National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Canada

Interactive Multiple Criteria Optimization for Capital Budgeting

By
Savvas Pissarides

Master's of Science thesis
System Science programme

University of Ottawa

ISBN 0-315-80027-5

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Acknowledgements

I wish to thank my supervisors Dr. Daniel Lane and Dr. Jean-Michel Thizy Associate Professors, Fuculty of Administration, University of Ottawa for their constant guidance, dedication and support. They never failed to point out how I could learn from my mistakes and they where always there for me even when distance was an issue. I thank you and with respect I dedicate this thesis to you.

Also I would like to thank Dr. Surendra Rawat, Adjoint Professor, Fuculty of Administation, University of Ottawa for identifying the problem, and for providing direction and guidance in developing the methodology and user requirements for the PC based tool DSS ORA. This research was partially funded by Bell Canada, where Dr. Rawat is an Associate Director. Surendra, I really appreciate all the extra miles you went for me.

Also I would like to thank Chris and Hastings for sharing that hole (System Science research assistants office) with me, where we shared many laughs and frustrations.

Abstract

This thesis presents a capital budgeting problem faced by a major telecommunications company. The purpose of this thesis is to address the capital budgeting problem in order to establish a framework for the measurement and evaluation of alternative capital allocation decisions which are compatible with the mission of the company. The solution method follows three major avenues of optimization: multiple criteria, multiple constraints and interactivity. The problem is solved using the Analytic Hierarchy Process to obtain an initial solution which is then improved by an interactive method allowing users to direct the search for an acceptable allocation. The method is implemented by a decision support system hinging on a graphic user interface. The support system has been used by practitioners to evaluate alternatives of a real problem. Results and enhancements are discussed.

Contents

Chapter 1

Introduction

Capital budgeting can be defined as the allocation of scarce resources between alternatives so as to meet objectives in the best possible way (Bromwich 1979). Capital budgeting is performed by widely different decision-makers from individual people and families, to governments and firms. For example, individual people purchase consumer goods, like houses. Governments plan construction of roads, and invest in agriculture and space research. Firms invest in production plants, research, development and marketing. All three entities are faced with a large number of possible investment opportunities to be financed out of a limited amount of funds. This thesis concentrates specifically on the investment decisions of firms.

Companies throughout the world are faced with the same capital budgeting problem. The problem of capital budgeting arises when different and competing alternative activities are to be funded out of a limited budget and not all alternatives can be totally funded. Therefore, decision-makers must decide which activities are most important to the company and should be given priority over the other activities.

This is how the problem of capital budgeting is usually resolved. First, decision-makers determine the total amount of capital to be allocated and what activities should be considered for funding. Typically the capital allocation exercise involves the choice of criteria under which the prioritization of alternatives is made. It is essential that decision-makers consider various criteria compatible with the direction of the firm and attempt to make decisions so as to achieve these criteria. Capital allocations also have to be made subject to organizational and operational constraints. The process of capital budgeting becomes more complicated in the absense of full information about the value of different alternative activities measured against the criteria set.

This research thesis addresses the capital budgeting problem in firms in order to establish a framework for the measurement and evaluation of alternative capital allocation decisions which are compatible with the mission of the company. Specifically, this thesis presents the development of an analytical method applied to the capital budgeting problem of a large telecommunications company. The definition of this capital budgeting problem is laid out in the next section. The last section of this introduction outlines the organization of the rest of the thesis.

1.1 Problem definition

This thesis considers the annual capital budgeting problem of a single functional group within a large telecommunications company where a set of discrete project alternatives need to be funded. Each alternative represents a capital programme which is responsible for the modernization of parts of the telecommunications network. These programmes are presented to the company's central capital budgeting decision-making body responsible for funding projects over all functional groups in the firm. A capital programme requires a certain level of funding that must remain within a limited annual budget. The central capital budgeting decision-makers must allocate the total budget among groups, given each group's capital programme submissions for the coming year. The objective of the allocation process is to fund programmes in a way that is most profitable for the company.

The capital budgeting decision-making body faces a number of difficulties. First, it may not be possible to allocate funds to the programmes at the level submitted by the functional groups. Therefore, the programmes need to be prioritized using criteria that reflect the company's mission. In the current practice, the central decision-makers have to define these criteria explicitly, and evaluate the expected contribution of each of the programmes to the criteria. These decisions involve human opinions which are formed by drawing upon the knowledge, influence and experience of each member in the decision-making body.

Second, in the allocation process, the decision-makers must consider a number of organizational and operational constraints. For example, there may exist a number of interdependencies where programmes cannot operate without the contribution of a certain number of other programmes funded at a minimum level. Such programme interdependencies must be accounted for in the allocation decisions.

Third, the managers of the functional groups may disagree with funding decisions made by the central capital decision-makers. Legitimate problems will require the decision-making body to reconsider an allocation by modifying the rules used to define the company criteria, the contributions of the programmes to these criteria, and the level of programme dependencies. The appeal-feedback mechanism provides evidence of the complexity and the uncertainty existing in the actual budgeting decision process. Much of the difficulties come from qualitative and subjective views of the company's missions and how the programmes are related to accomplishing this mission. What is needed is a more explicit, quantitative evaluation of the company's goals and how the alternative capital programmes can achieve them.

It is the purpose of this thesis to develop and present an analytical evaluation and Decision Support System (DSS) that can help the capital budget decision-makers analyze and explore effective decision options.

This thesis solves this particular capital budgeting problem by developing the following methodology:

1. Multiple Objective Formulation : The problem is first formulated as a multiple objective linear programme. The objectives represents criteria important to the companies missions.

2. Reduction : The multiple objective space is reduced to a single objective space by using empirical evidence from the decision makers. The single objective functions coefficients are estimated and an initial solution is calculated.

3. Interaction : The initial solution is improved by using an interactive user guided search for better solutions within the region defined by the constaints.

The methodogy is then applied in developing a Decision Support System for solving the capital budgeting of a major telecommunications company.

1.2 Plan of the thesis

The remainder of the thesis includes the following chapters:

Chapter 2 presents a survey of literature relevant to the thesis. The survey is divided into two parts. The first part covers literature related to the quantitative analysis of the capital budgeting problem starting with heuristic methods and ending with multiple criteria methods. The second part surveys relevant literature on multiple criteria decision-making methods, with emphasis on interactive decision-making aids.

Chapter 3 presents the formulation of the specific capital budgeting problem. The problem is presented as a multiple criteria problem and the method for solving the problem using an interactive approach is developed.

Chapter 4 presents the application of the method developed in the previous chapter. The method is applied to a large telecommunications company's annual problem of allocation of its budget to its corporate network modernization programmes. The Corporate Network Modernization group within the company is responsible for capital programmes that are required for the modernization of the various components of the company's telecommunications network. This chapter includes the presentation and development of the decision suport system DSS ORA. Results obtained are also presented.

In Chapter 5 the results obtained in the previous chapter, are discussed relative to the potential use of the tool for the company's annual budgeting problem. Problem extensions are also discussed in this closing chapter.

Chapter 2

Literature Review

This chapter presents a survey of the literature relevant to the specific capital budgeting problem described in the previous chapter and to the approach that this thesis will use to solve the problem. Section 2.1 presents heuristic and mathematical programming methods used to solve capital budgeting problems. Section 2.2 presents methods for solving multiple criteria decision making problems.

2.1 Capital Budgeting

Michael Bromwich (1979) explains that while economics is concerned with allocation of scarce resources between alternative uses, capital budgeting concentrates on these allocations over time. He defines capital budgeting as the theory that attempts to answer the following questions:

1. What specific investments should the firm accept?
2. What total amount of capital expenditure should the firm undertake?
3. How should the portfolio of projects be financed?

The next section examines various capital budgeting methods that seek to answer the first two of these questions.

2.1.1 Heuristic Methods

Dean (1951a, 1951b) was the first to propose a method for solving capital budgeting problems. He presented a technique for rationing capital among competing investment proposals based on a ranking of candidates using their respective indices of Net Present Value (NPV) relative to their capital requirements. The net present value is the difference in the present value of the cash inflows and outflows which are discounted using the firm's required rate of return.

Consider the following capital budgeting problem presented in Example 2.1 where the ratio of the net present value to the capital required is used to rank the three projects that need funding:

Example 2.1

Project i	NPV	Capital Required c_i	Ratio $\frac{NPV_i}{c_i}$
A	75	75M	1.0
B	90	135M	0.75
C	50	150M	0.3
Total Capital available		135M	

According to the ranking method proposed by Dean, project A is selected because it has the highest ratio with a net present value of 75 and leaves 60M. Notice that the highest total NPV is achieved by selecting project B with net present value of 90. This example does not consider partial project funding.

Lorie and Savage (1955) articulated some weaknesses in using such a heuristic method. They presented examples demonstrating:

1. Problems that develop because multiple projects under consideration are not independent.

2. Problems that develop when capital is rationed in more than a single time period.

3. Problems that develop when analysing projects that have both cash inflows and outflows dispersed over their lives.

On the other hand, it has been suggested by Fogler (1972) and later on by Forsyth and Owen (1981), that simple heuristic methods, such as the net present value index, produce reasonably good approximations of solutions attained by complex mathematical programming methods. Furthermore, heuristic methods are very easy to use and do not require any theoretical knowledge in the way that mathematical programming methods usually do.

2.1.2 Mathematical Programming Methods

Charnes, Cooper and Miller (1959) and Weingartner (1963) were the first to formulate a linear programming model to solve capital budgeting problems. The first formulated a linear programming model to assist the firm in performing capital budgeting considering both operating decisions and financial planning. The latter formulated a multi-period linear programming model that maximizes the total net present value given the capital requirements of the projects for each year. Formulation 2.1 shows the model proposed by Weingartner:

Formulation 2.1

$$\max \quad \sum_{j=1}^{n} b_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} c_{tj} x_j \leq C_t \qquad \text{for } t = 1 \ldots T$$

$$0 \leq x_j \leq 1 \qquad \text{for } j = 1 \ldots n$$

where

b_j is the net present value of project j,

c_{tj} is the capital required by project j at period t,

C_t is the maximum total permissible expenditure during period t,

x_j is the fraction of project j accepted,

n is the number of projects,

T is the number of time periods.

Optimal solutions to Formulation 2.1 can be easily obtained using the simplex algorithm. To illustrate the general approach, consider the following four period capital budgeting problem with four projects:

Example 2.2

Project	NPV_j	c_{1j}	c_{2j}	c_{3j}	c_{4j}
1	90	15	20	20	15
2	40	10	15	20	5
3	10	10	0	0	4
4	37	15	10	10	10
Budget available		$C_1 = \$40$	$C_2 = \$50$	$C_3 = \$40$	$C_4 = \$35$

The linear programming formulation is as follows:

$$\begin{aligned} \max \quad & 90x_1 + 40x_2 + 10x_3 + 37x_4 \\ \text{subject to} \quad & 15x_1 + 10x_2 + 10x_3 + 15x_4 \leq 40 \\ & 20x_1 + 15x_2 + 0x_3 + 10x_4 \leq 50 \\ & 20x_1 + 20x_2 + 0x_3 + 10x_4 \leq 40 \\ & 15x_1 + 5x_2 + 4x_3 + 10x_4 \leq 35 \\ & 0 \leq x_j \leq 1 \quad \text{for } j = 1 \ldots 4 \end{aligned}$$

The above linear programme is solved using Lindo/PC (Schrage 1986) and the optimum solution is $x_1 = 1,\ x_2 = 0.5,\ x_3 = 0.5,\ x_4 = 1$, with a total net present value of 152.

In Example 2.2 fractional allocation is allowed. However, sometimes it is not possible to have fractional projects. To select nonfractional projects, $0-1$ integer programming is required, where the decision variables are taken to be $x_j = 0$ or 1, indicating that the j^{th} project is rejected or accepted. Weingartner gave a $0-1$ integer programming formulation by adding to Formulation 2.1 the constraint:

$$x_j \in \{0,1\},\ x_j \in I, \text{ for } j = 1 \ldots n$$

thus forcing x_j to be either zero or one.

For example, consider the addition of this extra constraint to Example 2.2. The problem is solved again using Lindo/PC and the optimum solution is $x_1 = 1,\ x_2 = 1,\ x_3 = 1,\ x_4 = 0$ with total net present value of 140. Note that this solution could not have been discovered by simply rounding the linear programming solution. The best solution obtained by rounding is $x_1 = 1,\ x_2 = 0,\ x_3 = 1,\ x_4 = 1$ with total net present value of 137.

Generally, integer programmes are relatively harder to solve. For example, Clark et al. (1984) reported an integer programming formulation with 15 constraints and 28 projects in which four of the six solution algorithms tested failed to arrive at the optimal solution in 5 minutes of CPU time on an IBM 360-65; the two algorithms that did locate the optimal solution took 118 and 181 seconds. Similar problem formulations omiting integer requirements can be solved in almost 10 to 100 time units faster than the problem formulations with the integer requirements. Furthermore, there is no equivalent integer programming algorithm comparable to the simplex method. Hence, techniques based on an intelligent implicit enumeration have been devised for integer programmes and are referred to as branch-and-bound.

Weingartner's work has been adapted and extended in various ways. Other constraints like capacity, liquidity, and manpower have been added, and different objective functions have been proposed. To meet the complexity of various capital budgeting problems, other criteria for optimisation were added to the net present value until the multiple goal nature of capital budgeting problems was recognized.

Deckro, Spahr, and Herbert (1985) presented a procedure to model a goal programming based capital budgeting model as a non-linear polynomial expression where inter- and intra-goal trade-offs can be achieved subject to the marginal rates of substitution between preference levels. This was achieved by modelling the objective function as an expression of the deviational variables from each goal constraint. They suggested that the ability to vary the powers and coefficients of the deviational variables in the objective function replaces the need for strict pre-emption. The proposed model has three basic sets of goal constraints: maximization of NPV, cash flow budgeting and control of risk. Formulation 2.2 illustrates the model proposed by Deckro, Spahr and Herbert (1985).

Formulation 2.2

$$\text{minimize} \quad \sum_{t=2}^{T} \sum_{k=1}^{K} \sum_{j=1}^{2} w_{jkt} d_{jkt}^{P_{jkt}}$$

Subject to:

(1) NPV Goal $$\sum_{i=1}^{N} b_i x_i + d_{111} = \sum_{i \in S} b_i$$

(2) Cash Flow $$\sum_{i=1}^{N} -c_{i1} x_i + d_{121} - d_{221} = C_1$$

(3) $$\sum_{i=1}^{N} -c_{it} x_i + (1 + r_1) d_{12(t-1)} + d_{12t} + (1 + r_2) d_{22(t-1)} - d_{22t} = C_t, \; t = 2, \ldots, T$$

(4) Risk $$\sum_{i-1}^{N} a_i x_i \beta_i - \beta \sum_{i=1}^{N} a_i x_i + d_{131} - d_{231} = 0$$

(5) Non Negavity $$0 \leq x_i \leq 1, i = 1, \ldots, N$$

where

N is the number of projects,

w_{jkt} is a numeric weight for the deviation d_{jkt},

P_{jkt} is non-negative power associated with deviation d_{jkt},

S is the set of all indices of projects with positive NPV,

b_i is the net present value of all cash flows associated with project i

d_{111} is the deviational variable measuring the underachievement of the maximum positive net present value given by $\sum_{i \in S} b_i$,

$-c_{it}$ is the cash flow of project i in period t,

r_1 is interest rate received when lending funds,

r_2 is interest rate paid when borrowing funds,

C_t is the capital budget for period t from investments prior to $t = 1$,

T is the number of periods,

$K = 3$ is the number of goals namely NPV, Cash flow and Risk

d_{12t} is the amount of lending in period t,

d_{22t} is the amount of borrowing in period t,

a_i is the present value of the cash outflows for project i,

β_i is the systematic risk of project i,

β is the target level of systematic risk for the firm,

d_{131} is the amount the systematic risk below the target,

d_{231} is the amount the systematic risk is above the target, x_i is a variable representing the fraction of project i.

Equation (1) represents the maximization of the NPV goal constraint; equations (2) and (3) represent the cash flow goal constraints; equation (4) represents the control of risk goal constraint; and finally, equation (5) represent the bounds on the decision variables x_i.

Spronk (1981) extended the use of goal programming in capital budgeting by proposing an interactive multiple goal programming method. The method is based on a mutual and successive interplay between a decision-maker and an analyst. According to Spronk such interactive methods do not require explicit representation of the decision-maker's preference function or representations of trade-offs among competing objectives. Spronk also shows that these methods are superior to conventional goal programming techniques for solving capital budgeting problems.

2.2 Multiple Criteria Decision Making

Multiple criteria decision making (MCDM) is concerned with the methods and procedures by which multiple criteria can be formally incorporated into the analytical process. MCDM has two distinct areas: multiattribute decision analysis and multiple objective mathematical programming. Multiattribute decision analysis is most often applicable to problems with a small number of alternatives in an environment of uncertainty. On the other hand, multiple objective mathematical programming is most often applied to deterministic problems in which the number of feasible alternatives is large.

Section 2.2.1 examines multiple objective mathematical programming and in particular, interactive methods (Section 2.2.2) and goal programming (Section 2.2.3). Section 2.2.4 presents the Analytic Hierarchy Process as a special case of multiattribute decision analysis.

2.2.1 Multiple Objective Mathematical Programming

Methods for solving single objective mathematical programming problems for capital budgeting have been studied extensively for the past 30 years. However, decision-makers have found that complex real-world problems involve more than one objective each of which represents a criterion that is important to the company. For example, consider the following capital budgeting problem with five ideal objectives (optimising all above may not be feasible):

max { net present value }
min { capital investment }
min { annual operating expenses }
max { investment in projects related to environmental protection }
max { investment in projects in a given geographical area }

A traditional method of solving a mathematical programme with k objectives is first to assess the decision-maker's utility function U and then solve the mathematical programming problem:

$$\begin{aligned} \max \ & \{U(z_1, z_2, \ldots, z_k)\} \\ & \mathbf{c}^i\mathbf{x} = z_i \text{ where } 1 \leq i \leq k \\ \text{s.t. } & \mathbf{x} \in S \end{aligned}$$

where

k is the number of criteria of optimiozation.
$\mathbf{c}^i$ is the vector of linear objective function coefficients of the ith criterion.
z^i is the criterion value, the objective function value of the ith objective.
S is the feasible region defined by a set of linear constraints.
$\mathbf{x}$ is the decision variable vector of size n which defines a point in decision space.

The main difficulty with this approach is that it may not be possible to obtain a mathematical representation of the decision-maker's utility function U. Without explicit knowledge of the decision-maker's utility function the problem is solved by using a different

formulation and implicit information from the decision maker. For example, a multi-objective linear programme can be defined as:

$$\begin{aligned} &\max \{\mathbf{c}^1\mathbf{x} = z_1\} \\ &\max \{\mathbf{c}^2\mathbf{x} = z_2\} \\ &\qquad \vdots \\ &\max \{\mathbf{c}^k\mathbf{x} = z_k\} \\ &\text{s.t.} \quad \mathbf{x} \in S \end{aligned}$$

A frequently used method in multiple objective linear programming is the point estimate weighted sums approach where each objective $\mathbf{c}^i\mathbf{x}$ is multiplied by a strictly positive scalar weight λ_i. Then, the k weighted objectives are summed to form a composite objective function. The composite objective function is written as $\boldsymbol{\lambda}^T\mathbf{C}\mathbf{x}$, where $\mathbf{C}$ is the $k \times n$ criterion matrix whose rows are the $\mathbf{c}^i$. Each weighting vector $\boldsymbol{\lambda} \in R^k$ is normalized so that its elements remain non-negative and sum to one. Define Λ, the set of all such weighting vectors by:

$$\Lambda = \left\{ \boldsymbol{\lambda} \in R^k \mid \lambda_i \geq 0, \quad \sum_{i=1}^{k} \lambda_i = 1 \right\}$$

Therefore the problem reduces to:

$$\max\{\boldsymbol{\lambda}^T\mathbf{C}\mathbf{x} \mid \mathbf{x} \in S\}$$

The point estimate weighted sums method converts a multiple objective linear programme into a single objective linear programme that can be solved using standard linear programming algorithms. However, the difficulty lies in finding a "good" weighting vector. The following is an example with 3 objectives and 4 variables.

Example 2.3

$$\max \; 3x_1 + x_2 + 2x_3 + x_4 = z_1 \tag{6}$$
$$\max \; x_1 - x_2 + 2x_3 + 4x_4 = z_2 \tag{7}$$
$$\max \; x_1 + 5x_2 + x_3 + 2x_4 = z_3 \tag{8}$$

subject to:

$$\begin{aligned} 2x_1 + x_2 + 4x_3 + 3x_4 &\leq 60 \\ 3x_1 + 4x_2 + x_3 + 2x_4 &\leq 60 \end{aligned}$$

Suppose that (6) is twice as important as (7) and that (8) is as important as (6). Thus, choose $\boldsymbol{\lambda} = (0.4, 0.2, 0.4)$. The composite objective function is:

$$\max \; 1.8x_1 + 2.2x_2 + 1.6x_3 + 2x_4$$

The single objective mathematical programme is solved and the optimum solution is:

$$x_1 = 0, \; x_2 = 6, \; x_3 = 0, \text{ and, } \; x_4 = 18$$

giving

$$z_1 = 24, \; z_2 = 66, \; z_3 = 66$$

2.2.2 Interactive methods

Interactive methods are characterized by phases of decision-making alternating with phases of computation. This pattern is repeated until an optimal or near optimal solution is obtained. At each iteration a solution or a group of solutions is generated for examination by the decision-maker who provides information to the solution procedure.

The feedback between man and model enables the decision-maker to learn more about the problem under consideration. It enables the decision-maker to appreciate more fully the range of possibilities in the feasible region and how the objectives trade-off against one another. This should enable the decision maker to know better where to look for improved solutions and how to recognize a final solution upon encountering it. Among the first interactive methods developed are STEM (Benayoun et al. 1971) and the Geoffrion-Dyer-Feinberg procedure (Geoffrion et al. 1972). This section describes Zionts and Wallenius' (Z-W) method (1976 and 1983) for solving multiple objective linear programmes. The Z-W method considers pseudoconcave utility functions, thus considering only extreme points of the region defined by the constraints. The method guides the user in an exploration of solutions following a path of adjacent extreme points on the boundary.

The Z-W method is classified as a reduced weighting vector space method for solving the multiple criteria problem:

$$\begin{aligned} \max \{\mathbf{c}^1\mathbf{x} &= z_1\} \\ \max \{\mathbf{c}^2\mathbf{x} &= z_2\} \\ &\vdots \\ \max \{\mathbf{c}^k\mathbf{x} &= z_k\} \\ \text{s.t. } \mathbf{A}\mathbf{x} &= \mathbf{b} \end{aligned}$$

where $\mathbf{A}$ is a $m \times n$ full row rank matrix, $\mathbf{b}$ is an m-vector and $\mathbf{x}$ is the n-decision variable-vector . The method operates by iteratively presenting the user trade-off questions that may lead to points on the boundary of the surface defined by the constraints. From the responses, portions of the weighting vector space $\Lambda = \{\lambda \in R^k \mid \lambda_i > 0, \sum_{i=1}^{k} \lambda_i = 1\}$ are eliminated. The process continues until Λ has been reduced to a small enough region for a small solution to be identified.

Let $\mathbf{B}$ denote the basis, $\mathbf{C}_B$ the basic columns of criterion matrix $\mathbf{C}$, $\mathbf{C}_N$ the non-basic columns, and $\mathbf{N}$ the non-basic columns of the constraint matrix $\mathbf{A}$. Then let $\mathbf{W}$ denote the $k \times (n - m)$ reduced cost matrix where

$$\mathbf{W} = \mathbf{C}_N - \mathbf{C}_B\mathbf{B}^{-1}\mathbf{N}$$

Let $\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^{n-m}$,the trade-off vectors, be the columns of the reduced cost matrix $\mathbf{W}$. Based on the above definitions the following is a brief summary of the Z-W algorithm:

Step 1: Let $\Lambda^0 = \Lambda$. Using an arbitrary vector of positive weights $\lambda \in \Lambda^0$ (a λ-vector with equal weights is recommended), solve the composite LP

$$\max \{\lambda^T \mathbf{C}\mathbf{x} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$$

Let $\mathbf{x}^0$ denote the resulting solution and $\mathbf{z}^0$ its criterion vector. Set iteration counter to $h = 0$.

Step 2: For every non-basic variable, generate trade-off vectors $\mathbf{w}^1, \mathbf{w}^2 \ldots \mathbf{w}^{n-m}$ and present them to the decision-maker. For each trade-off vector, the decision-maker may answer:

- Yes (likes the trade-off).
- No (dislikes the trade-off).
- I don't know (unable to decide).

If none is preferred terminate with solution $\mathbf{x}^h$, $\mathbf{z}^h$.

Step 3: Depending on whether or not the decision-maker likes a trade-off vector $\overline{\mathbf{w}}$ write λ- constraints:

$\lambda^T \overline{\mathbf{w}} \geq \epsilon$	For each yes response
$\lambda^T \overline{\mathbf{w}} \leq -\epsilon$	For each no response

Employing the λ-constraints to reduce Λ^h, form Λ^{h+1}.

Step 4: Find a vector $\lambda^{h+1} \in \Lambda^{h+1}$. If no such vector λ^{h+1} exists, delete the oldest set of λ-constraints, and repeat. To obtain a good representation of Λ^{h+1}, compute the middlemost point which is the point where the minimum slack from the bounding level curves of the λ-constraints defining Λ^{h+1} is maximized.

Step 5: Using λ^{h+1}, solve the composite LP and obtain $\mathbf{x}^{h+1}$. Let $h = h + 1$. Go to step 2.

Steuer (1986) observed that Zionts and Wallenius' algorithm is not an ad-hoc procedure because the questions posed by the algorithm can be answered deterministically when one knows the decision-maker's utility function. He noticed that the algorithm can be very effective in rapidly reducing the weighting vector space in early iterations. However, the method cannot converge to nonextreme solutions.

2.2.3 Goal Programming

Initially goal programming (GP) was conceived as an application of single objective linear programming by Charnes and Cooper (1961) and gained popularity in the 1960s and 70s from the works of Ijiri (1965), Lee (1972) and Ignizio (1976). The purpose of goal programming is to establish a goal level of achievement for each criterion. Goal programming is distinguished from linear programming in that GP:

- Views objectives as goals
- Assigns priorities or weights to the achievement of the goals
- Uses deviational variables d_i^+ and d_i^- to measure overachievement and underachievment from target levels t_i
- Minimizes the weighted sums of deviational variables.

A multiple objective problem can have the following three types of goal criteria:

- Greater or less than.
- Equality.
- Range.

Therefore, a GP problem with one of each type of goal criterion is expressed as

$$\begin{array}{lll} \text{goal } \{\mathbf{c}^1\mathbf{x} = z_1\} & & (z_1 \geq t_1) \\ \text{goal } \{\mathbf{c}^2\mathbf{x} = z_2\} & & (z_2 = t_2) \\ \text{goal } \{\mathbf{c}^3\mathbf{x} = z_3\} & & (z_3 \in [t_3^a, t_3^b]) \\ \text{s.t.} & \mathbf{x} \in S & \end{array}$$

The information in parentheses on the right specifies the values of the variables z_i to be achieved in relation to the t_i target values.

The Archimedian model and the preemptive model are the main models in goal programming. Using the Archimedian model, the problem is reduced to a single objective by minimizing the weighted sum of the deviations from the goals. For example, the above GP is formulated as:

$$\begin{array}{lllllllllll} \min & \{w_1^- d_1^- & + & w_2^+ d_2^+ & + & w_2^- d_2^- & + & w_3^+ d_3^+ & + & w_3^- d_3^-\} & \\ \text{s.t.} & \mathbf{c}^1\mathbf{x} & + d_1^- & & & & & & & \geq & t_1 \\ & \mathbf{c}^2\mathbf{x} & & - d_2^+ & + & d_2^- & & & & = & t_2 \\ & \mathbf{c}^3\mathbf{x} & & & & & + d_3^- & & & \geq & t_3^a \\ & \mathbf{c}^3\mathbf{x} & & & & & & - d_3^+ & & \leq & t_3^b \\ & & & & & & & \mathbf{x} & & \in & S \\ & & & & & & & \text{all } d\text{'s} & & \geq & 0 \end{array}$$

where the w's are positive penalty weights and the d's are the undesirable deviational variables. The Archimedian GPs can be solved using conventional linear programming algorithms.

In preemptive (lexicographic) goal programming, the goals are grouped according to priorities. For example, consider:

$$\begin{array}{lll} \text{goal } \{\mathbf{c}^1\mathbf{x} = z_1\} & & P_1(z_1 \geq t_1) \\ \text{goal } \{\mathbf{c}^2\mathbf{x} = z_2\} & & P_2(z_2 \leq t_2) \\ \text{goal } \{\mathbf{c}^3\mathbf{x} = z_3\} & & P_3(z_3 = t_3) \\ \text{s.t.} & \mathbf{x} \in S & \end{array}$$

where P_j identifies the goals at priority level j and is larger than P_{j+1}. The above GP can be formulated in the following way:

$$\text{lex min} \quad \{d_1^+ \ , \ d_2^- \ , \ (d_3^+ + d_3^-)\}$$

$$\begin{array}{lrrrrrrr}
\text{s.t.} & \mathbf{c}^1\mathbf{x} & - d_1^+ & & & & \leq & t_1 \\
& \mathbf{c}^2\mathbf{x} & & + d_2^- & & & \geq & t_2 \\
& \mathbf{c}^3\mathbf{x} & & & - d_3^+ & + d_3^- & = & t_3 \\
& & & & & \mathbf{x} & \in & S \\
& & & & & \text{all } d\text{'s} & \geq & 0
\end{array}$$

The above problem is solved by using conventional LP algorithms where as many as three optimization stages may be required. In the first stage solve:

$$\min\{d_1^+\}$$

$$\begin{array}{lrrr}
\text{s.t.} & \mathbf{c}^1\mathbf{x} - d_1^+ & \leq & t_1 \\
& \mathbf{x} & \in & S \\
& d_1^+ & \geq & 0
\end{array}$$

If this problem has alternative solutions, form and solve the second stage problem:

$$\min\{d_2^+\}$$

$$\begin{array}{lrrrr}
\text{s.t.} & \mathbf{c}^1\mathbf{x} & \leq & t_1 & +d_1^{+*} \\
& \mathbf{c}^2\mathbf{x} + d_2^- & \geq & t_2 & \\
& \mathbf{x} & \in & S & \\
& d_2^+ & \geq & 0 &
\end{array}$$

where d_1^{+*} is the optimal value of d_1^+ from stage one. If the second stage problem has alternative solutions form and solve the third stage problem:

$$\min\{(d_3^+ + d_3^-)\}$$

$$\begin{array}{lrrrrr}
\text{s.t.} & \mathbf{c}^1\mathbf{x} & & \leq & t_1 & +d_1^{+*} \\
& \mathbf{c}^2\mathbf{x} & & \geq & t_2 & -d_2^{+*} \\
& \mathbf{c}^3\mathbf{x} - d_3^+ + d_3^- & & = & t_3 & \\
& \mathbf{x} & & \in & S & \\
& d_3^+, d_3^- & & \geq & 0 &
\end{array}$$

where d_2^{+*} is the optimal value of d_2^+ from stage two. Any solution to third stage lexicographically minimizes the preemptive GP.

2.2.4 Multiattribute Decision Analysis

Multiattribute decision analysis addresses problems where there are several criteria applicable over a number of decision alternatives to be selected. Each of the criteria is given a weight based on its importance relative to the problem at hand. For example, consider the choice of a new car where the colour criterion is twice important as gas mileage criterion. Multiattribute decision analysis has been applied successfully to difficult public policy problems such as locations of nuclear power plants and airports, and decisions on types of drug rehabilitation programs.

Saaty (1980) introduced the "The Analytic Hierarchy Process", a new method that uses pairwise comparisons to compute weights for the alternatives. In using the Analytic Hierarchy Process (AHP) the decision-maker starts by laying out the overall hierarchy of decision. This hierarchy reveals the criteria to be considered as well as the various discrete alternatives in the decision. For example, consider the three-level hierarchy shown in Figure 2.1 that was used to select a computer system.

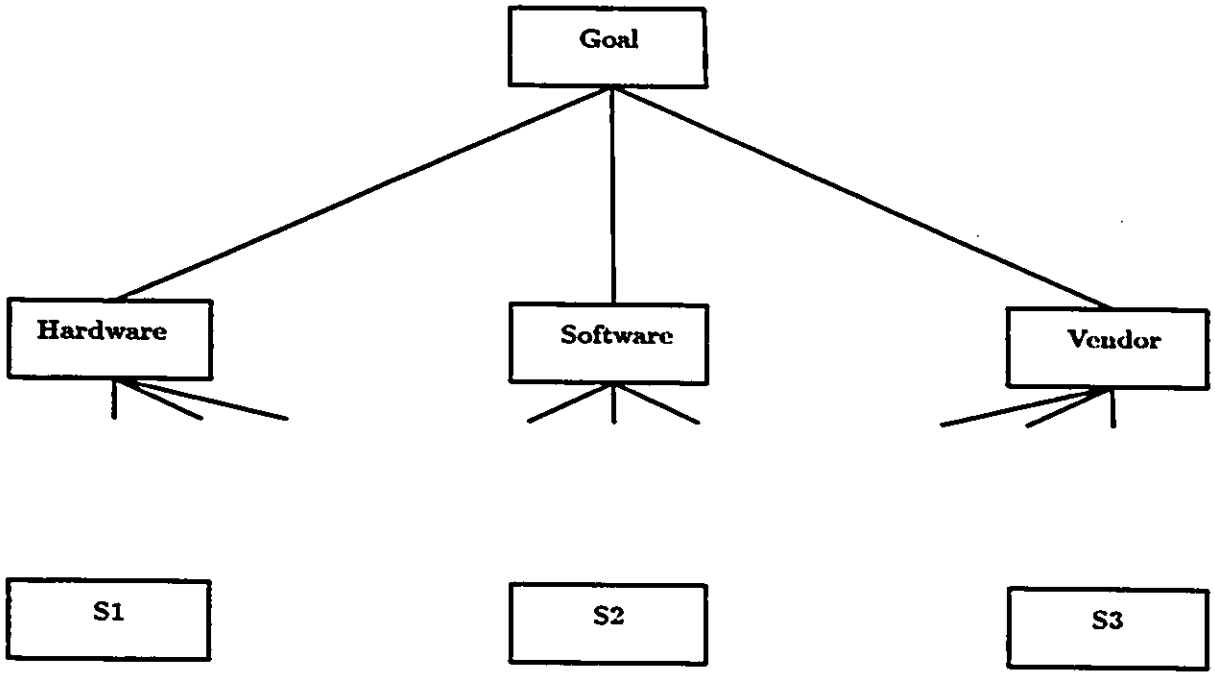

Figure 2.1. A three-level hierarchy.

The top level of the hierarchy describes the overall decision. In this example the goal is to select the best computer system. The middle level describes the criteria that are to be considered, in this example, hardware, software, and vendor support. The lower level of the decision hierarchy reveals the alternatives to be considered, in this example, three different computer systems S1, S2, and S3.

The main feature of AHP is the use of pairwise comparisons. The items in the hierarchy are compared in pairs with respect to their relative impact on a property that they both share. In the example above, S1, S2 and S3 are pairwise compared with respect to their relative impact on hardware, software and vendor, while hardware, software and vendor are pairwise compared with respect to their relative impact on the goal of the decision. This means that the three alternatives are pairwise compared three times, each time with respect to a criterion on the middle level, while the three criteria are compared only once. The result of a comparison is a value that depends on the following scale:

Value	Definition
1	Equal importance
3	Moderate importance
5	Strong importance
7	Demonstrated importance
9	Extreme importance
2,4,6,8	Intermediate values between two adjacent judgements
Reciprocals of above	If an item has one of the above numbers (e.g. 3) compared with a second item, then the second item has the reciprocal value (e.g. 1/3) when compared to the first

The values for a set of a pairwise comparisons are then entered in a matrix that has the following form:

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix}$$

where a_{ij} is the result of the comparison of the ith item to the jth item of the same level of the hierarchy. Observe that this matrix has reciprocal properties; that is:

$$a_{ji} = \frac{1}{a_{ij}} \text{ and } a_{ii} = 1$$

Suppose that the pairwise comparison matrix for the three computer system criteria is:

$$\begin{array}{c} \\ H \\ S \\ V \end{array} \begin{array}{c} \begin{array}{ccc} H & S & V \end{array} \\ \begin{pmatrix} 1 & 1/7 & 1/5 \\ 7 & 1 & 3 \\ 5 & 1/3 & 1 \end{pmatrix} \end{array}$$

where H, S, V stand for hardware, software and vendor respectively. The next step consists of computing the vector of criteria priority weights w_j by dividing the elements of each column by the sum of that column and adding the elements in each resulting row and dividing this sum by the number of elements in the row.

$$w_j = \sum_{i=1}^{n} \frac{a_{ij}}{n \sum_{k=1}^{n} a_{ik}}$$

Therefore the above matrix would yield the following weights:

$$\lambda = \begin{array}{c} H \\ S \\ V \end{array} \begin{pmatrix} 0.07 \\ 0.65 \\ 0.28 \end{pmatrix}$$

Suppose the three comparison matrices for the computer systems are:

$$\begin{array}{c} H \\ \\ S1 \\ S2 \\ S3 \end{array}\begin{array}{c} \begin{array}{ccc} S1 & S2 & S3 \end{array} \\ \begin{pmatrix} 1 & 1 & 5 \\ 1 & 1 & 5 \\ 1/5 & 1/5 & 1 \end{pmatrix} \end{array} \quad \begin{array}{c} S \\ \\ S1 \\ S2 \\ S3 \end{array}\begin{array}{c} \begin{array}{ccc} S1 & S2 & S3 \end{array} \\ \begin{pmatrix} 1 & 1/3 & 1/7 \\ 3 & 1 & 1/3 \\ 7 & 3 & 1 \end{pmatrix} \end{array} \quad \begin{array}{c} V \\ \\ S1 \\ S2 \\ S3 \end{array}\begin{array}{c} \begin{array}{ccc} S1 & S2 & S3 \end{array} \\ \begin{pmatrix} 1 & 3 & 1/5 \\ 1/3 & 1 & 1/7 \\ 5 & 7 & 1 \end{pmatrix} \end{array}$$

Using the method described above, the three matrices yield the following sytem priority weights:

$$\mathbf{C} = \begin{array}{c} \\ H \\ S \\ V \end{array}\begin{array}{c} \begin{array}{ccc} S1 & S2 & S3 \end{array} \\ \begin{pmatrix} 0.43 & 0.43 & 0.14 \\ 0.09 & 0.24 & 0.67 \\ 0.19 & 0.08 & 0.73 \end{pmatrix} \end{array}$$

To obtain the overall ranking of the computer systems, calculate the following:

$$\boldsymbol{\lambda}^T\mathbf{C} = \begin{array}{c} \begin{array}{ccc} S1 & S2 & S3 \end{array} \\ \begin{pmatrix} 0.14 & 0.21 & 0.65 \end{pmatrix} \end{array}$$

The computer system S3 received the highest final ranking and therefore is selected as the best computer system.

Saaty and Mariano (1979) applied the AHP methodology to an energy allocation problem. They described a method where optimum allocation of energy to industries in case of shortfall is made by linear programming subject to input-output constraints among the industries. AHP is used to develop an objective function that reflects the importance of each industry. Hence the objective function coefficients are the priority weights of the industries. In an example with three industries, namely P_1, P_2, P_3, the priority weights were calculated using a three-level hierarchy where the criteria are economic strength, environmental quality and national security. The pairwise comparisons yielded the following priority weights:

$$\alpha = \begin{array}{c} \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{pmatrix} 0.55 & 0.24 & 0.21 \end{pmatrix} \end{array}$$

The input-output or interdependence matrix of the industries is given by:

$$\begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \end{array}\begin{array}{c} \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{pmatrix} 1.097 & 0.227 & 0.190 \\ 0.080 & 1.065 & 0.060 \\ 0.039 & 0.332 & 1.207 \end{pmatrix} \end{array}$$

For simplicity, instead of using the industry input-output matrix to generate constraints, the coefficient in the (i,j) position of the above matrix was weighted by α_i and α_j and summed up over each row to obtain the following objective function coefficients:

$$c = \begin{array}{c} \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{pmatrix} 0.39 & 0.07 & 0.08 \end{pmatrix} \end{array}$$

The following is the energy usage for each industry:

Industry (P_j)	Energy Usage (R_i)
P_1	4,616
P_2	7,029
P_3	3,297
Total	14,942

It was assumed that the total energy has been cut back to a level of $R = 12,000$ units. Therefore the following linear programming problem was formulated :

$$\text{maximize} \quad z = c_i x_i$$

Subject to

$$x_i \leq R_i/R$$
$$x_1 + x_2 + x_3 = 1$$
$$\text{all } x\text{'s} \geq 0$$

where x_i is the number of energy units to be allocated, divided by the total number of units used by the three industries. The above linear programme yielded the following allocation:

$x_1 = 0.39$ (4,616 energy units allocated to P_1)

$x_2 = 0.34$ (4,087 energy units allocated to P_2)

$x_3 = 0.28$ (3,297 energy units allocated to P_3)

Chapter 3

Methodology

As described in the previous chapter, mathematical programming is one of the main methods in solving complex deterministic capital budgeting problems. In the methodology we first present the capital budgeting problem defined in Section 1.1 as a multiple objective mathematical programme. We then reduce the multiple objective space to a single objective function to obtain an initial solution. This solution is then improved by conducting a search for better solutions.

In this chapter, Section 3.1 presents the multiple objective formulation of the problem. Section 3.2 shows how the Analytic Hierarchy Process is used to reduce the multiple objective function to a single objective. Finally, Section 3.3 describes an interactive method in which the final solution is obtained by conducting a user-defined search within a region defined by the problem constraints.

3.1 Mathematical programming formulations

This section considers a multiple objective mathematical programming formulation for the capital budgeting problem and shows how this can be reduced to a single objective formulation. The goal is to maximize the total utility gained by the allocation of limited funds to the different projects. For example consider the following objectives:

Optimize { Revenue Generation }
Optimize { Revenue Protection}
Optimize { Savings }
Optimize { Strategic Importance }

The allocation of funds to the projects is made subject to a number of constraints which appear in the form of inequalities. The most important constraint is the capital funds constraint which represents the total availability of capital that may be allocated to all projects. Other constraints are related to the overall financial performance of the portfolio such as the maximum level of acceptable depreciation expense for each project. Constraints can also be used to represent dependencies or synergies among projects. Some projects cannot be implemented unless a percentage of another project is implemented. Finally, pairwise constraints force the allocation to a single project to be made within the levels that are defined by upper and lower allocation bounds for the project.

Assuming that the multiple objectives can be expressed algebraically, the following is the multiple objective mathematical programming formulation for the capital budgeting problem.

Formulation 3.1

$$\begin{aligned} &\max\ F_1(x_1,\ldots,x_n)\\ &\max\ F_2(x_1,\ldots,x_n)\\ &\qquad\vdots\\ &\max\ F_k(x_1,\ldots,x_n) \end{aligned}$$

subject to:

(1) Availability of Capital $\quad \sum_{i=1}^{n} x_i \leq C$

(2) Depreciation Limit $\quad \sum_{i=1}^{n} d_i x_i \leq D$

(3) No. of Employees Limit $\quad \sum_{i=1}^{n} e_i x_i \leq E$

(4) Software Expenditure Limit $\quad \sum_{i=1}^{n} s_i x_i \leq S$

(5) Dependencies and Synergies $\quad \sum_{i=1}^{n} a_{im} x_i \leq B_m \text{ for } m = 1 \ldots r$

(6) Bounds $\quad l_i \leq x_i \leq u_i \text{ for } i = 1 \ldots n$

where x_i denotes the dollar allocation assigned to project i. F_j is a function of x_i which algebraically expresses the objective j and k is the number a objectives. Inequality (1) represents the capital constraint where C is the total budget to be allocated among the projects. Inequality (2) represents the depreciation constraint where the coefficients d_i are the number of dollars depreciated per dollar allocated and D is the total amount of depreciation allowed. In a similar way, e_i represents the number of additional employees needed per dollar allocated and s_i is the software expense per dollar allocated to project i. E and S are the total number of employees and the software expense respectively. Number (5) is a set of r inequalities representing dependencies and synergies between projects. Finally, u_i and l_i represent the upper and lower fund allocation bounds for project i (Number (6)).

Each function F_j may be assumed to be linear as many financial indicators are, but they may be non-linear functions. In this formulation, the objective functions F_j have to be maximized. Their analytic expression is unknown but depend on multiple qualitative criteria important to the company's mission.

One approach to solve this problem is to use the point estimate weighted sums method thus reducing the multiple objective formulation to a single objective. Define $\lambda \in \Lambda$, where λ is a vector of λ_js and:

$$\Lambda = \left\{ \lambda \in R^k \mid \lambda_j \geq 0, \quad \sum_{j=1}^{k} \lambda_j = 1 \right\}$$

The weighted sums method reduces the multiple objective formulation to a single objective formulation with the following objective function:

$$\max \sum_{j=1}^{k} \lambda_j F_j(x_1, x_2, \ldots, x_n)$$

If the weight the functions F_j are linear in the variables x_i, then the problem formulation reduces to the following while the constraints remain the same:

Formulation 3.2

$$\max \sum_{i=1}^{n} c_i x_i$$

The c_i can be cosidered as priority weights which are dependent on the contribution that each project makes towards the quantifiable set of criteria that are important to the company's mission. The next section describes a method that provides empirical estimates for these priority weights.

3.2 The AHP Methodology

In the capital budgeting problem the Analytic Hierarchic Process (AHP) can be used to calculate priority weights for the projects which can then be used in a mathematical programming formulation to obtain an optimal allocation. The problem in using the multiple objective formulation is that the weight λ_j and the function F_j are unknown. This section shows how the coefficients c_i can be calculated assuming that the function F_j is linear in the variable x_i.

Using the AHP method, the capital budgeting decision group first has to choose a number of criteria. These criteria usually are important to the company's mission and are also relevant to the projects. The criteria are compared pairwise and a ranking is obtained. A compromise on a given comparison can be obtained either by discussion, or in the case of disagreement, by vote or by taking the geometric mean of the comparison by each member. In a similar manner, the projects are then compared pairwise for each criterion. The priority weights for the projects are obtained by taking the weighted sum of the results obtained by the two sets of pairwise comparisons. The priority weights represent the relative contribution of each project towards the company's mission. Priority weights can then be considered as a measure of the utility gained by the allocation of one unit of

resource to the projects and therefore they are estimates of the coefficients c_i discussed in the previous section.

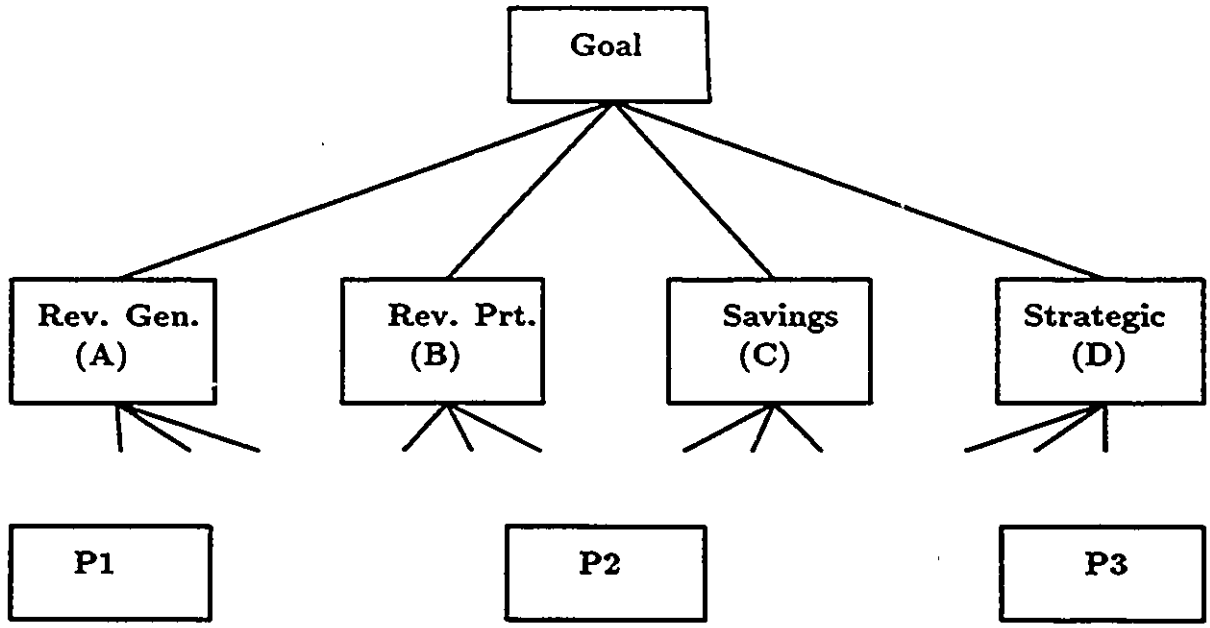

Figure 3.1 The AHP hierarchy used in the selection of telecommunications projects

For example, consider the allocation of funds to three projects, P1, P2 and P3 and suppose that the capital budgeting decision group chose the criteria, Revenue Generation, Revenue Protection, Savings and Strategic (A, B, C and D). For the sake of simplicity, we assume that all the comparisons were made unanimously in the decision-making group. The following is the comparison matrix for the criteria:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array}\!\!\begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 1 & 5 & 3 & 5 \\ 1/5 & 1 & 1/3 & 1 \\ 1/3 & 3 & 1 & 3 \\ 1/5 & 1 & 1/3 & 1 \end{pmatrix} \end{array} \text{ yielding } \lambda = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array}\!\!\begin{pmatrix} 0.55 \\ 0.10 \\ 0.25 \\ 0.10 \end{pmatrix}$$

The following are the matrices for the projects when compared under each criterion:

$$\begin{array}{c} \\ P1 \\ P2 \\ P3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} A & & \end{array} \\ \begin{pmatrix} P1 & P2 & P3 \\ 1 & 3 & 5 \\ 1/3 & 1 & 2 \\ 1/5 & 1/2 & 1 \end{pmatrix} \end{array} \quad \begin{array}{c} \\ P1 \\ P2 \\ P3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} B & & \end{array} \\ \begin{pmatrix} P1 & P2 & P3 \\ 1 & 1/2 & 1/7 \\ 2 & 1 & 1/5 \\ 7 & 5 & 1 \end{pmatrix} \end{array} \quad \begin{array}{c} \\ P1 \\ P2 \\ P3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} C & & \end{array} \\ \begin{pmatrix} P1 & P2 & P3 \\ 1 & 2 & 3 \\ 1/2 & 1 & 2 \\ 1/3 & 1/2 & 1 \end{pmatrix} \end{array}$$

$$\begin{array}{c} \\ P1 \\ P2 \\ P3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} D & & \end{array} \\ \begin{pmatrix} P1 & P2 & P3 \\ 1 & 1 & 2 \\ 1 & 1 & 2 \\ 1/2 & 1/2 & 1 \end{pmatrix} \end{array}.$$

The corresponding results for each matrix are the four columns of the following matrix:

$$\begin{pmatrix} 0.65 & 0.09 & 0.54 & 0.40 \\ 0.23 & 0.17 & 0.30 & 0.40 \\ 0.12 & 0.74 & 0.16 & 0.20 \end{pmatrix}$$

When the transpose of this matrix is multiplied with λ, this gives the final rankings :

$$\begin{matrix} P1 \\ P2 \\ P3 \end{matrix} \begin{pmatrix} 0.54 \\ 0.26 \\ 0.20 \end{pmatrix}$$

Thus, the following is the resulting objective function:

$$\text{maximize } 0.54x_1 + 0.26x_2 + 0.20x_3$$

As discussed in Section 2.2.4 , AHP is a method that has been applied successfully to many real life problems. However, one of its major disadvantages is that it does not take constrained problems into consideration. For example, consider a project which is highly dependent on others. The priority weight derived by AHP would not reflect this fact as it would overestimate the importance of the project to the company. Furthermore, sometimes the projects are not compatible with each other because of great differences in capital requirements. This can lead to overestimated priority weights of projects with higher capital requirements than the others. Another disadvantage of AHP is its dependence on human judgement. For example, a correct comparison requires each of the members of the capital decision group to have a very good knowledge of the projects.

This thesis uses AHP only to obtain an initial solution, which is then improved by the interactive method described in the next section, a method which has been developed to obtain a better solution that overcomes the problems of AHP, as mentioned above.

3.3 The interactive method approach

In the previous sections, we have seen the problem of defining and calculating priority weights for the projects that can be used as objective function coefficients. However, finding a set of priority weights that yield a satisfactory solution is not always enough. After producing the funding decision, the capital decision-making group also has to face possible disagreements from the functional group managers.

This section presents an interactive method that would help the decision-making group search for a better solution after being confronted by the group managers. At each step this method presents a set of trade-off vectors that allows the user to trade the allocation in one project against the others without violating the constraints.

Consider the following multi-objective problem:

$$\begin{aligned} \max \ & F_j(\mathbf{x}) \text{ for } j = 1,\ldots,k \\ \text{s.t } & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{x} \leq \mathbf{0} \end{aligned}$$

where $\mathbf{x} = (x_1,\ldots,x_n)^T$, $\mathbf{A}$ is a full row rank $m \times n$ matrix, $\mathbf{b}$ is an m-vector and the F_j are unknown linear functions in $\mathbf{x}$. In section 3.1 it was shown that the linear multiple objective formulation can be reduced to a single objective with objective function:

$$\max \mathbf{cx}$$

where $\mathbf{c} = (c_1,\ldots,c_n)$. Suppose that $\mathbf{c}$ is estimated using AHP and a solution $\mathbf{x}$ is obtained. Let $\mathbf{x}$ be partitioned such that $\mathbf{x} = (\mathbf{x}_N, \mathbf{x}_B)$ where $\mathbf{x}_N$ is the subvector of nonbasic variables and $\mathbf{x}_B$ is the subvector of basic variables. Let $\mathbf{A}$ be partitioned such that $\mathbf{A} = [\mathbf{N}|\mathbf{B}]$, where $\mathbf{N}$ consists of the nonbasic columns of $\mathbf{A}$ and $\mathbf{B}$ consists of the basic columns. Hence:

$$\mathbf{Ax} = \mathbf{b}$$

by partitioning $\mathbf{A}$ and $\mathbf{x}$ we have:

$$\mathbf{N}\mathbf{x}_N + \mathbf{B}\mathbf{x}_B = \mathbf{b}$$

multiplying both sides by $\mathbf{B}^{-1}$:

$$\mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N + \mathbf{I}\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$$

which is equivalent to:

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N$$

Let $\mathbf{Y} = \mathbf{B}^{-1}\mathbf{N}$. Each component y_{ij} describes the amount of decrease in the basic variables caused by an increment $\Delta \mathbf{x}_{N_j} = 1$. Hence we define $\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^n$ as the trade-off vectors to be the columns of $\mathbf{Y}$.

Step 0: Let $\mathbf{x}_B^0 = \mathbf{B}^{-1}(\mathbf{b} - \mathbf{N}\mathbf{x}_N^0)$ be the initial solution. Let the iteration counter h be set to 0.

Step 1: For every nonbasic variable , trade-off vectors are generated and presented to the user.

Step 2: For any $k \in [1,\ldots,n]$, select an amount of trade-off such that

$$\min_{h:\mathbf{y}_h^k > 0} \left\{ \frac{\mathbf{x}_{B_K}}{\mathbf{y}_h^k} \right\} \geq \Delta \mathbf{x}_{N_k} \geq \max_{h:\mathbf{y}_h^k < 0} \left\{ \frac{\mathbf{x}_{B_k}}{\mathbf{y}_h^k}; -\mathbf{x}_{N_k} \right\}$$

While $\mathbf{x}_B^h - \mathbf{y}^k > \mathbf{0}$ do

$$\mathbf{x}_B^{h+1} = \mathbf{x}_B^h - \mathbf{y}^k$$

and $h = h + 1$.

Select p such that $(\mathbf{x}_B^h - \mathbf{y}^k)_p = 0$. Calculate

$$(\mathbf{B}')^{-1} = \mathbf{B}^{-1}\mathbf{J}$$

where

$$\mathbf{J} = (\mathbf{e}_1, \ldots, \mathbf{e}_{k-1}, \upsilon_k, \mathbf{e}_{k+1}, \ldots, \mathbf{e}_n)$$

and

$$\upsilon_k = (-y_{1k}/y_{pk}, -y_{2k}/y_{pk}, \ldots, -1/y_{pk}, \ldots, -y_{m-1}/y_{pk}, -y_m/y_{pk})$$

Substitute $(\mathbf{B}')^{-1}$ for $\mathbf{B}^{-1}$ and calculate the new $\mathbf{Y}$ and $\mathbf{x}_B^h$. Go to step 1. If no trade-off is selected terminate with solution $\mathbf{x}_B^h$.

Note that the user, in repeating Step 2, can use the same trade-off or choose a new one. Furthermore, if the problem at hand has inequality constraints, the inequality constraints can be transformed to equality constraints by adding slack variables. Then the trade-off vectors can be modified so that the user has only the decision variables to consider.

Each tradeoff vector is associated with a non-basic variable. When a trade-off vector is associated with a slack variable used in a constraint (artificial variable), then the same solution is obtained either by using the trade-off or by changing the right-hand side of the constraint by one unit and then solving the modified problem using simplex. Therefore, this method can be either considered as

1. A search through the feasible region for a better solution or

2. A restriction of the resources actually used to the feasible in order to obtain a satisfactory solution.

In the next paragraphs the method is considered as in number **1.**

The main advantage of this method is that it allows the user to search for a solution without violating any of the constraints. Furthermore, $\mathbf{x}_B^h$ does not have to be an extreme point thus allowing the user to terminate with a solution within the region specified by the constraints. This method also does not require an analytic expression of the objective function and it helps the user to get a better understanding of the problem at hand by exploring the region specified by the constraints.

Z-W assumed that the utility function was pseudoconcave (Section 2.2.2), i.e. that more $\mathbf{x}$ is better, hence a maximum utility point would be extreme. In this method we relax this assumption by letting the users decide by how much they can increase a component of $\mathbf{x}$. They are free to choose any feasible direction, but the method presents only those directions agreeing with pseudoconcavity.

Chapter 4

Application

This chapter describes the development of the microcomputer-based Decision Support System for Optimal Resource Allocation (DSS ORA) that solves capital budgeting problems. DSS ORA is designed to be used by capital budgeting decision-makers as a tool that aids in decision-making. DSS ORA was developed for a major telecommunications company where the alternative capital projects are programmes that modernize a specific part of the telecommunications network. Each year, this company faces the capital budgeting problem in which each programme manager makes a recommendation to the capital budgeting decision group asking for a certain level of funding. The capital budgeting decision group, through discussions and negotiations within the group and with the programme managers, decides what level of funding each programme should received. However, there is no structured method for this process, and it has been pointed out by some managers that the decisions can at times be non-optimal and non-feasible.

Section 4.1 gives a description of the software development and the implementation of each module that constitutes DSS ORA. Section 4.2 shows how DSS ORA was used in a particular capital budgeting problem together with the results obtained.

4.1 Software Development

DSS ORA was developed for operation in Microsoft Windows environment and therefore has the standard *Graphic User Interface* (GUI) available to most Windows aplications including windows, pull down menus, dialog boxes, and icons. The software for DSS ORA was written using *Turbo Pascal for Windows* (TPW) Version 1.0, a Windows application development kit and including all the standard features of *Turbo Pascal* (TP). It also supports *Object Oriented Programming* (OOP) which has been incorporated in TP since version 5.5.

In developing DSS ORA, major emphasis was given to accommodate as many Windows GUI features as possible. This makes the tool very easy to learn and to use since it has a user interface similar to most standard Microsoft Windows applications. The user interface was written using the *Object Windows Library* (OWL) that is included in TPW and includes all the objects and methods that are needed to use the Windows GUI features in a program written in Pascal.

OWL is a collection of object definitions. Each object describes a GUI feature for example, a "window". Each object has fields and methods. The fields define all the data structures that are needed for the functioning of the feature and the methods are procedures or functions that implement the feature.

DSS ORA is divided into modules that perform the individual functions that constitute the overall program. Each module is implemented as a unit which is a small library of objects that are responsible for the functioning of the module. Each of these units uses objects and methods taken from OWL plus other external routines that were written specifically for each unit. This section will describe the implementation and the functionality of each module. Emphasis will be given to the modules that implement the methodology developed in Chapter 3.

The following is a list with brief descriptions of the key modules that constitute DSS ORA:

•**Programme Requirements** : Requests information for each of the alternative capital programmes that are to participate in the allocation. The information includes depreciation expense, software expenditure, manpower requirements and upper and lower bounds of capital required. (Module name in DSS ORA: "Data Entry").

•**Programme Importance Assessment** : Using the AHP methodology, it calculates priority weights for each programme that are later used in the initial solution module. (Module name in DSS ORA: "Value Assessment").

•**Resource Availability** : Requests information on the total capital availability and total depreciation expense, additional manpower and software expenditure allowed. Also the user can input any extra constraints. (Module name in DSS for ORA: "Problem Definition").

•**Initial Solution** : Optimally allocates funds to each program using all the information collected or calculated in the previous modules. (Module name in DSS for ORA:"Optimal Solution").

•**Sensitivity Analysis** : Provides the user the opportunity to change interactively the inputs obtained from the previous modules and to get a new allocation. (Module name in DSS ORA:"What-if Analysis").

Figure 4.1 shows the mutual relationship, function and name (in quotes) of the system modules of DSS ORA:

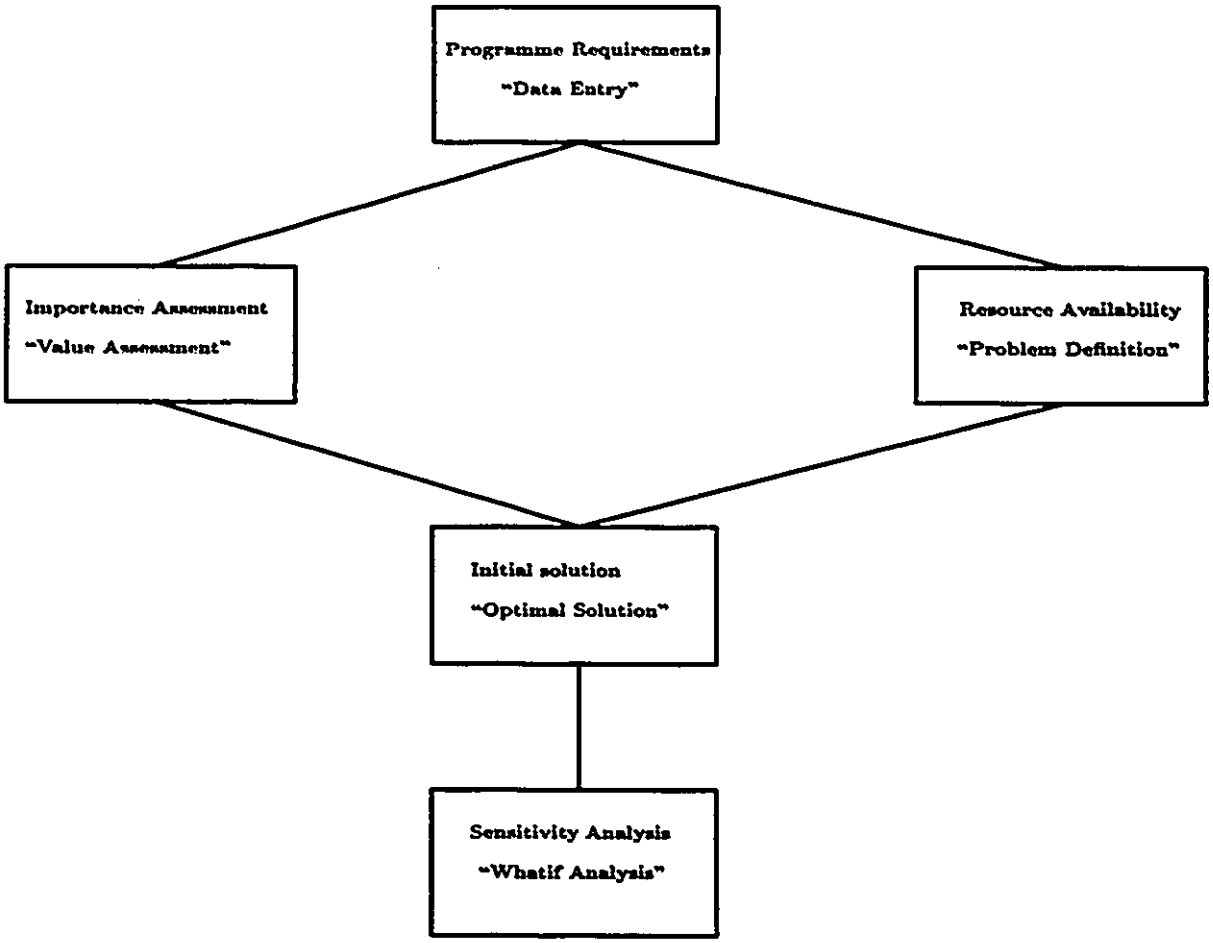

Figure 4.1 DSS ORA module diagram

The following sections provides details on each of the system modules. In Appendix C which describes DSS ORA software, Figure C.1 details the system modules, names and flow.

4.1.1 Programme Requirements

Data Entry requests information on the different programmes that participate in the allocation. For each programme the following data are required:

- Name
- Net Present Value (NPV)
- Rate of Return (ROR)
- Ranking
- Depreciation
- EQE (Equivalent employment or Manpower requirements)
- Software Expenditure
- Upper and Lower bounds of capital required

where Ranking is the user's evaluation of the programme's relative importance or value with respect to the other programmes. If these numbers are not overwritten by the programme importance assessment module then they are used as objective function coefficients by the initial solution module. Depreciation is a percentage of the capital funds given to a program that is expected to be depreciated. EQE is a measure of extra manpower needed per dollar, and software expenditure is the expected expenditure on software per dollar allocated. Finally upper and lower bound are the lower and upper limits of allocation to a programme. NPV and ROR stand for Net Present Value and Rate of Return indicating profitability of the programme. Depreciation, EQE, Software Expenditure, upper and lower bounds of capital are used as constraint coefficients in the initial solution module as in the constraint set for Formulation 3.2.

Software Implementation. The Data Entry unit (Figure B.1) uses *Multiple Document Interface* (MDI) objects that allows the opening of many data entry windows at the same time. Each programme has its own data entry window where the user enters all the above information. The user can delete a programme from the allocation by discarding its window and can add a programme by opening a new one. The current maximum number of programmes is fifteen. The data for all programmes is saved in a text file to be used later on by the other modules.

4.1.2 Programme Importance Assessment

This module implements the AHP methodology described in Section *3.3*. It calculates priority weights for each programme which substitutes the ranking numbers that were entered at the data entry module.

Criteria Comparisons. The user is asked first to input the names of the criteria. At most five criteria can be entered. Then the criteria are compared pairwise (Figure B.2), where for each pair of criteria the user has to indicate one of the following:

- Equal
- Moderately More Important
- Strongly More Important
- Very Strongly More Important
- Absolutely More Important

Programme Comparisons. When all the comparisons are made, the system presents the user with the criteria rankings. This is followed by the comparison of pairs of programmes (Figure B.3). The programmes are compared pairwise according to each criterion following the same procedure as in the criteria comparisons. At the end of the comparisons for a given criterion, the system presents the user with the programme ranking according to the current criterion together with the inconsistency ratio. If this ratio is too high, the user can go back and compare the pairs of programmes again. When the programmes are all compared according to all criteria, the system presents the user with the overall programme and criteria rankings together with the overall inconsistency ratio. Again, if the ratio is too high, the user can revisit the comparisons in order to obtain a better ratio. The programme rankings calculated in this module are priority weights that show the importance of each programme relative to the others with respect to the multiple criteria. These priority weights are used as objective function coefficients in the initial solution module.

4.1.3 Resource Availability

This module asks the user to input the following information:

- Total Capital
- Total Depreciation allowed
- Total EQE allowed
- Total Software Expenditure allowed
- Additional Constraints

The first four items in the list above are the values of the right hand sides of the capital, depreciation, EQE, and software expenditure constraints (Figure B.4). The last item is a special feature that allows the user to input interactively any additional constraints (Figure B.5). For each constraint the user is asked to input the coefficients for each programme, the direction of the inequality and the right hand side of the constraint. The user is allowed up to five additional constraints.

Software Implementation. MDI objects are used for the Additional Constraints feature where each constraint has its own data entry window.

4.1.4 Initial Solution

This module uses Formulation 3.1 (reduced to single objective function, formulation Section 3.1) to calculate an initial allocation to the programmes by using a revised simplex method algorithm.

Mathematical Implementation. The revised simplex method, otherwise known as the simplex method with multipliers, is a modification of the simplex method that significantly reduces the total number of calculations that must be performed as each iteration of the algorithm (Dantzig, 1963). The revised simplex method, rather than updating the entire tableau at each iteration, computes only those coefficients that are needed to identify the pivot element. The reduced costs must be determined so that the entering variable can be chosen. However, the variable that leaves the basis is determined by the minimum ratio rule, so that only the updated coefficients of the entering variable and the current right hand-side values are needed for this purpose. Therefore, the revised simplex method keeps track of only enough information to compute the reduced costs and the minimum-ratio rule at each iteration. The lower bounds shown in Formulation 3.1 are treated implicitly in the variable definitions while upper bounds are treated explicitly as constraints.

A bounded problem of this type can be solved by adding slack variables to the upper bound constraints and surplus variables to the lower bound constraints, thus converting them into equalities. This approach handles the bounding constraints explicitly. This module handles only the upper bound constraints explicitly but it handles the lower bound constraints implicitly thus achieving a substantial decrease in the number of constraints. Every lower bound

$$x_i \geq l_i$$

is converted to zero by defining a new variable:

$$\overline{x}_i = x_i - l_i \text{ for } i = 1 \ldots n$$

and substituting $\overline{x}_i + l_i$ for x_i everywhere in the formulation (Dantzig 1963). After solving the problem with respect to $\overline{x}_i$ the final allocation x_i to programme i is then :

$$x_i = \overline{x}_i + l_i \text{ if } \overline{x}_i \text{ is basic}$$

$$x_i = l_i \text{ otherwise}$$

Software Implementation. The simplex tableau is implemented using a Turbo Pascal object. The object performs the following main functions of the tableau:

- Keeping track of all the elements in the tableau
- Keeping track of the size of the tableau
- Performing pivot operations

The following is the object definition as it is used in DSS ORA software listing:

```
Type
  Mat = object
   rows,cols : integer;
   data : ^real;
   constructor Init(Maxrows,Maxcols : integer);
   procedure Setcell(Row,Col : integer ; item : real);
   function Getcell(Row,Col : integer) :  real;
   procedure Pivot(mrow, mcol, invar, outvar : integer);
   procedure Showit(var Tfile : text);
   destructor Done;virtual;
  end;
```

The object definition is split into two parts: the data portion and the procedural portion. The data portion defines the variables that are used by the object. These variables can only be used by this object and its descendant objects. The variables `rows` and `cols` represent the size of the tableau while the variable `data` is a pointer that points to the memory location where the tableau is stored. The procedural portion is a collection of procedures and functions called methods that define the functionality of the object. The constructor `Init` is used first to initialize the object. It allocates the memory required by the object and initializes all entries to zero. The procedure `Setcell` assigns the value item to the location defined by the variables `Row` and `Col`. The function `Getcell` returns the value that has been assigned to the location defined by the variables `Row` and `Col`. The procedure `Pivot` changes the basis by performing a pivot by introducing the non basic variable column `invar` and extracting the basic variable row `outvar`. The procedure `Showit` copies the entire tableau to the external file `Tfile` and destructor `Done` destroys the object by freeing the memory that was allocated.

The object representation for the simplex tableau uses the Object Oriented feature of encapsulation where data and code are bound together, thus allowing only a known set of methods to modify the variables defined in the data portion. Furthermore, considering that the tableau represents a polyhedron, the object can be considered as a representation of a polyhedron that has the ability to pivot itself from facet to facet.

In representing the simplex tableau as an object, better memory management is achieved. The most common way to represent a matrix is to use a two dimensional array. Turbo Pascal does not allow variable array size, therefore this causes waste of memory since the array must be sized to contain the biggest tableau occuring in the application. In this case, the size of the array would be 16 x 40. This would include 15 programmes with 4 standard constraints plus 5 additional constraints plus 30 constraints for the upper and lower bounds. Instead, the tableau is represented by an object which keeps track of the width and height of the tableau as the well as the tableau elements, thus obtaining better memory management.

4.1.5 What-if Analysis

The main purpose of What-if Analysis is to present the solution to the user and also to allow change of some of the inputs from the previous modules in order to obtain a more "desirable" allocation. Furthermore, it allows the user to trade the allocation in one programme against the others without violating the constraints.

What-if analysis displays information on priority weights, current allocation, and constraints (Figure B.6). For each programme, the module displays the following :

- Original priority weight
- Current priority weight
- A scroll bar to change the priority weight
- Original allocation
- Current allocation
- A scroll bar to change the allocation

For each constraint, the module also displays the following:

- Original right hand side value
- Current right hand side value
- A scroll bar to allow the change of the right hand side value
- Original sum of the left hand side
- Current sum of the left hand side

There are three different ways in which this module allows the user to change the allocation:

1. Change of priority weights (objective function coefficients).
2. Change of the right hand side of the constraints.

3. Change of the allocation itself.

With the first two ways, the allocation is modified by selecting the menu item "Optimize" on the screen, which accepts the new data as input to the simplex algorithm described in the previous module, and changes the allocation numbers and scroll bars to reflect the changes in the allocation. In this way, the user can do sensitivity analysis on both the priority weights and the constraint right hand side values at the same time.

Using the third method, the user can change the allocation to the programmes by simply changing the position of the cursors on the scroll bars using the mouse. Each of the cursors on the scroll bars can move until one of the constraints is violated. For example, the user can achieve a more preferred solution by decreasing the allocation in one programme and increasing the allocation in another directly. However, this solution may not be optimal with respect to the current priority weights. Consider Example 4.1, a four programme example, where the priority weights, depreciations, bounds, and current allocations are:

Example 4.1

Programme	Weight	Depr.	U.Bound	L.Bound	Allocation $(M)
A	0.34	0.2	50	10	50
B	0.31	0.3	20	10	20
C	0.27	0.1	30	10	30
D	0.08	0.2	40	10	20

Capital available : $120(M)
Maximum Depreciation : $23(M)

Suppose the user wants to increase the allocation to Programme **D**. This can be achieved by using the priority weight scrollbars to increase the priority weight of **D** from 0.08 to 0.30 and selecting the menu item "Optimize". The system normalizes the priority weights and solves the problem using a new objective function based on the newly-normalized priority weights. The following shows the new allocation :

Programme	Weight	Allocation $ (M)
A	0.28	50
B	0.25	20
C	0.22	10
D	0.25	40

Otherwise, the user using the allocation scrollbars can decrease, for example, the allocation to Programme **A** from 50 to 40 and increase the allocation to Propramme **D** from 20 to 30. In this latter approach any arbitrary allocation will be possible only within the constraints specified.

Another way that the user can change the allocation is by using the "More What if" special feature (Figure B.7). This is the implementation of the methodology developed in Section 3.3. When using this method, the system presents the user with a number of trade-offs. In Section 3.3, trade-offs were generated using the non-basic columns of the simplex tableau. However, the trade-offs presented to the user are only with respect to the main variables. Each trade-off shows by how much the allocation to each programme varies every time the user decides to change the allocation. The user changes the allocation by pressing a button on the screen. The changes are reflected by scroll bars that show the allocation levels to each programme.

As the user accesses this feature, the system presents the user with a number of trade-offs. Then the user selects a trade-off and changes the allocation by pressing the button on the screen. The user is also allowed to switch to a different trade-off at any time. All the changes are automatically recorded on the simplex tableau where the system constantly keeps track of the right hand sides. When one of the right hand side values becomes zero, then the system performs a new pivot introducing the variable associated with the trade-off in use to the basis. Then the systems presents the user with a new trade-off list. This is followed by a new selection of a trade-off by the user. This process continues at the discretion of the user.

Consider Example 4.1 in its original form. The system would suggest the following four trade-offs:

Tradeoff	A	B	C	D
1	0.0	-1.0	0.0	1.5
2	0.0	0.0	-1.0	0.5
3	-1.0	0.0	0.0	1.0
4	0.0	0.0	0.0	-5.0

Suppose that the user choses Trade-off **2**. Then at every move the allocation on Programme **C** would decrease by 1.0 and on Programme **D** would increase by 0.5. The allocation on Programmes **A** and **B** stays the same. Therefore after using the same trade-off for five times the allocation will be:

A	50.0 (M)
B	20.0 (M)
C	25.0 (M)
D	22.5 (M)

The same allocation is obtained by decreasing the upper bound of Programme **C** by 5. The reason is that trade-off **2** is associated with the slack variable of the constraint representing this upper bound which originally was a non-basic variable. Each time the trade-off is used, the slack variable is increased by one unit which has the same effect as decreasing the upper bound associated with the right-hand side of that constraint by one unit.

4.2 DSS ORA Trial†

DSS ORA was first tested on five managers responsible for the management of the programme portfolio of a major telecommunications company. Because of lack of consensus, this exercise was repeated for the second time where finally concensous was reached. A third trial was also performed, this time on a second set of managers responsible for the implemetation of programmes. The results obtained from the third trial were remarkably close to the results obtained from the second trial. In the next Sections we present the description and the results obtained from the second trial.

The purpose of the trial was to test whether DSS ORA can work within a group and also help the managers decide what to do in the event that the programmes requested for funding do not get fully funded. The department was requesting funding for five modernization programmes, namely **TEM**, **SSM**, **UDM**, **DCR** and **ACCESS** where

TEM is responsible for the modernization of transmission equipment,
SSM is responsible for the modernization of special services,
UDM is responsible for the modernization of urban distribution,
DCR is responsible for implementing traffic routing capabilities and
ACCESS is responsible for introducing fiber optics facilities.

At that time, the size of the total funding given to the department was uncertain. The next sections give a description of the trial according to each module of DSS ORA.

4.2.1 Programme Requirements

The entries for NPV, ROR, depreciation, EQE and software expenditure were irrelevant for this particular set of programmes so they were left blank. The upper and lower bounds for capital requirements for the programmes were as follows:

Programme	Lower $(M)	Upper $(M)
TEM	5	10
SSM	20	40
UDM	20	50
DCR	10	20
ACCESS	20	50

† The numbers used in the trial were set for demonstration purposes only and do not reflect reality.

4.2.2 Programme Importance Assessment

Each manager was given a questionnaire requiring the results of pairwise comparisons for each programme according to the following criteria:

- Revenue Generation
- Revenue Protection
- Savings
- Strategic Importance

The managers were required to perform five sets of pairwise comparisons, one set for the criteria and four for the programs when compared to each criterion. The results obtained from each set of comparisons made by each manager are displayed in Appendix A. The following table show the final results:

Table 4.1: Programme Importance Assessment results

Programme	Based on inputs from manager # 1	# 2	# 3	# 4	# 5	Based on Average inputs
TEM	0.079	0.045	0.037	0.061	0.066	0.063
SSM	0.322	0.170	0.466	0.351	0.251	0.311
UDM	0.135	0.067	0.111	0.078	0.158	0.111
DCR	0.239	0.467	0.169	0.125	0.086	0.206
ACCESS	0.226	0.250	0.216	0.386	0.474	0.306

The last column of Table 4.1 was obtained by taking the weighted averages of the last columns of tables A.1 to A.5. The results show significant variability which shows that the individual managers have quite different interpretations of the importance of the company criteria and how the programmes reflect these criteria. In the discussion that followed, the managers considered, as a group, the importance of each programme and accepted the averages unanimously.

4.2.3 Resource Availability

The total availability of capital was set at $120(M). Since depreciation, EQE, and software expenditure are irrelevant, their entries were left blank. No additional constraints were added.

4.2.4 Initial Solution

Figure 4.2 shows the mathematical programming formulation of the capital budgeting problem.

$$\max \quad 0.090x_{TEM} + 0.314x_{SSM} + 0.120x_{UDM} + 0.185x_{DCR} + 0.291x_{ACC}$$

subject to:

$$x_{TEM} + x_{SSM} + x_{UDM} + x_{DCR} + x_{ACC} \leq 120$$
$$5 \leq x_{TEM} \leq 10$$
$$20 \leq x_{SSM} \leq 40$$
$$20 \leq x_{UDM} \leq 50$$
$$10 \leq x_{DCR} \leq 20$$
$$20 \leq x_{ACC} \leq 50$$

Figure 4.2 The capital budgeting problem formulation.

Here x_i is the allocation in millions of dollars to programme i. The above mathematical programme yields the following allocation:

TEM	5 (M)
SSM	40 (M)
UDM	20 (M)
DCR	10 (M)
ACCESS	45 (M)

The managers pointed out that this allocation is dependent on the lower bounds. Furthermore they agreed that the allocations are independent of the value of the priority weights but are dependent on the rank of the programmes. In other words, most of the funding goes to the programmes with the highest rank.

4.2.5 What-if Analysis

What-if analysis was perfomed on different selected lower bound values. The aim was to show the effect of the lower bounds on the allocation. DSS ORA was run four times using equal lower bounds on all programmes as shown in the following table:

Programmes	1	2	3	4
TEM	0	5	10	10
SSM	0	5	10	20
UDM	0	5	10	20
DCR	0	5	10	20
ACCESS	0	5	10	20

For the 4th run, the lower bound on **TEM** was set to 10 to be consistent with the original bound set foor the programme **TEM**.

The allocations obtained are shown in the following table:

Programme	1	2	3	4
TEM	0	5	10	10
SSM	40	40	40	40
UDM	10	5	10	20
DCR	20	20	10	20
ACCESS	50	50	50	30

The allocation to **TEM** increases from 0 to 10. Notice that the allocation is at the lower bound in all four runs. The allocation to **SSM** stays constant at the upper bound in all runs while the allocation to **UDM** varies. Notice that on runs 2, 3 and 4 the allocation to **UDM** is at the lower bound. Similar variation is observed on **DCR** where the allocation is at the lower bound on runs 3 and 4. The allocation to **ACCESS** is at the upper bound in runs 1, 2 and 3 and drops to 30 on the fourth run.

Chapter 5

Discussion

This thesis addressed the capital budgeting problem faced by a major telecommunications company. In this problem, each manager presented a modernization programme to the company's central capital budgeting decision-making group responsible for funding projects over all function groups in the firm.

It was recognized in Section 1.1 that the particular capital budgeting problem to be solved had the following three problem characteristics:

1. Multiple Criteria. Since it is not possible to allocate funds to the programmes at the levels submitted by their managers, the programmes need to be prioritized using criteria that reflect the company's mission. Therefore, the decision group has to define these criteria together with the expected contribution each of these programmes would make to the criteria.

2. Multiple Constraints. The decision-making group has to consider a number of organizational and operational constraints.

3. Interactivity. After an initial allocation, legitimate problems will require the decision-making group to reconsider this allocation. Therefore, there is a need for an interactive method that will assist the decision group to reach a more preferable allocation.

Mathematical programming was used in dealing with the first problem. It was shown, in Section 3.1, that when the algebraic form of the criteria is known, the multiple criteria problem can be reduced to a single objective linear program which in turn can be solved using standard linear programming algorithms. In Section 3.2 it was shown how the AHP methodology can be used to provide single objective function coefficients when faced with multiple criteria whose algebraic form is unknown.

A solution to the capital budgeting problem can be obtained using mathematical programming and the AHP methodology. However, this solution may not be acceptable to the decision makers. In Section 3.3 it was shown how an interactive method can be used to obtain a better solution without having to deal with multiple criteria. Using trade-off questions, the decision-makers conduct an exploration of the region defined by the constraints, looking for a better solution. However, this method is limited by the number of trade-off questions (one for every non-basic variable) it can produce. Furthermore, the variety of the trade-off questions depends only on the constraint coefficients.

Chapter 4 described the development of DSS ORA: a practical software applicating the methodology developed in Chapter 3. DSS ORA formulates the capital budgeting problem at hand as a linear program and uses an implementation of AHP to obtain an approximation of the objective function coefficients. Furthermore, DSS ORA implements the interactive method developed in Section 3.3 which is used to obtain a better solution. As a decision support system DSS ORA is not designed to make decisions but rather to aid the decision-makers to make better decisions.

The DSS ORA trial proved that the system can be used in a group setting. However, the only module of DSS ORA tested in this trial was the programme importance assessment module in which it was shown how consensus can be reached by taking the average of the results obtained by each manager. The trial also showed that a good decision can only be made through the process of discussion and negotiation. The results obtained by the programme importance assessment module showed great variability between the managers. However, through discussion they accepted the average results unanimously. Therefore, tools like DSS ORA can provide a common ground upon which discussions and negotiations can be based to arrive at an acceptable compromised result.

The methodology and DSS ORA in this thesis were developed because of the lack of standard capital budgeting techniques used in the business world. The aim was to develop a system that could help the decision-makers to incorporate as much information and data possible to obtain a better and acceptable solution to the resource allocation problem. The methodology and DSS ORA have been developed for only a small portfolio of programmes. The actual capital budgeting problems that the company faces are of the size of up to 100 programmes and therefore more work is needed to be done to expand DSS ORA to deal with larger number of programmes.

A way to deal with this situation is to split the larger portfolio of as many as 100 programmes into smaller ones of up to 10 programmes which are independent from each other, and to perform capital budgeting within the smaller portfolios. DSS ORA in its current form would then be applied for the allocation of funds within each smaller portfolio. The question that arises then, is on what criteria this split would be based? and what tools should be used to aid the decision-makers in splitting this large portfolio?

One way of dealing with this problem is to group together the programmes that have dependencies and synergies between them. The programmes that show high dependencies and synergies can be combined to form a single programme thus reducing the number of programmes. These programmes would then be grouped together according to some functional criteria to form independent portfolios. The next problem would then be the allocation of funds to each portfolio. This can be done by performing capital budgeting regarding the different portfolios as alternatives competing for funds.

The development of a database that would include detailed information for each programmes can be used as a tool in splitting the big portfolio. This information would reveal the various dependencies and synergies reguired for the grouping of the alternatives. The development of the database requires research to reveal meaningful data and information that would help the decision-makers identify the dependencies and synergies between the alternatives. Research is also required for the development of a method to deal with the case of conflicting dependencies. This problem would require the ranking of the conflicting dependencies according to the company's mission, thus allowing the use of those dependencies that are important to the company.

There is also a need for better methods of prioritization. The method used by this thesis (AHP) is certainly easy and simple to use but it has certain drawbacks (Section 3.2). Development of a new method is needed to deal with group decision-making and the problem of combining the weights obtained from each member of the decision-making group into meaningful results that would help the members to reach a consensus. Saaty (1980) suggests the use of the geometric mean of the comparison results obtained from each member to form a single comparison matrix. He points out, however, that this matrix would be inconsistent.

This thesis also identifies the need for a structured, well-documented methods to specify a set of generic rules for capital budgeting decision-making within the company. These rules would form a standard code that would be followed every year by the capital budgeting decision-making body and could be built into a rule-based knowledge based expert system. A continuous evaluation of these rules would help the adaption of this method to the company's needs. These rules would also define the relations between the functional groups and the central capital budgeting decision-making body establishing an appeal-feedback mechanism between the groups and the decision-making body for obtaining more desirable allocations to the programmes.

There is also a further need for the development of computerized interactive methods with improved user interface that would help decision-makers with minimum training time to use these methods to search for better allocation of corporate resources.

Chapter 6

Bibliography

Baumol, W and Quandt, R. 1965. Investment and Discount Rates Under Capital Rationing - A Programing Approach, *Economic Journal*, 316-329.

Benayoun, R. J., De Montgolfier, J. and Latitchev, O. 1971. Linear Programming with Multiple Objective Functions: Step Method (STEM), *Mathematical Programming*, **1**, 366-375.

Bierman, H. Jr. 1988. *Implementing Capital Budgeting Techniquess*, Ballinger Publishing Company, Cambridge, Massachusetts.

Brans, J. P and Vincke, P. 1985. A Preference Ranking Organization Method: The PROMETHEE Method for MCDM, *Management Science*, **31**, 647-656.

Bromwich, M. 1979. *The Economic of Capital Budgeting*, Pitman Publishing Limited.

Charnes, A., Cooper, W., and Miller, M. H. 1959. An Application of Linear Programming to Financial Budgeting and the Cost of Funds, *The Journal of Business*, 20-46.

Charnes, A., Cooper, W. and Ferguson, R. O. 1955. Optimal Estimation of Executive Compensation by Linear Programming, *Management Science*, **14**, B423-B430.

Charnes, A. and Cooper, W. 1961, *Management Models and Industrial Applications of Linear Programming*, John Willey & Sons, New York.

Ching-Lai, H. and Abu, S. Md. M. 1979. Multiple Objective Decision Making Methods and Applications, *Lectures in Economics and Mathematical Systems*, 164.

Clark, J. J., Hinderland, T. J. and Pritchard, R. E. 1984. *Capital Budgeting*, Prentice-Hall Inc., Englewood Cliffs, New Jersey.

Crum, L. R. and Derkinderen, F. G. J. 1981. *Capital Budgeting Under Conditions of Uncertainty*, Martinus Nijhoff Publishing.

Dantzig, G. B. 1963. *Linear Programming and Extensions*, Princeton University Press, Princeton, New York.

Dean, J. 1951a. *Managerial Economics*, Prentice Hall Inc., Englewood Cliffs, New Jersey.

Dean, J. 1951b. *Capital Budgeting*, Columbia University Press, New York.

Deckro, R. F., Spahr, R. W. and Herbert, J. E. 1985. Preference Trade-offs in Capital Budgeting Decisions, *IIE Transactions*, **17**, 332-337.

Farguhar, P. H. 1977. A survey of Multiattribute Utility Theory and Applications, *Studies in Management Science*, North Holland Publications Amsterdam, **6**, 59-89.

Fogler, H, R. 1972. Overkill in Capital Budgeting Technique?, *Financial Management 1*,**1**, 92-62.

Forsyth, J. D. and Owen, D. O. 1981. Capital Rationing Methods, *Capital Budgeting Under Conditions of Uncertainty*, Martinus Nijhoff Publishing, 213-236.

Geoffrion, A. M., Dyer, J. S. and Feinberg A. 1972. An Interactive Approach for Multiple Optimization with Applications to the Operation of an Academic Department, *Management Science*, **19**, 357-368.

Ijiri, Y. 1965. *Management Goals and Accounting for Control*, Rand McNally, Chicago.

Ignizio, J. P. *Goal Programming and Extensions*, D. C. Heath, Lexington, Massachusetts.

Lee, S. M. 1972. *Goal Programming for Decision Analysis*, Auerbach Publishers, Philadelphia.

Lorie, J. and Savage, L. 1955. Three Problems in Capital Rationing, *Journals of Business*, **28**, 229-239.

MacCrimmon, K. R. 1973. *An Overview of Multiple Objective Decision Making in MCDM*, University of South Carolina Press, Columbia S.C.

Nijcamp, P. and Sprong, J. 1978. Goal Programming for Decision Making, *Ricera Operativa*, special issue on Multi Criteria Decision Making.

Pettaway, R. H. 1972. Integer Programming in Capital Budgeting: A Note on Computational Experience, *Journal of Financial and Quantitative Analysis*, 665-672.

Roy, B. 1973. How Outranking Relations Help Multicriteria Decision Making, *Multiple Criteria Decision Making*, University of South Carolina, Columbia.

Saaty, T. L. 1980. *The Analytic Hierarchy Process*, McGraw- Hill, Inc.

Saaty, T. L. and R. S. Mariano. 1979. Rationing Energy to Industries; Priorities and Input-Output Dependence, *Energy Systems and Policy.*

Salo, A. and Hamalainen, R. P. 1989. A Modelling and Decisions Aid for Supporting Telecommunications Investments, *IIEE International Conference on the Systems, Man and Cybernetics*, conference proceeding, 1, 115-18.

Schrage, L. 1986. *Linear, Integer, and Quadratic Programming with LINDO*, The Scientific Press.

Spronk, J. 1981. Interactive Multiple Goal Programming as an Aid for Capital Budgeting and Financial Planning with Multiple Goals, *Capital Budgeting Under Conditions of Uncertainty*, Martinus Nijhoff Publishings, 188-212.

Srinivasan, V. and Shocker, A. D. 1973. Linear Programming Techniques for Multidimensional Analysis of Preferences, *Psychometrica*, 38, 337-369.

Steuer, R. E. 1986. *Multiple Criteria Optimization: Theory, Computation, and Application*, John Willey & Sons Inc., New York.

Tversky, A. and Kahneman, D. 1974. Judgement Under Uncertainty: Heuritstics and Biases, *Science*, 185, 1124-1131.

Weingartner, H. M. 1963. *Mathematical Programming and the Analysis of the Capital Budgeting Problems*, Prentice-Hall, Englewood Cliffs, New Jersey.

Wilkes, F. M. 1983. *Capital Budgeting Techniques*, Willey.

Zionts, S. and Wallenius, J. 1976. An Interactive Programming Method for Solving the Multiple Criteria Problem, *Management Science*, 22, 652-663.

Zeleny, M. 1972. *Linear Multiobjective Programming*, Ph.D. Thesis, University of Rochester, New York.

Appendix A

Results from the DSS ORA trial

DSS ORA was tested on five managers responsible for the management of the programme portfolio of a major telecommunications company. The following tables present the full results of the Programme Importance Assessment module for the trial. For more information on the trial see Section 4.2.

Table A.1: Criteria Values

Criterion	Manager # 1	# 2	# 3	# 4	# 5	Average
Revenue Generation	0.55	0.07	0.38	0.17	0.56	0.34
Revenue Protection	0.09	0.07	0.38	0.17	0.10	0.16
Savings	0.28	0.58	0.18	0.17	0.25	0.29
Strategic	0.08	0.28	0.05	0.05	0.10	0.20

Table A.2: Program values according to Revenue Generation

Programme	Manager # 1	# 2	# 3	# 4	# 5	Average
TEM	0.09	0.11	0.04	0.06	0.06	0.073
SSM	0.41	0.33	0.50	0.36	0.25	0.372
UDM	0.17	0.11	0.06	0.15	0.13	0.125
DCR	0.11	0.11	0.15	0.06	0.07	0.101
ACCESS	0.21	0.33	0.25	0.36	0.48	0.329

Table A.3: Program Values according to Revenue Protection

Programme	Manager # 1	# 2	# 3	# 4	# 5	Average
TEM	0.10	0.09	0.04	0.07	0.09	0.077
SSM	0.33	0.28	0.50	0.50	0.40	0.402
UDM	0.11	0.12	0.17	0.08	0.11	0.117
DCR	0.11	0.23	0.08	0.12	0.07	0.120
ACCESS	0.36	0.28	0.22	0.23	0.33	0.284

Table A.4: Program Values according to Savings

Programme	Manager # 1	# 2	# 3	# 4	# 5	Average
TEM	0.06	0.04	0.04	0.07	0.07	0.054
SSM	0.19	0.16	0.34	0.46	0.06	0.243
UDM	0.09	0.07	0.12	0.07	0.25	0.121
DCR	0.50	0.57	0.38	0.20	0.13	0.354
ACCESS	0.17	0.16	0.13	0.20	0.48	0.228

Table A.5: Program Values according to Strategic

Programme	Manager # 1	# 2	# 3	# 4	# 5	Average
TEM	0.07	0.04	0.02	0.05	0.06	0.048
SSM	0.16	0.12	0.44	0.26	0.20	0.235
UDM	0.07	0.04	0.07	0.05	0.10	0.067
DCR	0.37	0.40	0.24	0.12	0.08	0.244
ACCESS	0.33	0.40	0.23	0.51	0.55	0.405

Table A.6: Final resutls

Programme	Based on inputs from manager # 1	# 2	# 3	# 4	# 5	Based on Average inputs
TEM	0.079	0.045	0.037	0.061	0.066	0.063
SSM	0.322	0.170	0.466	0.351	0.215	0.311
UDM	0.135	0.067	0.111	0.078	0.158	0.111
DCR	0.239	0.467	0.169	0.125	0.086	0.206
ACCESS	0.226	0.250	0.216	0.386	0.474	0.306

Appendix B

DSS ORA Screens

B.1 Programme requirements

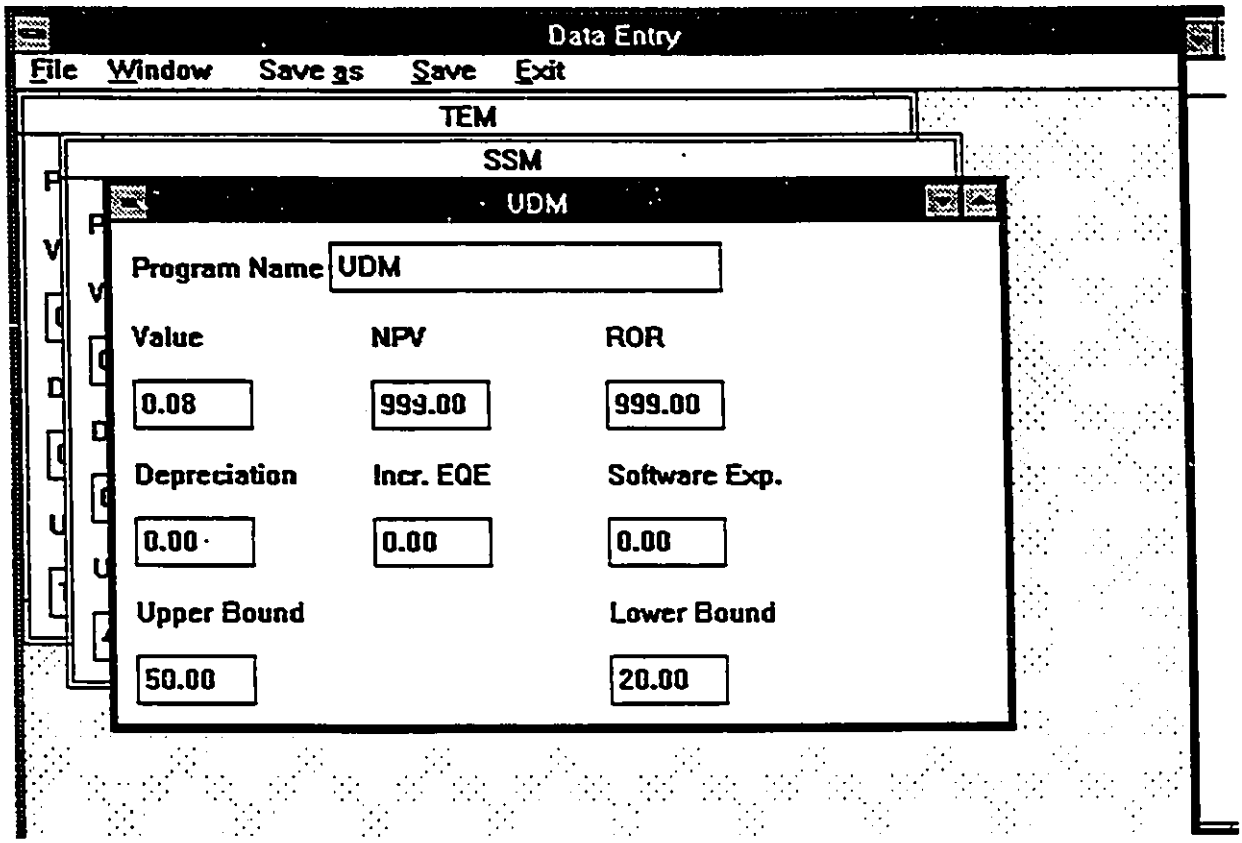

Figure B.1 The Programme Requirements Windows

Figure B.1 shows the programme requirement windows where each programme has its own data entry window. Each box is a data entry area where the user can input data using the keyboard. The bar under the title bar of the window is the menu where the user can select commands using the mouse.

B.2 Programme importance assessment

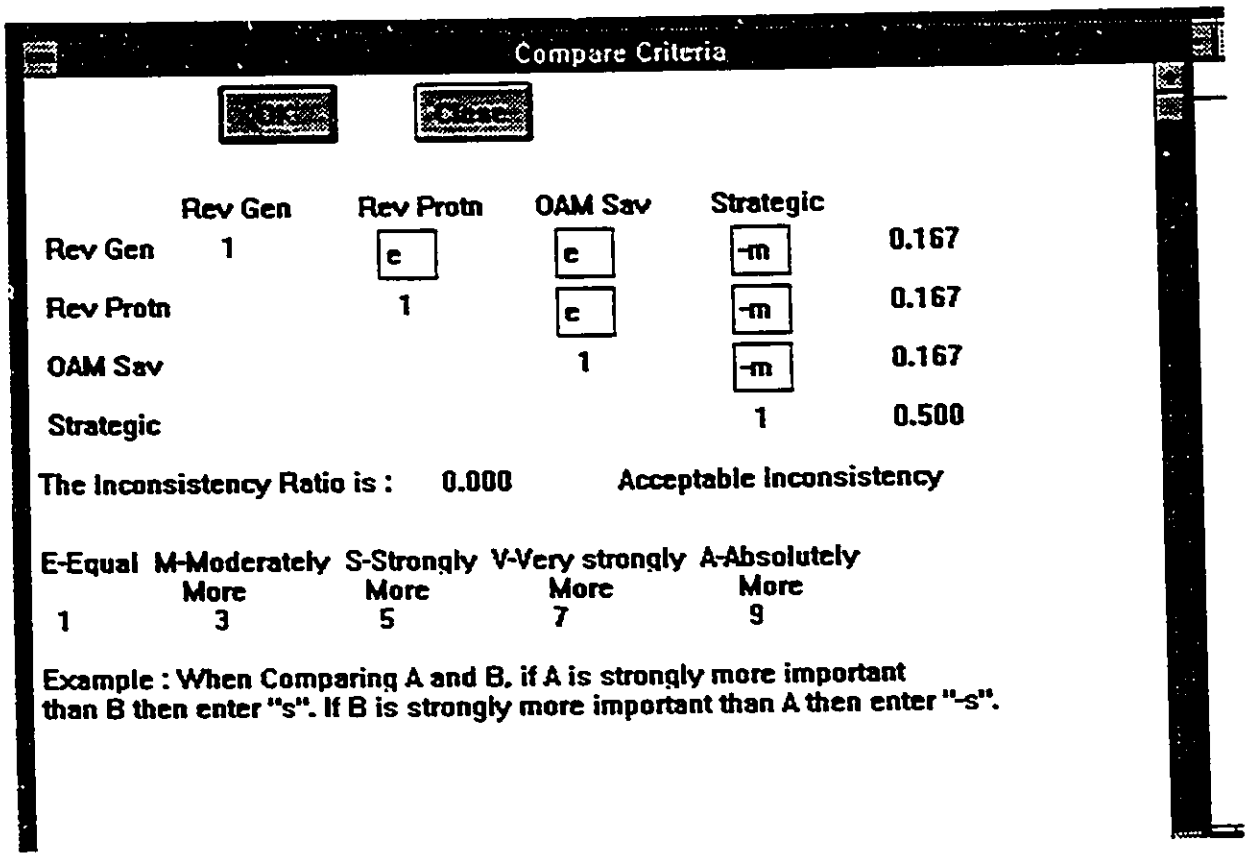

Figure B.2 Criteria Comparisons Window

Figure B.2 shows the criteria comparisons window. The top boxes with the text "OK" and "Close" represent buttons that the user can click by using the mouse. The "OK" button calculates and displays the criteria priority weights while the "Close" button closes the window and returns to the main menu. The boxes under the criteria names are data entry areas for the user to input the criteria comparison results. The numbers on the right are the calculated criteria priority weights.

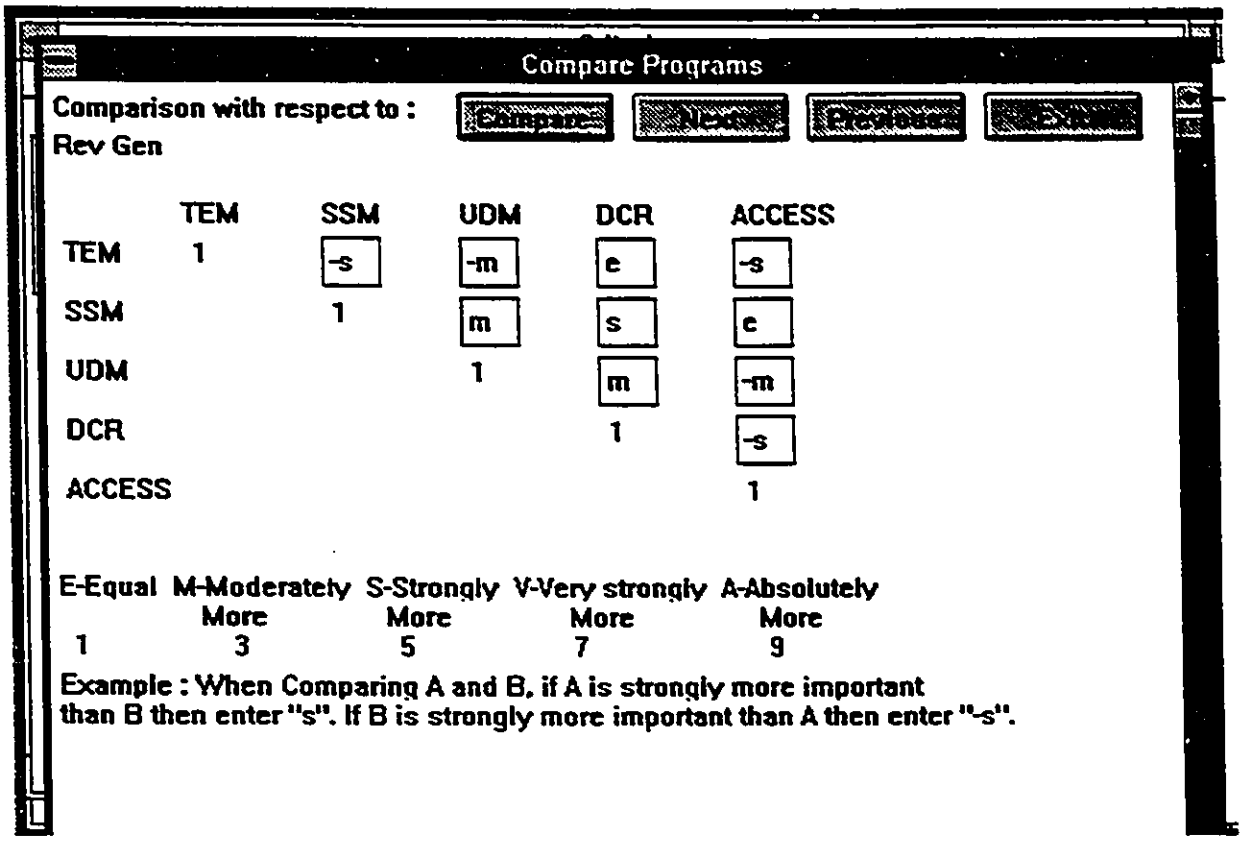

Figure B.3 Programme Comparisons Window

Figure B.3 shows the Criteria comparisons window. This particular window is used for the programme comparisons with respect to the Revenue criterion. The top boxes are again buttons. The "Compare" button calculates and displays the programme priority weights , the "Next" button opens the window for the programme comparisons with respect to the next criterion, the "Previous" button opens the window for the programme comparisons with respect to the previous criterion, and finally the "Exit" button, which is only applicable when the comparisons against all criteria are made, closes the window and returns to the main menu. The boxes under the programme names are data entry areas where the user inputs the comparison results.

B.3 Resource availability

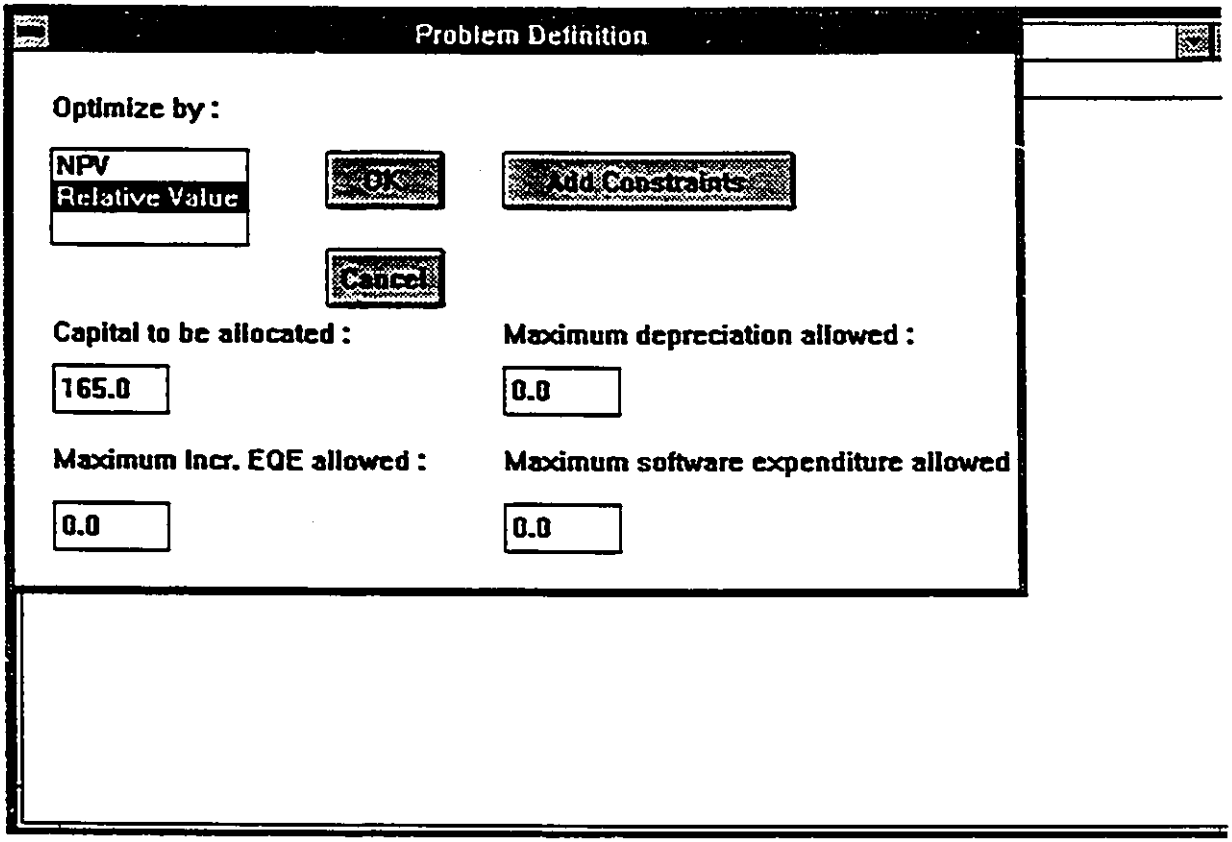

Figure B.4 Resource Availability Window

Figure B.4 shows the resource availability window. The top boxes with the texts "OK" and "Add Constraints" are again buttons. The "OK" button when selected saves the data in the data entry areas and closes the window while the "Add Constraints" button opens the Additional constraints window. The boxes with numbers are data entry areas where the user inputs the values for capital to be allocated, maximum depreciation allowed, maximum EQE allowed, and maximum software expenditure allowed respectively.

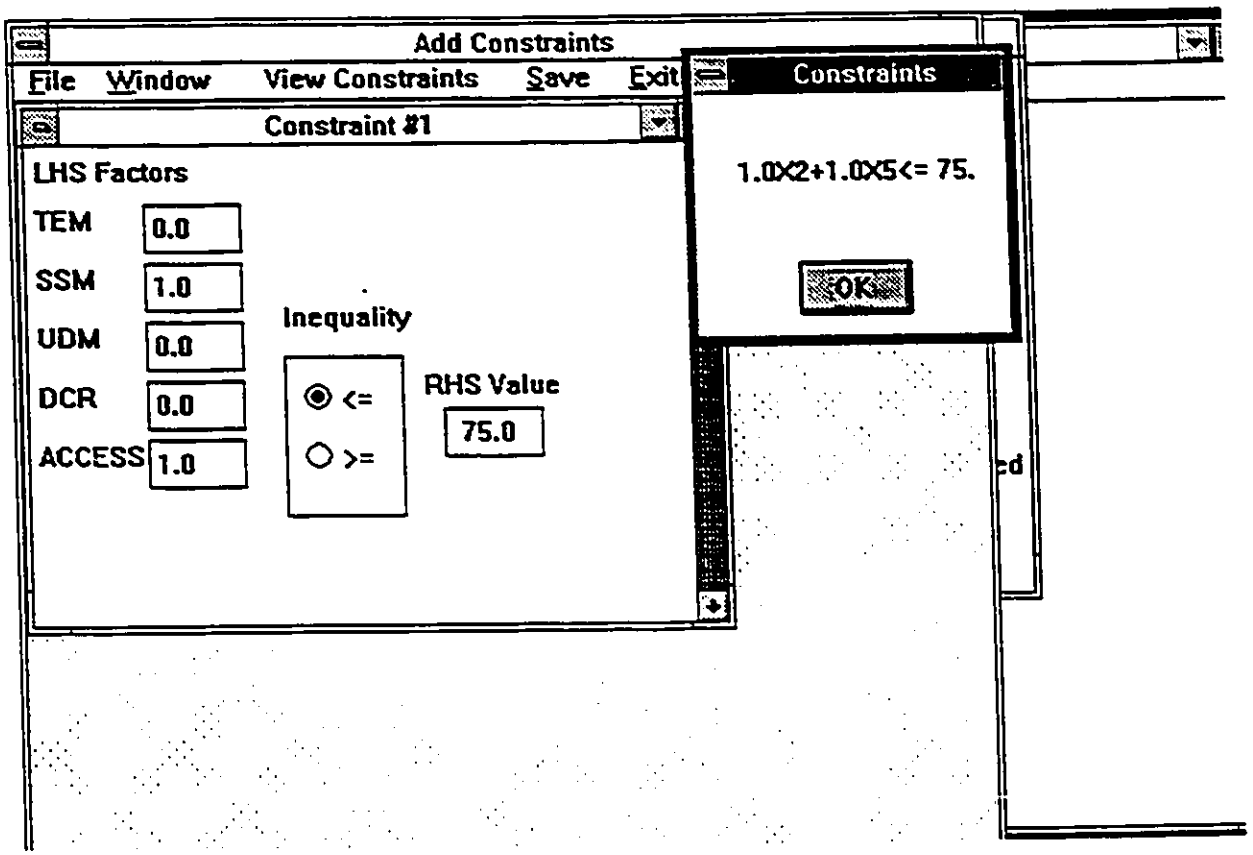

Figure B.5 Additional Constraints Window

Figure B.5 shows the additional constraints window. The boxes against each programme name are data entry areas where the user inputs the constraint coefficients. Then selects the direction of the inequality by using the mouse and clicking in the circle opposite the disarable direction. The box under the heading "RHS Value" is again a data entry area where the user inputs the value of the right hand side of the constraint. Finally the window on the right shows the constraint in algebraic form.

B.4 Whatif Analysis

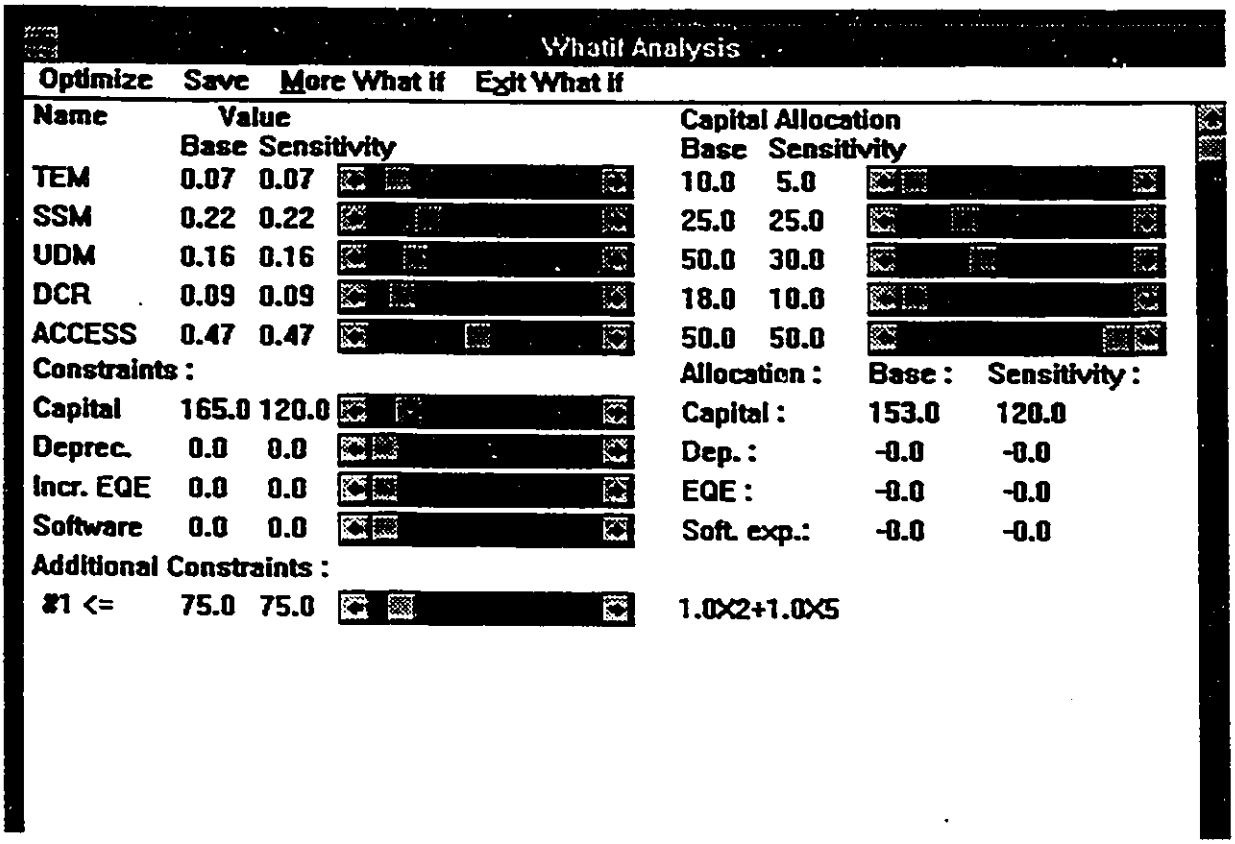

Figure B.6 Whatif Analysis Window

Figure B.6 shows the whatif analysis window. Opposite each programme name are the following: a) The initial priority weight calculated by the programme importance module, b) the current priority weight, c) a scroll bar that the user can use to change the priority weight, d) the initial allocation, e) current allocation f) a scroll bar that the user can use to change the allocation. Under the heading "Constrains" are a) the name of each constraint, b) the original right hand side value of the constraint, c) the current right hand side value of the constraint, and d) a scroll bar that the user can use to change the value of the right hand side. Under the heading "Allocation" are for each constraint the following : a) the value of the initial left hand side, and d) the current value of the left hand side.

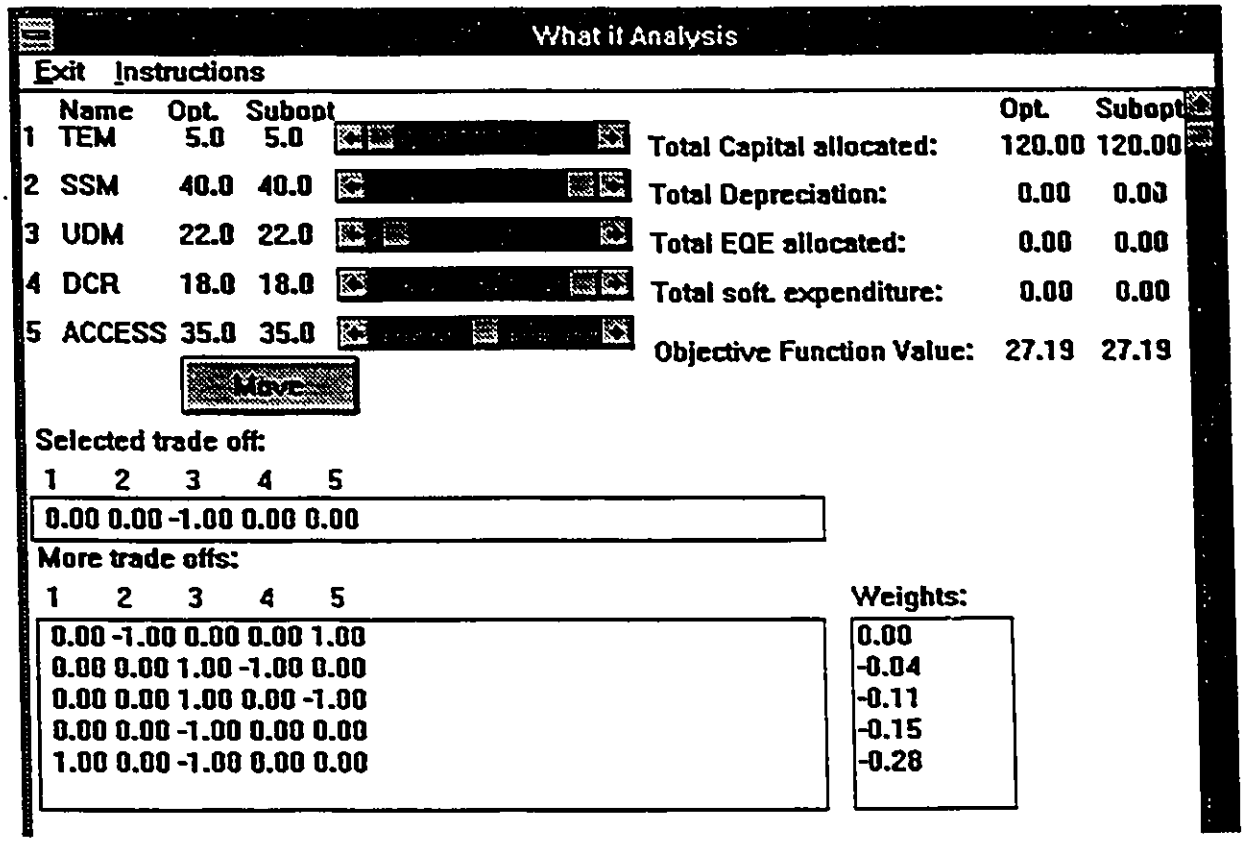

Figure B.7 More Whatif Analysis Window

Figure B.7 shows the More Whatif analysis window. Opposite each programme name are the following : a) the initial allocation, b) current allocation, and a scroll bar displaying the current allocation. On the upper left corner are the initial and current left hand side values of the constraints and the initial and current objective function value. The box with text "Move" is a button that changes the allocation according to the current trade-off which is displayed under the button. The list with the trade-offs is diplayed at the bottom together with the weights for each trade-off. The weights denote the change in the objective function at each move.

Appendix C

DSS ORA Data flow diagram

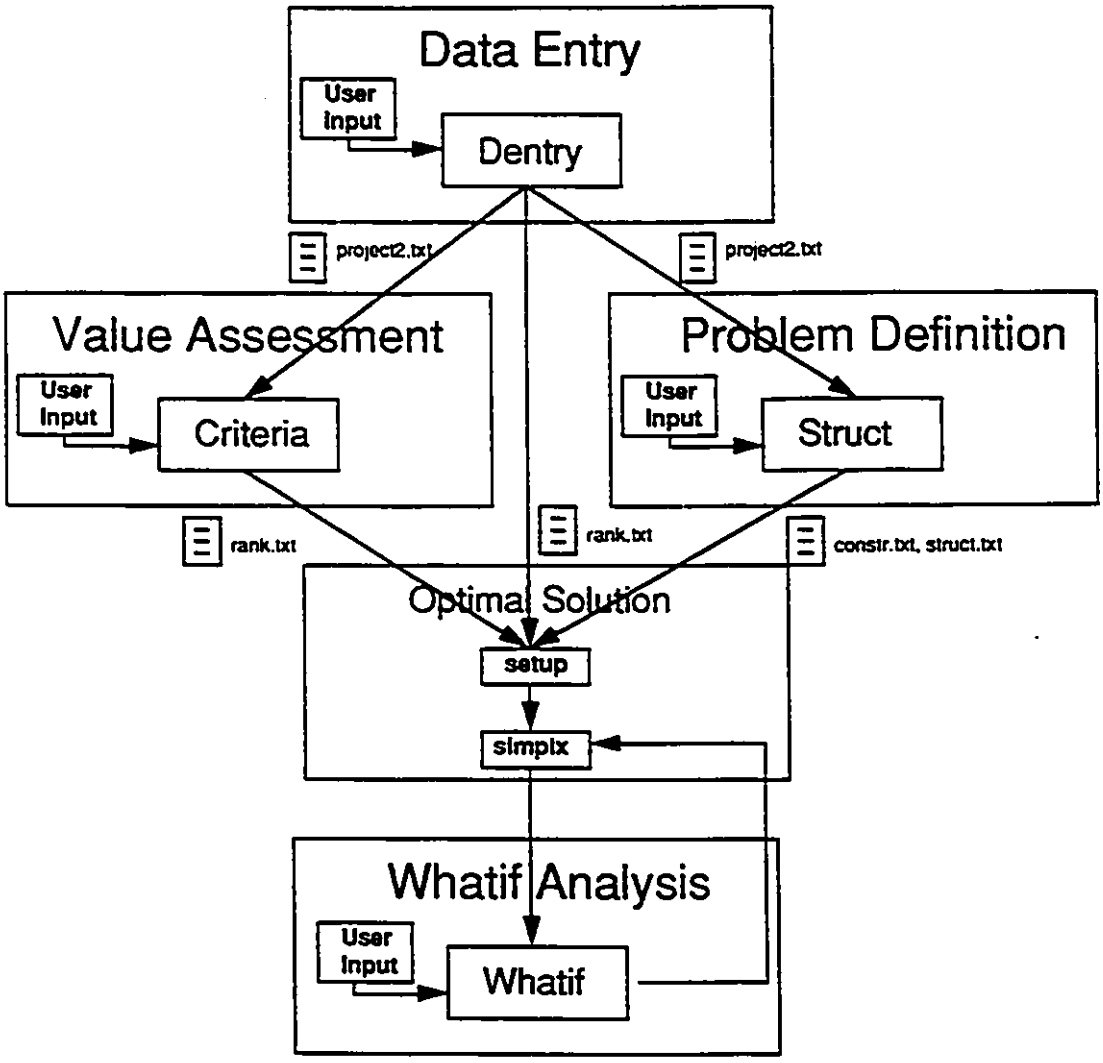

Figure C.1. DSS ORA Data flow diagram

Figure C.1 shows the complete DSS ORA data flow diagram with PASCAL module names. Each big box represents a DSS ORA module with module name as it is used in DSS ORA (Section 4.1). The smaller boxes inside the bigger boxes represent Pascal units that implement the modules represented by the bigger boxes. The arrows represent flow of variables or data files. The data files are represented by small rectangles which appear next to the arrows. The data file "project2.txt" has all the information saved from at the Data Entry module while the data file "rank.txt" has the programme priority weights. The data file "constr".txt has the coefficients for additional constraints and the data file "struct.txt" has the values of the right hand sides of the capital, depreciation, EQE and software expenditure constraint. The arrows that connect the Optimal Soution and Whatif Analsis modules represent variable flows which include object variables representing simplex tableaus and programme allocations.