAX	34.24	34.24	34.24	34.24	34.24
VCOS	707.84	707.84	707.84	707.84	707.84
FCOS	501.63	501.63	501.63	501.63	501.63
COUT	1243.71	1243.71	1243.71	1243.71	1243.71
NECF	173.39	173.29	173.29	173.29	173.29
DRAT 12%	0.32	0.29	0.26	0.23	0.20
PV	55.79	49.28	44.48	39.71	35.46

Item\Year	15
NSAL	1417.00
DEPR	
RESI	16.90
KNOW	
CAIN	1433.90
INVE	
TAX	34.24

VCOS	*707.84*
FCOS	*501.63*
COUT	*1243.71*
NECF	*190.19*
DRAT 12%	*0.18*
PV	*34.57*

IRR=31.15%

NPV=577.08

After calculating NPV and IRR, the backward chaining expert system asks the user whether optimizing is the goal. If the user answers yes, then the expert system will ask user what is the optimizing goal. According to user's answer the expert system gives user an optimizing strategy (Table 4).

The Backward Chaining Expert System for Optimizing

*Enter YES or NO to indicate whether optimizing is the goal-->***YES**

*Enter YES or NO to indicate optimizing tax only-->***NO**

*Enter YES or No to indicate optimizing financial reporting only-->***NO**

*Enter YES or NO to indicate optimizing both goals-->***YES**

Suggestion: Use Plan B.

By using the suggested optimizing strategy and rerunning the mathematical model, the experimental system reconstructs the projected cash flow statement, and recalculates NPV and IRR.

CASH FLOW STATEMENT

Item\Year	*0*	*1*	*2*	*3*	*4*
NSAL		*173.15*	*426.10*	*717.60*	*1034.60*
DEPR		*28.88*	*25.80*	*22.72*	*19.64*
RESI					
KNOW		*5.40*	*5.40*	*5.40*	*5.40*
CAIN	*0.00*	*207.43*	*457.30*	*745.72*	*1059.64*
INVE					
TAX					*9.47*

VCOS		102.84	216.68	344.04	513.71
FCOS		145.86	199.22	288.63	406.81
COUT	280.00	248.70	415.90	623.67	929.99
NECF	-280.00	-41.27	41.40	113.05	129.65
DRAT 12%	1.00	0.89	0.80	0.71	0.64
PV	-280.00	-36.85	33.00	80.47	82.39

Item\Year	5	6	7	8	9
NSAL	1417.00	1417.00	1417.00	1417.00	1417.00
DEPR	16.56	13.48	10.40	7.31	4.88
RESI					
KNOW	5.40				
CAIN	1438.96	1430.48	1427.40	1424.31	1421.88
INVE					
TAX	15.40	16.11	16.36	16.62	16.82
VCOS	707.84	707.84	707.84	707.84	707.84
FCOS	523.69	515.11	512.03	508.94	506.51
COUT	1246.83	1239.05	1236.23	1233.40	1231.16
NECF	192.13	191.43	191.17	190.91	190.71
DRAT 12%	0.57	0.51	0.45	0.40	0.36
PV	109.02	96.98	86.48	77.11	68.77

Item\Year	10	11	12	13	14
NSAL	1417.00	1417.00	1417.00	1417.00	1417.00
DEPR	2.44				
RESI					
KNOW					
CAIN	1419.44	1417.00	1417.00	1417.00	1417.00
INVE					
TAX	33.84	34.24	34.24	34.24	34.24
VCOS	707.84	707.84	707.84 7	07.84	707.84
FCOS	504.07	501.63	501.63	501.63	501.63
COUT	1245.75	1243.71	1243.71	1243.71	1243.71
NECF	173.69	173.29	173.29	73.29	173.29

DRAT 12%	0.32	0.29	0.26	0.23	0.20
PV	55.92	49.28	44.48	39.71	35.46

Item\Year	15
NSAL	1417.00
DEPR	
RESI	16.90
KNOW	
CAIN	1433.90
INVE	
TAX	34.24
VCOS	707.84
FCOS	501.63
COUT	1243.71
NECF	190.19
DRAT 12%	0.18
PV	34.57

IRR=31.15%

NPV=577.51

CHAPTER V

SUMMARY, CONCLUSIONS, AND FUTURE WORK

This research attempted to assess the possibility of using a hybrid approach (combining a mathematical model with expert systems) to solve complex investment decision problems.

The results of running the experimental system indicated that it is feasible to combine a mathematical model with different kinds of expert systems in solving complex investment decision problems. Both the forward chaining expert system and the backward chaining expert system worked well with the mathematical model. The algorithm used for the depreciation forward chaining expert system could also be used for a taxation expert system or a sales and marketing expert system. The algorithm for the backward chaining expert system design was suitable for optimizing investment strategies.

The mathematical model design presented in this thesis covered most aspects of joint venture investment decision making. In practice, it is unnecessary to use all the methods introduced in Chapter III to make an investment decision. Some of the methods, such as the methods in ratio analysis, are optional. The algorithms used for the forward chaining expert system and the backward chaining expert system were easy to implement and to combine with the mathematical model. They are suitable for relatively small expert systems.

This research addressed the designs of the mathematical model, forward chaining expert system, and backward chaining expert system. Because of the hybrid approach of combing a mathematical model with different expert systems, further research work must be pursued in the areas of interpretations and recommendations. In addition, due to the existence of many uncertain factors that are hard to project, such as foreign exchange rate, interest rate, and inflation, future research work also needs to be done in developing new models and methods to deal with uncertainty.

REFERENCES

1. Anonymous. "Expert System Helps Audit Commercial Loans." WG & L Accounting News Vol. 9(1). 1988 Spring: 20-21.

2. Biggs, S. F. "Financial Analysis' Information Search in the Assessment of Corporate Earing Power." Accounting, Organizations and Society Vol. 9 (314): 313-323; 1984.

3. Bouwman, M. J. Towards Expert Systems: the Analysis of Expert Financial Behavior. Technical Report. Engene, OR: University of Oregon; 1985.

4. Brigham, E. F. and Gapenski, L. C. Intermediate Financial Management. Chicago, IL: The Dryden Press; 1990.

5. Buchanan, B. G. and Shortliffe, E. H. Rule-Based Expert System: The MYCIN Experiments of the Stanford Heuristic Programming Project. Reading, MA: Addison-Wesley Publishing Company; 1984.

6. Chase, M. D. and Shim, J. K. "Artificial Intelligence and Big Six Accounting -- A Survey of the Current Use of Expert System in Modern Accounting Environment." Computer and Industrial Engineering [CIE] Vol. 21: 205 - 209; 1991.

7. Chui, C. Y. Artificial Intelligence and Its Applications to Capital Budgeting Decisions Under Uncertainty. Auburn University. PHD dissertation 1992.

8. Clocksin, W. and Mellish, C. Programming in PROLOG. New York, NY: Springer-Verlag; 1987.

9. Console, L.; Portinale, L.; Theseider Dupre, D.; and Torasso, P. "Combining Heuristic Reasoning with Causal Reasoning in Diagnostic Problem Solving." In: David, J. M; Krivine, J. P; Simmons, R. Ed. Second Generation Expert System. New York, NY: Springer-Verlag; 1993; 46-68.

10. Ernet, G. W. and Newell, A. GPS: A Case Study in Generality and Problem-Solving. New York, NY: Academic Press; 1969.

11. Foltin, L. C. and Smith, L. M. "Accounting Expert System." CPA Journal Vol. 64(11): 46-53; 1994.

12. Forgy, C. L. The OPS5 User Manual. Technical Report CMU-CS_79_132 Computer Science Department. Carnegie-Mellon University; 1981.

13. Hamscher, W. The Business Analyzer: "A Second Generation Approach to Financial Decision Support." In: David, J. M; Krivine, J. P; Simmons, R. Ed. Second Generation Expert System. New York, NY: Springer-Verlag; 1993; 93-109.

14. Han, M. H. A Knowledge-Based Decision Support System (KBDSS) For Corporate Portfolio Analysis: An Integrative Model Approach (Portfolio Analysis). Georgia State University. PHD dissertation 1992.

15. Harman, P. and Sawyer, B. Creating Expert System for Business and Industry. New York, NY: John Wiley & Sons, Inc.; 1990.

16. Helfert, E. A. Techniques of Financial Analysis. Momewood, IL: Business One Irwin; 1991.

17. Kieso, D. E. and Weygandt, J. J. Intermediate Accounting. New York, NY. John Wiley & Sons, Inc.; 1992.

18. Koton, P. "Combining Causal Models and Case-Based Reasoning." In: David, J. M; Krivine, J. P; Simmons, R. Ed. Second Generation Expert System. New York, NY: Springer-Verlag; 1993; 69-78.

19. Krushelnycky, E. Merging Rule-Based Expertise: A Financial Management Cost Accounting System Application. University of Toronto (Canada). PHD dissertation 1994.

20. Lenard, M. J. A Comparison of Statistical, Expert System and Hybrid Models Applied to the Auditor's Going Concern Assessment (Auditors). Kent State University. PHD dissertation 1995.

21. Levine, R. I. and Drang; Edelson B. AI and expert system. New York, NY: Mcgraw-Hill, Inc.; 1990.

22. Lindsay, R. K.; Buchanan, B.; Feigenbaum, E. A. and Lederberg, J. Application of Artificial Intelligence to Chemistry: The DENDRAL Project. New York, NY: McGraw-Hall; 1980.

23. Loofbourrow, T. "Expert System Are Still Alive." Informationweek. 1995 July 17: 104.

24. McCarthy J. "History of LISP." In: Barr, A. and Feigenbaum, E. A. The Handbook of Artificial Intelligence Vol. 2. Reading, MA: Addison-Wesley Publishing Company; 1981.

25. McDermott, J. R1: "A Rule-Based Configurer of Computer Systems." Artificial Intelligence Vol. 1(1); 1979

26. McDermott, J. "R1's Formative Years." AI Magazine Vol. 2(2); 1982.

27. McEacharn, E. M. A Fuzzy Reasoning Expert System For Planning-Stage Materiality Judgments. Louisiana Tech University. DBA dissertation 1994.

28. Medsker, L. and Liebouitz, J. Design and Development of Expert System and Neural Networks. New York, NY: Macmillan Publishing Company; 1994.

29. Michaelson, R. H. "Development of an Expert System to Assist in the Classification of Estate Tax Returns." Accounting Horizons Vol. 2(4). 1988 December: 63-70.

30. Mui, C. and McCarthy, W. E. "FSA: Applying AI Techniques to the Familization Phase of Financial Decision Making." IEEE Expert. 1989 February: 38-48.

31. Mui, C. K.; Hassel, C. F.; and Curtis.L. C. "The Financial Statement Analyzer" In Liebowitz, J. Ed. Expert Systems for Business and Management. Englewood Cliffs, NJ. Yourdon Press; 1990: 127-142.

32. Newell, A. and Simon, H. Human Problem Solving. Englewood Cliffs, NJ: Prentice-Hall; 1972.

33. Paillet, R. "Multiple Models for Emergency Planning." In: David, J. M; Krivine, J. P; Simmons, R. Ed. Second Generation Expert System. New York, NY: Springer-Verlag; 1993; 161-173.

34. Ravi, T. A Knowledge-Based Approach to the Design, Simulation, and Evaluation of Flexible Manufacturing Systems. McMaster University (Canada). PHD dissertation 1994.

35. Savory, S. E. "Tools for Building Expert System." In: Savory, S. E. Ed. Expert System in the Organization: an Introduction for Decision-Makers. Chichester, West Sussex, England: Ellis Horwood Limited; 1988.

36. Sena, J. A. and Smith, L. M. "A Sample Expert System for Financial Statement Analysis." Journal of Accounting and EDP (JAD) Vol. 3(2). 1987 Summer: 15-22.

37. Shortliffe, E. H. Computer-Based Medical Consultations: MYCIN. New York, NY. American Elsevier; 1976.

38. Simmons, R. "Generate, Test and Debud: A Paradigm for Combining Associational and Causal Reasoning." In: David, J. M; Krivine, J. P; Simmons, R. Ed. Second Generation Expert System. New York, NY: Springer-Verlag; 1993; 79-92.

39. Simon, H. A. The New Science of Management Decision. New York, NY: Harper Brother, 1960.

40. Simon, H. and Ericsson, K. "Verbal Report as Data." Psychological Review Vol. 87 (3): 215-251; 1980.

41. Sindi, A. A. Acceptance of Information Technology User Acceptance Model for Expert Systems. The University of Michigan. PHD dissertation 1992.

42. Steinbart, P. J. "The Construction of a Rule-Based Expert System as a Method for Studying Materiality Judgments." Accounting Review Vol. 62(10). 1987

43. Swinney, L. S. Expert System in Auditing: Decision Aid or Decision Subjugation. The University of Nebraska-Lincoln. PHD dissertation 1993.

44. Turban, E. Decision Support and Expert Systems: Management Support System. New York, NY: Macmillan Publishing Company; 1993.

45. Vasahely, M. A. and Lin, T. W. Advanced Auditing. Reading, MA; Addison-Wesley Publishing Company; 1988.

46. Walsh, C. Key Management Ratios. London, UK: Pitman Publishing; 1993.

APPENDIX A

THE SOURCE CODE OF THE EXPERIMENTAL SYSTEM

```
#include <stdio.h>
#include <string.h>



/* Functions declaration */
     void DATA_COLLECT (float DATA[300][17]);
     void DEPR_EXP (int pp, float DEPR[11][17]);
          void search (char clvarlt[40][3], char cndvar[10][3], int fp,
                    int *sn, int f);
          void check (char varlt[10][3], char equi[10], char choi[10],
                    char v[3], int instlt[10]);
     void MATH (float DATA[300][17]);
     void BACK_EXP (float DATA[300][17]);
          void bcheck (int *bsn, int bf, char bvarble[3],
                    char bconclt[10][3]);
          void bstack (int *bsp, int bsn, int bstatsk[11],
                    int bclausk[11]);
          void binstantiate (char bvarble[3], char bvarlt[10][3],
                    int binstlt[11], char goal[4], char taxx[4],
                    char both[4], char fina[4]);
     void REPORT (float DATA[300][17]);

main ()
{
     float worksheet[300][17];
     int i, j;

     printf("The Experimental System for Financial Feasibility\n");
     printf("\n\n\n");

     for (i=0; i<=16; i++)
          for (j=0; j<300; j++)
               worksheet[j][i]=0.0;
     DATA_COLLECT(worksheet);
     printf("DATA completed\n");
     MATH (worksheet);
```

```
    printf("CALCULATION completed\n");
    REPORT(worksheet);
    BACK_EXP(worksheet);
}

void DATA_COLLECT (float DATA[300][17])
{   int i, j;
    float COST[24][17], DEPR[11][17], WAGE[24][17];
    float INCO[10][17], REVE[17][17], CASH[14][17];
    FILE *mfp;

    printf("        ...... reading data from data file.\n");
    printf("\n\n\n");

    /* initiallization */
    for (j=0; j<=16; j++)
    {    for (i=0; i<=23; i++)
              COST[i][j] = 0.0;
         for (i=0; i<10; i++)
              DEPR[i][j] = 0.0;
         for (i=0; i<=23; i++)
              WAGE[i][j] = 0.0;
         for (i=0; i<=9; i++)
              INCO[i][j] = 0.0;
         for (i=0; i<=16; i++)
              REVE[i][j] = 0.0;
         for (i=0; i<=13; i++)
              CASH[i][j] = 0.0;
    }

    /* read data from data file */

    /* cost file */
    if ((mfp = fopen("cost.dat", "r")) == NULL)
    {    printf("Can not open cost file!\n");
         exit(1);
    }
    for (i=1; i<=23; i++)
         for (j=1; j<=15; j++)
              fscanf(mfp, "%f", &COST[i][j]);
    fclose(mfp);

    /* wage file */
    if ((mfp = fopen("wage.dat", "r")) == NULL)
    {    printf("Can not open wage file!\n");
```

```
      exit(1);
}
for (i=1; i<=23; i++)
      for (j=1; j<=15; j++)
            fscanf(mfp, "%f", &WAGE[i][j]);
fclose(mfp);

/* equipment file */

printf("Forward Chaining Expert System for Selecting ");
printf(" Depreciation Method\n");

if ((mfp = fopen("equi.dat", "r")) == NULL)
{     printf("Can not open equipment file!\n");
      exit(1);
}
for (i=1; i<=9; i=i+3)
{     fscanf(mfp, "%f", &DEPR[i][1]);
      DEPR_EXP (i, DEPR);
}
fclose(mfp);

/* income file */
if ((mfp = fopen("inco.dat", "r")) == NULL)
{     printf("Can not open income file!\n");
      exit(1);
}
for (i=1; i<=9; i++)
      for (j=1; j<=15; j++)
            fscanf(mfp, "%f", &INCO[i][j]);
fclose(mfp);

/* revenue file */
if ((mfp = fopen("reve.dat", "r")) == NULL)
{     printf("Can not open revenue file!\n");
      exit(1);
}
for (i=1; i<=15; i++)
      for (j=1; j<=15; j++)
            fscanf(mfp, "%f", &REVE[i][j]);
fclose(mfp);

/* cash flow file */
if ((mfp = fopen("cash.dat", "r")) == NULL)
{     printf("Can not open cash flow file!\n");
```

```
            exit(1);
        }
        for (i=1; i<=6; i++)
            for (j=0; j<=15; j++)
                fscanf(mfp, "%f", &CASH[i][j]);
        fclose(mfp);

        /* combine all data into a worksheet file */
        for (j=0; j<=16; j++)
        {    for (i=1; i<=23; i++)
                DATA[i][j]=WAGE[i][j];

            for (i=41; i<=50; i++)
                DATA[i][j]=DEPR[i-40][j];

            for (i=81; i<=103; i++)
                DATA[i][j]=COST[i-80][j];

            for (i=121; i<=129; i++)
                DATA[i][j]=INCO[i-120][j];

            for (i=161; i<=176; i++)
                DATA[i][j]=REVE[i-160][j];

            for (i=201; i<=213; i++)
                DATA[i][j]=CASH[i-200][j];
        }

}

void DEPR_EXP (int pp, float DEPR[11][17])
{       float u_depr, sum_year, remainder, percent, salvage;
        float u_avg, u_total, base;
        char varlt[11][3], clvarlt[41][3], cndvar[11][3];
        char equi[10], choi[10], c[3], vp[3], v[3];
        int f, i, j, k, s, fp, bp, gr, sn, cn, entry, instlt[11];
        int i1, t1, t2, end, year;


        /* initialization */
        fp = 1;
        bp = 1;
        for (i=0; i<=40; i++)
            strcpy(clvarlt[i], "");
        for (i=0; i<=10; i++)
```

```
{     strcpy(cndvar[i], "");
      strcpy(varlt[i], "");
      instlt[i]=0;
}

/* construct variable list */
strcpy(varlt[1], "EQU");
strcpy(varlt[2], "CHO");

/* construct variable clause list */
strcpy (clvarlt[1], "EQU");
strcpy (clvarlt[5], "EQU");
strcpy (clvarlt[9], "EQU");
strcpy (clvarlt[13], "EQU");
strcpy (clvarlt[14], "CHO");
strcpy (clvarlt[17], "EQU");
strcpy (clvarlt[18], "CHO");

/* catagory */
if (pp == 1)
      printf("Electronic or Computer Equipment.\n");
else if (pp == 4)
      printf("Transportation Equipment.\n");
else
      printf("Building.\n");

/* inference engine */
strcpy(cndvar[bp], "EQU");
bp = bp+1;
sn = 1; cn = 1;
f = 1;
entry=0;
while (entry==0)
{     entry=1;
      search (clvarlt, cndvar, fp, &sn, f);
      /*printf("-> %i\n", sn);*/
      cn = 1;
      if (sn != 0)
      {     i = 4*(sn-1)+cn;
            strcpy(v, clvarlt[i]);
            while (strncmp(v, "", 3)!=0)
            {
                  check(varlt, equi, choi,
                        v, instlt);
                  cn = cn + 1;
```

```
        i = 4*(sn-1)+cn;
        strcpy(v, clvarlt[i]);
    }
    s = 0;
    /* if part of rules */
    switch(sn)
    {       case 1: if (strcmp(equi, "UNI") == 0) s=1;
                    break;
            case 2: if (strcmp(equi, "DEF") == 0) s=1;
                    break;
            case 3: if (strcmp(equi, "TIM") == 0) s=1;
                    break;
            case 4: if ((strcmp(equi, "DEC") == 0) &&
                    (strcmp(choi, "SYD") == 0)) s=1;
                    break;
            case 5: if ((strcmp(equi, "DEC") == 0) &&
                    (strcmp(choi, "DBM") == 0)) s=1;
                    break;
    }
    /* then part of rules */
    if (s == 1)
    {/* invoke then part of relus */
    switch(sn)
    {case 1: /* Activity Method */
        printf("*** Activity Method ***\n");
        printf("Average Units Cost A Year:->");
        scanf("%f", &u_avg);
        printf("Total Usable Units:->");
        scanf("%f", &u_total);
        salvage = DEPR[pp][1]*0.1;
        base = DEPR[pp][1]-salvage;
        u_depr = (base*u_avg)/u_total;
        for (i1=1; i1<=15; i1++)
        {       base = base-u_depr;
                if (base>0)
                {       DEPR[pp+1][i1]=u_depr;
                        DEPR[pp+2][i1]=base+salvage;
                }
                else
                {       base=base+u_depr;
                        DEPR[pp+1][i1]=base;
                        base = 0;
                        DEPR[pp+2][i1]=salvage;
                }
        }
```

```
        break;
    case 2: /* Default Method */

    case 3: /*Straight-Line Method */
        printf("*** Straight-Line Method ***\n");
        printf("How Long Can the equipment be
Used:->");

        scanf("%i", &year);
        salvage = DEPR[pp][1]*0.1;
        base = DEPR[pp][1]-salvage;
        u_depr = base/year;
        for (i1=1; i1<=16; i1++)
        {     base = base-u_depr;
              if (base>0)
              {     DEPR[pp+1][i1]=u_depr;
                    DEPR[pp+2][i1]=base+salvage;
              }
              else
              {     base = base+u_depr;
                    DEPR[pp+1][i1]=base;
                    base = 0;
                    DEPR[pp+2][i1] = salvage;
              }

        }
        break;

    case 4: /* Sum-of-the-Years'-Digits */
        printf("*** Sum-of-the-Years'-Digits ***\n");
        printf("How Long Can the equipment be
Used:->");

        scanf("%i", &year);
        sum_year = 0;
        for (i1=1; i1<=year; i1++)
              sum_year = sum_year+i1;
        salvage = DEPR[pp][1]*0.1;
        base = DEPR[pp][1]-salvage;
        remainder = DEPR[pp][1];
        if (year<15) end=1;
        else end=year-15+1;
        for (i1=year; i1>=end; i1--)
        {
              DEPR[pp+1][year-i1+1]=base*
                          i1/sum_year;
              remainder = remainder-
```

```
                    DEPR[pp+1][year-i1+1];
                DEPR[pp+2][year-i1+1]=remainder;
            }
            if (year<15)
                for (i1=year+1; i1<=15; i1++)
                    DEPR[pp+2][i1]=remainder;
            break;

        case 5:  /* Declining-Balance Method */
            printf("*** Declining_Balance Method ***\n");
            printf("The life term of the euqipment:->");
            scanf("%i", &year);
            percent=2/year;
            remainder=DEPR[pp][1];
            if (year<=15) end=year;
            else end=15;
            for (i1=1; i1<=end; i1++)
            {    DEPR[pp+1][i1]=remainder*percent;
                remainder=remainder-DEPR[pp+1][i1];
                DEPR[pp+2][i1]=remainder;
            }
            if (year<15)
                for (i1=year+1; i1<=15; i1++)
                    DEPR[pp+2][i1]=remainder;
            break;

        }
        }
        else
        {    f = sn+1;
            entry=0;
        }
    }
    else
    {    fp = fp+1;
        if (fp<bp)
        {    f = 1;
            entry=0;
        }
    }
  }
}

void search (char clvarlt[41][3], char cndvar[11][3], int fp, int *sn, int f)
{    int flag, cn, k;
```

```
    flag=0; cn=0; k=0;
    *sn=f;
    while ((flag==0) && (*sn<=10))
    {     cn=cn+1;
          k=(*sn-1)*4+cn;
          while((strncmp(clvarlt[k], cndvar[fp], 3)!=0)&&(cn<4))
          {     cn=cn+1;
                k=(*sn-1)*4+cn;
          }
          if(strncmp(clvarlt[k], cndvar[fp], 3)==0) flag=1;
          if(flag==0) *sn=*sn+1;
    }
    if (flag==0) *sn=0;
}

void check (char varlt[10][3], char equi[10], char choi[10],
          char v[3], int instlt[10])
{     int i;
    i=1;
    while((strncmp(v, varlt[i], 3)!=0)&&(i<=10)) i=i+1;
    if(instlt[i]!=1)
    {     instlt[i]=1;
          switch(i)
          {     case 1:
                printf("Usage can be devided into units UNI\n");
                printf("Life can be measured by year or month TIM\n");
                printf("Productity decreased by time DEC\n");
                printf("Do not know DEF\n");
                scanf("%s", equi);
                break;

                case 2:
                printf("Sum-of-the-years'-digits SYD\n");
                printf("Declining-balance-method DBM\n");
                scanf("%s", choi);
                break;
          }
    }
}


void MATH(float DATA[300][17])
{     int i, j;
    float npv, IRR, k1, k2, diff;
```

```
float m1, m2, m3;      /* mid-variables */

/* calculate wage */
for (j=1; j<=15; j++)
{    /* management 1 */
     DATA[3][j]=DATA[1][j]*DATA[2][j]*12.0;

     /* engineer 1.1 */
     DATA[6][j]=DATA[4][j]*DATA[5][j]*12.0;

     /* engineer 1.2 */
     DATA[12][j]=DATA[10][j]*DATA[11][j]*12.0;

     /* others 1 */
     DATA[9][j]=DATA[7][j]*DATA[8][j]*12.0;

     /* subtotal 1 */
     DATA[13][j]=DATA[3][j]+DATA[6][j]+DATA[9][j]+DATA[12][j];

     /* management 2 */
     DATA[16][j]=DATA[14][j]*DATA[15][j]*12.0;

     /* engineer 2 */
     DATA[19][j]=DATA[17][j]*DATA[18][j]*12.0;

     /* subtotal 2 */
     DATA[20][j]=DATA[16][j]+DATA[19][j];

     /* total */
     DATA[21][j]=DATA[13][j]+DATA[20][j];

     /* management total */
     DATA[22][j]=DATA[3][j]+DATA[16][j];

     /* other total */
     DATA[23][j]=DATA[21][j]-DATA[22][j];
}

/* calculate depreciation */
for (j=1; j<=15; j++)
     DATA[50][j]=DATA[42][j]+DATA[45][j]+DATA[48][j];

/* calculate cost */
for (j=1; j<=15; j++)
{    /* variable cost */
```

```
    DATA[81][j]=DATA[23][j];
    DATA[85][j]=DATA[81][j]+DATA[82][j]+DATA[83][j]+
          DATA[84][j];

    /* fixed cost */
    DATA[102][j]=0.0;
    DATA[86][j]=DATA[22][j];
    DATA[89][j]=DATA[50][j];
    if (j<=5) DATA[101][j]=5.4;
    for (i=86; i<=101; i++)
         DATA[102][j]=DATA[102][j]+DATA[i][j];

    /* total cost */
    DATA[103][j]=DATA[85][j]+DATA[102][j];
}


/* calculate income */
for (j=1; j<=15; j++)
{   /* domestic income */
    DATA[125][j]=DATA[121][j]+DATA[122][j]+DATA[123][j]+
          DATA[124][j];

    /* abroad income */
    DATA[128][j]=DATA[126][j]+DATA[127][j];

    /* total income */
    DATA[129][j]=DATA[125][j]+DATA[128][j];
}

/* calculate revenue */
for (j=1; j<=15; j++)
{   /* total sales */
    DATA[161][j]=DATA[128][j];
    DATA[162][j]=DATA[125][j];
    DATA[163][j]=DATA[161][j]+DATA[162][j];

    /* net sales */
    DATA[166][j]=DATA[163][j]-DATA[164][j]-DATA[165][j];

    /* profit before tax */
    DATA[167][j]=DATA[85][j];
    DATA[168][j]=DATA[102][j];
    DATA[169][j]=DATA[166][j]-DATA[167][j]-DATA[168][j];
```

```
    /* tax */
    DATA[171][j]=DATA[170][j]*DATA[169][j];
    /* revenue after tax */
    DATA[172][j]=DATA[169][j]-DATA[171][j];
}

/* calculate cash flow */
DATA[203][15]=DATA[43][15]+DATA[46][15]+DATA[49][15];
DATA[206][0]=280.00;
for (j=0; j<=15; j++)
{   /* in */
    DATA[201][j]=DATA[166][j];
    DATA[202][j]=DATA[50][j];
    DATA[205][j]=DATA[201][j]+DATA[202][j]+DATA[203][j]+
          DATA[204][j];

    /* out */
    DATA[207][j]=DATA[171][j];
    DATA[208][j]=DATA[85][j];
    DATA[209][j]=DATA[102][j];
    DATA[210][j]=DATA[206][j]+DATA[207][j]+DATA[208][j]+
          DATA[209][j];

    /* net */
    DATA[211][j]=DATA[205][j]-DATA[210][j];
}

/* total */
for (i=1; i<=300; i++)
{   DATA[i][16]=0;
    for (j=0; j<=15; j++)
        DATA[i][16]=DATA[i][16]+DATA[i][j];
}

/* NPV */
DATA[212][0]=1;
m1=1/(1+0.12);
for (j=1; j<=15; j++)
    DATA[212][j]=DATA[212][j-1]*m1;
DATA[213][16]=0;
for (j=0; j<=15; j++)
{   DATA[213][j]=DATA[211][j]*DATA[212][j];
    DATA[213][16]=DATA[213][16]+DATA[213][j];
}
```

```
    /* IRR */
    if (DATA[211][16]>0)
    {   k1=0.0; k2=1.0;
        diff=1;
        while (diff>0.001)
        {   IRR=(k1+k2)/2;
            DATA[214][0]=1;
            for (j=1; j<=15; j++)
                DATA[214][j]=DATA[214][j-1]*(1/(1+IRR));
            DATA[215][16]=0;
            for (j=0; j<=15; j++)
            {   DATA[215][j]=DATA[211][j]*DATA[214][j];
                DATA[215][16]=DATA[215][16]+DATA[215][j];
            }
            if (DATA[215][16]<0)
                k2=IRR;
            else
                k1=IRR;
            diff=k2-k1;
        }
        DATA[216][1]=IRR;      /* IRR value */
    }
}


void BACK_EXP(float DATA[300][17])
{   char bconclt[11][3], bvarlt[11][3], bclvarlt[41][3];
    char bvarble[3];
    char goal[4], poss[4], taxx[4], chan[4], fina[4], both[4];
    int binstlt[11], bstatsk[11], bclausk[11];
    int bsp, bsn, bf, bi, bj, bs, bk;

    /* initialization */
    bsp=11;
    for(bi=1; bi<=10; bi++)
    {   strcpy(bconclt[bi], "");
        strcpy(bvarlt[bi], "");
        binstlt[bi]=0.0;
        bstatsk[bi]=0.0;
        bclausk[bi]=0.0;
    }
    for (bi=1; bi<=40; bi++)
        strcpy(bclvarlt[bi], "");

    /* conclusion list */
```

```
strcpy(bconclt[1], "CH");
strcpy(bconclt[2], "PO");
strcpy(bconclt[3], "CH");
strcpy(bconclt[4], "CH");
strcpy(bconclt[5], "CH");
strcpy(bconclt[6], "CH");

/* variable list */
strcpy(bvarlt[1], "GO");
strcpy(bvarlt[2], "TA");
strcpy(bvarlt[3], "FI");
strcpy(bvarlt[4], "BO");

/* clause variable list */
strcpy(bclvarlt[1], "GO");
strcpy(bclvarlt[5], "GO");
strcpy(bclvarlt[9], "GO");
strcpy(bclvarlt[10], "TA");
strcpy(bclvarlt[13], "PO");
strcpy(bclvarlt[14], "FI");
strcpy(bclvarlt[15], "BO");
strcpy(bclvarlt[17], "PO");
strcpy(bclvarlt[18], "FI");
strcpy(bclvarlt[19], "BO");
strcpy(bclvarlt[21], "PO");
strcpy(bclvarlt[22], "FI");
/* varble */
strcpy(goal, "");
strcpy(taxx, "");
strcpy(both, "");
strcpy(fina, "");
strcpy(poss, "");
strcpy(chan, "");

/* inference engine */
printf("Enter Goal:->");
scanf("%s", bvarble);
loop: bf=1;
bcheck(&bsn, bf, bvarble, bconclt);
if (bsn!=0)
{     do
      {     bstack(&bsp, bsn, bstatsk, bclausk);
            do
            {     loop1: bi=((bstatsk[bsp]-1)*4+bclausk[bsp]);
                  /* clause variable */
```

```
        strcpy(bvarble, bclvarlt[bi]);
        if (strncmp(bvarble, "", 3)!=0)
        {/* is this clause variable a conclusion*/

            bf=1;
            bcheck(&bsn, bf, bvarble, bconclt);
            if (bsn!=0)
                goto loop;
            binstantiate(bvarble, bvarlt, binstlt,
                goal, taxx, both, fina);
            bclausk[bsp]=bclausk[bsp]+1;
        }
    }while(strncmp(bvarble, "", 3)!=0);
    /* no more clauses check IF part of statement*/
    bsn=bstatsk[bsp];
    bs=0;
    /* IF part */
    switch(bsn)
    {       /* IF part of rule 1 */
        case 1: if(strcmp(goal, "NO")==0) bs=1;
            break;
        /* If part of rule 2 */
        case 2: if(strcmp(goal, "YES")==0) bs=1;
            break;
        /* IF part of rule 3 */
        case 3: if((strcmp(goal, "YES")==0) &&
            (strcmp(taxx, "YES")==0)) bs=1;
            break;
        /* IF part of rule 4 */
        case 4: if((strcmp(poss, "YES")==0) &&
            (strcmp(fina, "NO")==0) &&
            (strcmp(both, "YES")==0)) bs=1;
            break;
        /* IF part of rule 5 */
        case 5: if((strcmp(poss, "YES")==0) &&
            (strcmp(fina, "NO")==0) &&
            (strcmp(both, "NO")==0)) bs=1;
            break;
        /* IF part of rule 6 */
        case 6: if((strcmp(poss, "YES")==0) &&
            (strcmp(fina, "YES")==0)) bs=1;
            break;
    }
    if (bs!=1)
    {       bi=bstatsk[bsp];
```

```
            strcpy(bvarble, bconclt[bi]);
            bf=bstatsk[bsp]+1;
            bcheck(&bsn, bf, bvarble, bconclt);
            bsp=bsp+1;
        }
}while((bs!=1) && (bsn!=0));

/* THEN part */
if(bsn!=0)
{       switch (bsn)
        {       /* THEN part of rule 1 */
            case 1: strcpy(chan, "NO");
                printf("No optimizing goal!\n");
                break;
            /* THEN part of rule 2 */
            case 2: strcpy(poss, "YES");
                printf("Possible optimizing!\n");
                break;
            /* THEN part of rule 3 */
            case 3: strcpy(chan, "YES");
                printf("Use Plan A optimizing!\n");
                break;
            /* THEN part of rule 4 */
            case 4: strcpy(chan, "YES");
                printf("Use Plan B optimizing!\n");
                break;
            /* THEN part of rule 5 */
            case 5: strcpy(chan, "NO");
                printf("No optimizing needed!\n");
                break;
            /* THEN part of rule 6 */
            case 6: strcpy(chan, "YES");
                printf("Use Plan C optimizing!\n");
                break;
        }
        /* pop the stack */
        bsp=bsp+1;
        if (bsp>=11)
            printf(" SUCCESS \n");
        else
        {       /* stack is not empty */
            bclausk[bsp]=bclausk[bsp]+1;
            goto loop1;
        }
}
```

```
    }
}

void bcheck(int *bsn, int bf, char bvarble[3], char bconclt[10][3])
{   int bi;
    *bsn=0;
    bi=bf;
    while((strncmp(bvarble, bconclt[bi],3)!=0) && (bi<7))
        bi=bi+1;
    if (strncmp(bvarble, bconclt[bi], 3)==0) *bsn=bi;
}

void bstack(int *bsp, int bsn, int bstatsk[11], int bclausk[11])
{   *bsp=*bsp-1;
    bstatsk[*bsp]=bsn;
    bclausk[*bsp]=1;
}

void binstantiate(char bvarble[3], char bvarlt[10][3], int binstlt[11],
        char goal[4], char taxx[4], char both[4], char fina[4])
{   int bi;
    bi=1;
    while((strncmp(bvarble, bvarlt[bi], 3)!=0) && (bi<5)) bi=bi+1;
    if((strncmp(bvarble, bvarlt[bi], 3)==0) && (binstlt[bi]!=1))
    {   binstlt[bi]=1;
        switch(bi)
        {   case 1: printf("Enter YES or NO to indicate
whether");
                printf(" there is a optimizing goal :->");
                scanf("%s", goal);
                break;
            case 2: printf("Enter YES or NO to indicate ");
                printf("optimizing tax goal only :->");
                scanf("%s", taxx);
                break;
            case 3: printf("Enter YES or NO to indicate
optimizing financial reporting goal only :->");
                scanf("%s", fina);
                break;
            case 4: printf("Enter YES or NO to indicate
optimizing both goals :->");
                scanf("%s", both);
                break;
        }
    }
```

```
}


void REPORT(float DATA[300][17])
{     int i, j;
      char d_item[7][5] = {"PRIN", "DEPR", "NVAL", "PRIN", "DEPR",
                    "NVAL", "TOTA"};
      char c_item[13][5] = {"NSAL", "DEPR", "RESI", "KNOW", "CAIN",
                    "INVE", "TAX", "VCOS", "FCOS", "COUT",
                    "NECF", "DRAT", "PV"};

      printf("\n\n");
      printf("I|Y  ---------- item|year\n");
      printf("PRIN ---------- principal\n");
      printf("DEPR ---------- depreciation  expense\n");
      printf("NVAL ---------- net value\n");
      printf("TOTA ---------- total depreciation expenses\n");
      printf("NSAL ---------- net sales\n");
      printf("RESI ---------- residule value\n");
      printf("CAIN ---------- cash inflow\n");
      printf("INVE ---------- investment\n");
      printf("VCOS ---------- variable cost\n");
      printf("FCOS ---------- fixed cost\n");
      printf("COUT ---------- cash outflow\n");
      printf("NECF ---------- net cash flow\n");
      printf("DRAT ---------- discount rate 12 precent\n");
      printf("PV   ---------- present value\n");
      printf("KNOW ---------- know how\n");
      printf("IRR  ---------- internal rate of return\n");
      printf("NPV  ---------- net present value\n");
      printf("\n\n");

      printf("THE DEPRECIATION SCHEDULE        [unit: 10,000 dollar]\n");

      printf("I|Y\t");
      for (i=0; i<=7; i++)
            printf("%d\t", i);
      printf("\n");

      printf("Computer or Electronic Equipment\n");

      for (i=41; i<=43; i++)
      {     printf("%s\t", d_item[i-41]);
            for (j=0; j<=7; j++)
```

```
            printf("%4.2f\t", DATA[i][j]);
        printf("\n");
    }

    printf("Transporation Equipment\n");

    for (i=44; i<=46; i++)
    {       printf("%s\t", d_item[i-41]);
        for (j=0; j<=7; j++)
            printf("%4.2f\t", DATA[i][j]);
        printf("\n");
    }

    printf("%s\t", d_item[6]);
    for (j=0; j<=7; j++)
        printf("%4.2f\t", DATA[50][j]);
    printf("\n\n");

    printf("I|Y\t");
    for (i=8; i<=15; i++)
        printf("%d\t", i);
    printf("\n");

    printf("Computer or Electronic Equipment\n");

    for (i=41; i<=43; i++)
    {       printf("%s\t", d_item[i-41]);
        for (j=8; j<=15; j++)
            printf("%4.2f\t", DATA[i][j]);
        printf("\n");
    }

    printf("Transporation Equipment\n");

    for (i=44; i<=46; i++)
    {       printf("%s\t", d_item[i-41]);
        for (j=8; j<=15; j++)
            printf("%4.2f\t", DATA[i][j]);
        printf("\n");
    }

    printf("%s\t", d_item[6]);
    for (j=8; j<=15; j++)
        printf("%4.2f\t", DATA[50][j]);
    printf("\n");
```

```
    printf("\n\n");


    printf("THE CASH FLOW STATEMENT          [unit: 10,000 dollar]\n");

    printf("I|Y\t");
    for (i=0; i<=7; i++)
        printf("%d\t", i);
    printf("\n");

    for (i=201; i<=213; i++)
    {    printf("%s\t", c_item[i-201]);
        for (j=0; j<=7; j++)
            printf("%4.2f\t", DATA[i][j]);
        printf("\n");
    }
    printf("\n");

    printf("I|Y\t");
    for (i=8; i<=15; i++)
        printf("%d\t", i);
    printf("\n");
    for (i=201; i<=213; i++)
    {    printf("%s\t", c_item[i-201]);
        for (j=8; j<=15; j++)
            printf("%4.2f\t", DATA[i][j]);
        printf("\n");
    }
    printf("\n\n");
    printf("NPV= %4.2f\n", DATA[213][16]);
    printf("IRR= %4.2f\n", DATA[216][1]*100);
}
```

APPENDIX B

THE RESULT OF EXPERIMENTAL SYSTEM

The Experimental System for Financial Feasibility

...... reading data from data file.

Forward Chaining Expert System for Selecting Depreciation Method
Electronic or Computer Equipment.
Usage can be devided into units UNI
Life can be measured by year or month TIM
Productity decreased by time DEC
Do not know DEF
TIM
*** Straight-Line Method ***
How Long Can the equipment be Used:->5
Transportation Equipment.
Usage can be devided into units UNI
Life can be measured by year or month TIM
Productity decreased by time DEC
Do not know DEF
DEC
Sum-of-the-years'-digits SYD
Declining-balance-method DBM
SYD
*** Sum-of-the-Years'-Digits ***
How Long Can the equipment be Used:->5
Building.
Usage can be devided into units UNI
Life can be measured by year or month TIM
Productity decreased by time DEC
Do not know DEF
TIM
*** Straight-Line Method ***
How Long Can the equipment be Used:->5
DATA completed
CALCULATION completed

I|Y ---------- item|year
PRIN ---------- principal
DEPR ---------- depreciation expense
NVAL ---------- net value
TOTA ---------- total depreciation expenses
NSAL ---------- net sales
RESI ---------- residule value
CAIN ---------- cash inflow
INVE ---------- investment
VCOS ---------- variable cost
FCOS ---------- fixed cost
COUT ---------- cash outflow
NECF ---------- net cash flow
DRAT ---------- discount rate 12 precent
PV ---------- present value
KNOW ---------- know how
IRR ---------- internal rate of return
NPV ---------- net present value

THE DEPRECIATION SCHEDULE [unit: 10,000 dollar]

I|Y	0	1	2	3	4	5	6	7
Computer or Electronic Equipment								
PRIN	0.00	149.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	26.82	26.82	26.82	26.82	26.82	0.00	0.00
NVAL	0.00	122.18	95.36	68.54	41.72	14.90	14.90	14.90
Transporation Equipment								
PRIN	0.00	20.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	6.00	4.80	3.60	2.40	1.20	0.00	0.00
NVAL	0.00	14.00	9.20	5.60	3.20	2.00	2.00	2.00
TOTA	0.00	32.82	31.62	30.42	29.22	28.02	0.00	0.00

I|Y	8	9	10	11	12	13	14	15
Computer or Electronic Equipment								
PRIN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NVAL	14.90	14.90	14.90	14.90	14.90	14.90	14.90	14.90
Transporation Equipment								
PRIN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NVAL	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
TOTA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

THE CASH FLOW STATEMENT [unit: 10,000 dollar]

I|Y	0	1	2	3	4	5	6	7
NSAL	0.00	173.15	426.10	717.60	1034.60	1417.00	1417.00	1417.00
DEPR	0.00	32.82	31.62	30.42	29.22	28.02	0.00	0.00
RESI	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
KNOW	0.00	5.40	5.40	5.40	5.40	5.40	0.00	0.00
CAIN	0.00	211.37	463.12	753.42	1069.22	1450.42	1417.00	1417.00
INVE	280.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TAX	0.00	0.00	0.00	0.00	8.67	14.45	17.23	17.23
VCOS	0.00	102.84	216.68	344.04	513.71	707.84	707.84	707.84
FCOS	0.00	149.80	205.04	296.33	416.39	535.05	501.63	501.63
COUT	280.00	252.64	421.72	640.37	938.78	1257.34	1226.69	1226.69
NECF	-280.00	-41.27	41.40	113.05	130.44	193.08	190.31	190.31
DRAT	1.00	0.89	0.80	0.71	0.64	0.57	0.51	0.45
PV	-280.00	-36.85	33.00	80.47	82.90	109.56	96.42	86.09

I|Y	8	9	10	11	12	13	14	15
NSAL	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00
DEPR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RESI	0.00	0.00	0.00	0.00	0.00	0.00	0.00	16.90
KNOW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CAIN	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00	1433.90
INVE	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TAX	17.23	17.23	34.24	34.24	34.24	34.24	34.24	34.24
VCOS	707.84	707.84	707.84	707.84	707.84	707.84	707.84	707.84
FCOS	501.63	501.63	501.63	501.63	501.63	501.63	501.63	501.63
COUT	1226.69	1226.69	1243.71	1243.71	1243.71	1243.71	1243.71	1243.71
NECF	190.31	190.31	173.29	173.29	173.29	173.29	173.29	190.19
DRAT	0.40	0.36	0.32	0.29	0.26	0.23	0.20	0.18
PV	76.86	68.63	55.79	49.82	44.48	39.71	35.46	34.75

NPV= 577.08
IRR= 31.15
Enter Goal:->CH
Enter YES or NO to indicate whether there is a optimizing goal :->YES
Enter YES or NO to indicate optimizing tax goal only :->NO
Possible optimizing!
Enter YES or NO to indicate optimizing financial reporting goal only :->NO
Enter YES or NO to indicate optimizing both goals :->YES
Use Plan B optimizing!
SUCCESS
% a.out
The Experimental System for Financial Feasibility

...... reading data from data file.

Forward Chaining Expert System for Selecting Depreciation Method
Electronic or Computer Equipment.
Usage can be devided into units UNI
Life can be measured by year or month TIM
Productity decreased by time DEC
Do not know DEF
DEC
Sum-of-the-years'-digits SYD
Declining-balance-method DBM
SYD
*** Sum-of-the-Years'-Digits ***
How Long Can the equipment be Used:->10
Transportation Equipment.
Usage can be devided into units UNI
Life can be measured by year or month TIM
Productity decreased by time DEC
Do not know DEF
DEC
Sum-of-the-years'-digits SYD
Declining-balance-method DBM
SYD
*** Sum-of-the-Years'-Digits ***
How Long Can the equipment be Used:->7
Building.
Usage can be devided into units UNI
Life can be measured by year or month TIM
Productity decreased by time DEC
Do not know DEF
TIM
*** Straight-Line Method ***
How Long Can the equipment be Used:->5
DATA completed
CALCULATION completed

I|Y ---------- item|year
PRIN ---------- principal
DEPR ---------- depreciation expense

NVAL ---------- net value
TOTA ---------- total depreciation expenses
NSAL ---------- net sales
RESI ---------- residule value
CAIN ---------- cash inflow
INVE ---------- investment
VCOS ---------- variable cost
FCOS ---------- fixed cost
COUT ---------- cash outflow
NECF ---------- net cash flow
DRAT ---------- discount rate 12 precent
PV ---------- present value
KNOW ---------- know how
IRR ---------- internal rate of return
NPV ---------- net present value

THE DEPRECIATION SCHEDULE [unit: 10,000 dollar]

I|Y	0	1	2	3	4	5	6	7
Computer or Electronic Equipment								
PRIN	0.00	149.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	24.38	21.94	19.51	17.07	14.63	12.19	9.75
NVAL	0.00	124.62	102.67	83.17	66.10	51.47	39.28	29.53
Transporation Equipment								
PRIN	0.00	20.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	4.50	3.86	3.21	2.57	1.93	1.29	0.64
NVAL	0.00	15.50	11.64	8.43	5.86	3.93	2.64	2.00
TOTA	0.00	28.88	25.80	22.72	19.64	16.56	13.48	10.40

I|Y	8	9	10	11	12	13	14	15
Computer or Electronic Equipment								
PRIN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	7.31	4.88	2.44	0.00	0.00	0.00	0.00	0.00
NVAL	22.21	17.34	14.90	14.90	14.90	14.90	14.90	14.90
Transporation Equipment								
PRIN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
DEPR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NVAL	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
TOTA	7.31	4.88	2.44	0.00	0.00	0.00	0.00	0.00

THE CASH FLOW STATEMENT [unit: 10,000 dollar]

I|Y	0	1	2	3	4	5	6	7
NSAL	0.00	173.15	426.10	717.60	1034.60	1417.00	1417.00	1417.00
DEPR	0.00	28.88	25.80	22.72	19.64	16.56	13.48	10.40

RESI	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
KNOW	0.00	5.40	5.40	5.40	5.40	5.40	0.00	0.00
CAIN	0.00	207.43	457.30	745.72	1059.64	1438.96	1430.48	1427.40
INVE	280.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TAX	0.00	0.00	0.00	0.00	9.47	15.40	16.11	16.36
VCOS	0.00	102.84	216.68	344.04	513.71	707.84	707.84	707.84
FCOS	0.00	145.86	199.22	288.63	406.81	523.59	515.11	512.03
COUT	280.00	248.70	415.90	632.67	929.99	1246.83	1239.05	1236.23
NECF	-280.00	-41.27	41.40	113.05	129.65	192.13	191.43	191.17
DRAT	1.00	0.89	0.80	0.71	0.64	0.57	0.51	0.45
PV	-280.00	-36.85	33.00	80.47	82.39	109.02	96.98	86.48

I|Y	8	9	10	11	12	13	14	15
NSAL	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00	1417.00
DEPR	7.31	4.88	2.44	0.00	0.00	0.00	0.00	0.00
RESI	0.00	0.00	0.00	0.00	0.00	0.00	0.00	16.90
KNOW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CAIN	1424.31	1421.88	1419.44	1417.00	1417.00	1417.00	1417.00	1433.90
INVE	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TAX	16.62	16.82	33.84	34.24	34.24	34.24	34.24	34.24
VCOS	707.84	707.84	707.84	707.84	707.84	707.84	707.84	707.84
FCOS	508.94	506.51	504.07	501.63	501.63	501.63	501.63	501.63
COUT	1233.40	1231.16	1245.75	1243.71	1243.71	1243.71	1243.71	1243.71
NECF	190.91	190.71	173.69	173.29	173.29	173.29	173.29	190.19
DRAT	0.40	0.36	0.32	0.29	0.26	0.23	0.20	0.18
PV	77.11	68.77	55.92	49.82	44.48	39.71	35.46	34.75

NPV= 577.51
IRR= 31.15
Enter Goal:->%

APPENDIX C

GLOSSARY

Artificial intelligence A subdivision of Computer Science devoted to creating computer software and hardware that attempt to produce results such as those produced by human.

Backward chaining A goal-driven approach in which the inference engine starts from an expectation of what is to happen, then seeks evidence that supports (or contradicts) the expectation.

Break-even point The level of activity at which the fixed costs of an operation are just covered by the contribution from sales. At this point neither a profit nor a loss ensures.

Cash flow Cash provided and used by operating, investing, and financing activities during the period.

Depreciation The accounting process of allocating the cost of tangible assets to expense I in a systematic and rational manner to those periods expected from the use of the asset.

Expert system A decision-making and/or problem-solving package of computer hardware and software that can reach a level of performance comparable to -- or even exceeding that of -- a human expert in some specialized and usually narrow problem area.

Fixed cost A type of cost where the total expenditure does not vary with the level of activity or output.

Forward chaining A data driven approach in which the inference engine starts from available information, then try to draw conclusions.

Going concern The concept stating that the entity is a viable concern and that in the normal course of events it is not expected to fail.

Inference engine A computer program that provides a methodology for reasoning about information in the knowledge base and for formulating conclusions.

Internal rate of return (IRR) The rate of discount that brings the present value of all the cash flows associated with a capital investment to zero. It measures the effective yield on the investment. If this yield is greater than the 'hurdle rate' the investment is deemed to be financially desirable and vise versa.

Know how Intangible assets such as pattern, formula, and copyright.

Knowledge base A data base contains knowledge necessary for understanding, formulating, and solving problems.

Net present value (NPV) A positive or negative value arrived at by discounting the cash flow from a capital project by the desired rate of return. If the value is positive, it means that the project is financially desirable and vice versa.

Present value (PV) A sum calculated by discounting the stream of future cash flow from a project using an interest rate equal to the desired rate of return. It differs from *net present value* in that the amount of the investment is not included in the cash flow.

Principal Total cost for an asset at the time of acquisition. It is usually the original book value of an asset.

Profit margin A percentage calculated by dividing net income by net sales for the period. It provides some indication of the buffer available in case of higher costs or lower sales in the future.

Profitability index A measure for assessing the relative merit of an investment by expressing the present value of the future cash flows as a percentage of the invest amount.

Residual value The estimated fair value of the asset at the time of disposal.

Variable cost A type of cost where the total expenditure varies in proportion to activity or output.

APPENDIX D

ACRONYMS

BCESD	Backward Chaining Expert System Design
BCR	Benefit-Cost Rate
CAIN	Cash Inflow
COUT	Cash Outflow
COPASS	Corporate Portfolio Analysis Support System
DBM	Declining Balance Method
DEC	Decrease
DEF	Default
DEPR	Depreciation Expense
DRAT	Discount Rate
EDGAR	Electronic Data Gathering, Analysis and Retrieval
FCESD	Forward Chaining Expert System Design
FCOS	Fixed Cost
FMS	Flexible Manufacturing System
FSA	Financial Statement Analyzer
INVE	Investment
IRR	Internal Rate of Return
KBDSS	Knowledge-Based Decision Support System
KNOW	Know How
MIRR	Modified Internal Rate of Return
NECF	Net Cash Flow
NSAL	Net Sales
NPV	Net Present Value

NPVR	Net Present Value Rate
NVAL	Net Value
PI	Profitability Index
PRIN	Principal
PV	Present Value
RESI	Residual Value
SEC	The US Securities and Exchange Commission
SFAM	Structured Factors Analysis Support Model
SYD	Sum of the Years' Digits
TIM	Time
TOTA	Total
TV	Terminal Value
UNI	Unit
VCOS	Variable Cost

2

VITA

Min Zhang

Candidate for the Degree of

Master of Science

Thesis: USING A MATHEMATICAL MODEL IN CONJUNCTION WITH EXPERT SYSTEM TO ASSIST INVESTMENT DECISIONS

Major Field: Computer Science

Biographical:

Personal Data: Born in Shanghai, P. R. China, On August 25, 1964, the son of Jianchang Zhang and Xiulian Jiao.

Education: Graduated from Shanghai Electronic Technology College, Shanghai, P. R. China in July 1983; received Bachelor of Science Degree in Engineering in Computer Application from Shanghai Second Polytechnic University, Shanghai, P. R. China in July 1988. Completed the requirements for the Master of Science degree with a major in Computer Science at Oklahoma State University in July 1996.

Experience: Grew up in Shanghai, P. R. China; employed by Shanghai Electronic Technology College as a technician from 1983 to 1987; employed by Bank of China as an associate engineer from 1987 to 1990.

Professional Membership: Tulsa Computer Society.