Journal of International Information Management

Volume 9 | Issue 1 Article 3

2000

An artificial intelligence approach to goal programming goal structure development using AHP

Jagan Iyengar
University of Wisconsin

Mukesh Srivatsava
University of Cambridge

An artificial intelligence approach to goal programming goal structure development using AHP

Jagan Iyengar
University of Wisconsin - Whitewater

Mukesh Srivatsava
University of Cambridge

ABSTRACT

An expert system is proposed to help non-operation research users to formulate goal programs. The developmental tool being used is VP-Expert. The proposed expert system will used constraint information to assist users in goal selection. Goal structure will be constructed using a pairwise comparison technique, similar to the AHP approach.

INTRODUCTION

Over the past decade Decision Support Systems (DSS) and Expert Systems (ES) have evolved as developmental tools for decision support technology. These approaches allow management scientists to exploit real world problems and disseminate information to the decision makers in the format that is suitable to their environment (Binbasioglu & Jarke, 1986; Greenberg, 1983; Greenberg, 1987). Decision support system generators are used as modeling tools to help decision makers formulate and solve problems. For complex problem models where multiple objective functions constitute goals, and models have multiple goals, an intermediary is normally used to facilitate modeling process. This results in an elaborate process of formulating, and executing models that are indirect, and often confusing (Greenberg, 1987). Therefore, it is necessary to provide the decision maker with an automatic model building tool. The need for such tool becomes all the more significant when the decision maker is not an operation researcher, but an OR user.

Linear programming models have only one goal; linear goal programming models have multiple goals. Both models have several assumptions in common such as proportionality, divisibility, additivity, divisibility, and deterministic coefficients. The commonalities between GP and LP make studies of LP model formulation a point of reference for approaching modeling. There is an overlap in GP and LP modeling, perhaps because of this reason, the issue of GP modeling has not received significant attention by the academic community (Greenberg, 1987). Over the years GP has been accepted as an important tool for decision making by the academic community.

The purpose of this research paper is to propose development of an expert system that will allow the operation research (OR) decision maker to formulate and analyze Multiple Criteria Decision Making (MCDM) problem involving product mix. This paper is divided into three sections. The first section discusses the expert system features. The second one describes the constraint formulation process and the third section proposes methodology for goal structure development.

EXPERT SYSTEM DESIGN

The proposed expert system is designed for a non-literate OR user. Several features of the model formulation expert system help in overcoming limitation to some extent of the previous approaches. To use this expert system prior knowledge or expertise of operation research or management science is not expected. However, DM is expected to have reasonable informaiton about the problem content and context. The context information is used by the expert system to classify the problem, and content information to extract the quantitative information for the formulation of the problem. The expert system will be capable of offering various kinds of examples of constraints for consultation in non-technical terms. The user (DM) is expected to respond to the questions regarding problem constraints. Incorrectly formulated constraints will be skimmed out by the expert system. The expert system will translate constraint into simple English and present it to the DM for confirmation. The DM is expected to have the knowledge to reject unwanted constraints. The user interface is kept free of technology, and the system is capable of handling input error for most input values. DMs will have access to help module for consultation prior to making input for most input decisions.

Earlier approaches have had several shortcomings. Murphy and Stohr (1985) provide an overview of a system that uses AI techniques to formulate large linear programs. This system does not address problem specific issues. System design is not appropriate for non-OR user. The user interface design issue has not been addressed. The system does not query the user for data automatically based on prior responses. The system restricts applications to large, integer LP models.

Greenberg uses a computer-aided analysis of LP program called ANALYZE (1983). It uses several tools and mathematical techniques that aid in the understanding of LP after their formulation. The system has several limitations. Its current application addresses only transportation problems, which by nature are well structured. Also, the conceptual development for the explanations provided by the system is rather trivial and not generic enough to be adapted for other problems. The system does not address the model formulation issue or provide any help regarding it.

McDermott (1982) has developed a model management system technique called the Acturian Consulting System (ACS) based upon formulation by configuration concept. An expert system queries the user, determines the tools needed for modelling, and retrieves a library of programs. The PROLOG based ACS system automatically selects a combination of models, guided by its knowledge base and by a high-level problem specification provided by the user. Since model execution often requires data and external programs, Sivansankaran and Jarke employ a hierarchical control structure to make ACS more functional. The hierarchical structure consists of a surface level and an execution level. The surface level selects and manipulates the appropriate models, while execution level retrieves any needed data and executes model selected by the planner.

Lack of knowledge about which constraints to formulate is a serious limitation present in the modeling approaches by Murphy and Stohr (1985), Binbasioglu and Jarke (1986), and Krishnan and Lee (1988).

After formulation, an important component of modeling is its capacity to rerun the model, study the solution, and modify the original model. The model formulation process presented in this research is based upon the scientific method approach. Problem classification, model formulation, solution determination, user recommendations, problem modification and rerun will all be performed in one continuous loop. Considerable attention has been given to the user interface in the expert system. It is designed to be friendly and consistent, and serve as a user tracking device. User friendliness includes simple and appealing screen design, use of simple language, and ability to handle errors. Consistency is shown by requesting information in similar manner for all the constraints to make the user feel at ease. The expert serves as a tracking device for the decision maker by displaying instructions to keep the user informed about forthcoming screens.

CONSTRAINT FORMULATION METHODOLOGY

The expert system consists of knowledge base, user interface, and inference engine. The knowledge base contains information that enables OR users to formulate an unlimited number of constraints (subject to storage and memory limitation of the system). The inference engine takes the decision maker through different steps outlined in Figure 1. At the beginning of every consultation, the expert system queries the decision maker about his/her management science background. The response determines the extent of help extended during consultation. Also, once familiar with the system, an OR decision maker may bypass the elaborate help facility.

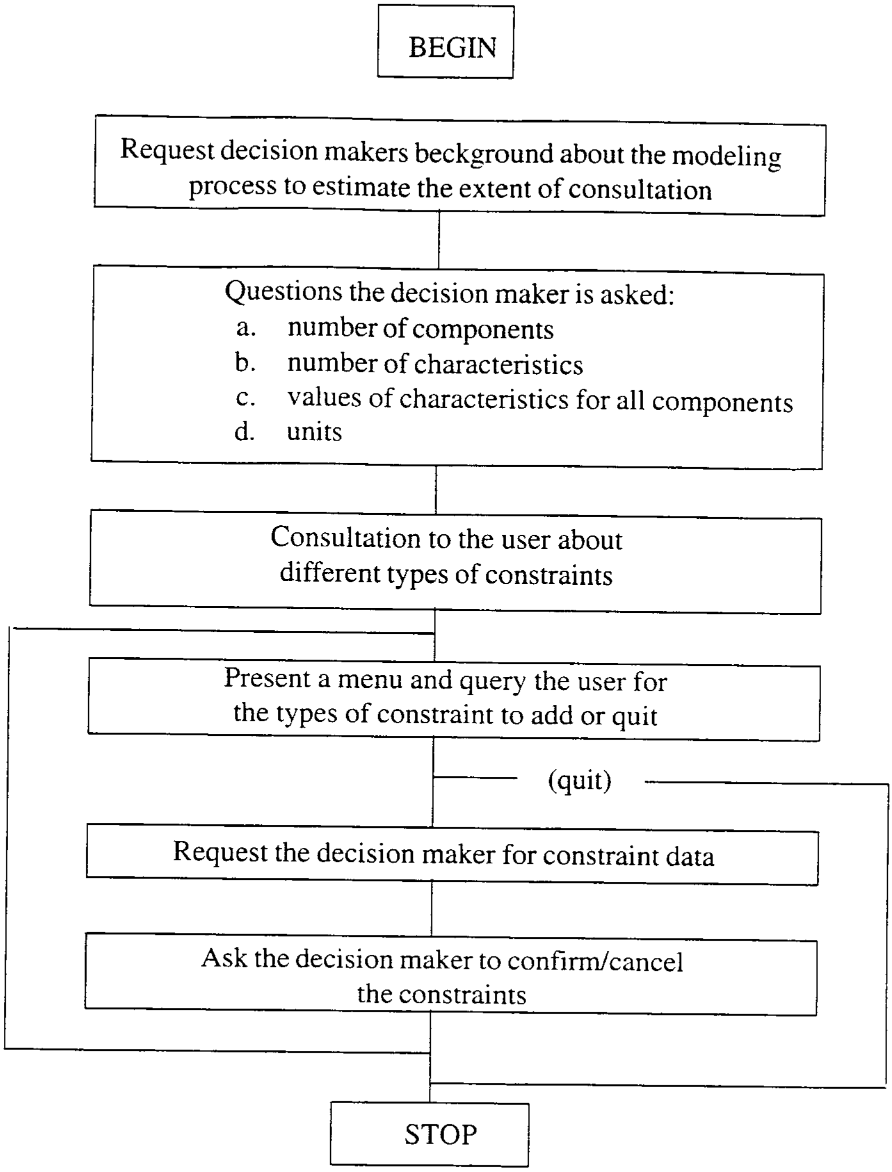
For every consultation, the expert system queries the decision maker management science background. Based upon the responses the system determines the extent of consultation needed. Once the user becomes familiar with the system, decision maker may bypass the elaborate consultation module.

The constraint formulation is designed in such a way that the decision maker is asked for some basic facts about the problem. Since most of the product mix constraints contain component names and their characteristic values, the expert system will request the following information of the user:

a) number of components
b) number of characteristics
c) names of components
d) names of characteristics
e) values of characteristics
f) numbers (units) for measuring components.

There are a number of constraints encountered in a product mix problem. Each of these constraints represent a restriction on components, number of units, mix characteristics, and their characteristics. Three types of component restructions. Component amount may be more than, less than, or equal to a certain amount.

Figure 1. The Formulation Process



Three input values are needed from a decision maker to formulate component restriction constraints:

a. Name of the component
b. Type of restriction
c. Amount of restriction

Three types of restruction: less_than_equal_to, equal_to, and more_than_equal_to are available to the decision maker in menu form. Selection is made by moving a highlighted cursor over the desired response and pressing the RETURN key. Selection changes are made by pressing the DELETE key and moving the highlighted cursor over the new response. Confirmation takes place by invoking the END key.

The decision maker selects units used to measure components during the first part of consultation. With the help of the consultation module the decision maker is able to respond to the expert system's query like "What is the maximum number or amount or quantity of product?" The decision maker could enter the number directly.

Figure 2 shows the constraint development for a component restriction.

The expert system requires the user to confirm each constraint. For constraints on component restrictions, a typical conformation screen query may state "Limit the acount of a component to less than 50 ?" Strike YES followed by the END key to confirm. Strike NO key to cancel. All cancel constraints are deleted from memory.

Figure 2. Restriction on Components

RESTRICTION OF COMPONENTS

Names to be restricted?	x4
Type of restriction	less_than_equal_to
Amount to be restricted (units)	50
Constraint generated by the expert system:	x4=< 50

There are three types of restrictions on the product mix. It can be more than, equal to, or less than a specific amount.

Four input values are needed from the decision maker to formulate the constraints:

a) Number of components
b) Names of all the components
c) Types of restriction (e.g., less_than_equal_to)
d) Amount of restriction in appropriate units.

In requesting the decision maker for the product mix amount, the expert system constructs an intelligent query based upon previous input. The decision maker enters the amount at the terminal directly.

There are three types of product mix characteristics restrictions. The product mix characteristic can be more than, less than, and equal to than a specified number. Input values are needed from the decision maker to formulate this constraint.

Figure 3. Restriction on Product Mix Amount

Number of components: 4

Names of Components: x1 x2 x3 x4

Types of restriction: Less_than_equal_to

Number of units to be restricted? 200

Constraint generated by the expert system:

x1 + x2 + x3 + x4 <= 200

The expert system requests the restriction: less_than_equal_to, equal_to, and more_than_equal_to are available to the decision maker in the form of menu. The expert system requests the restriction amount for the constraints by constructing intelligent querying (Figure 3).

Figure 4. Restriction on the Product Mix Characteristics

'Number of components: [4]

Names of component: [x1] [x2] [x3] [x4]

Number of characteristics: [3]

Names of characteristics: [c1] [c2] [c3]

Which characteristic you wish to select? [c2]

Values of c2 for all components [1.6] [2.2] [4.5] [3.6]

Types of restriction: [less_than_equal_to]

Value to be restricted at ? [2.5]

Constraints generated by the expert system:

$$\frac{1.6\ x1 + 2.2\ x2 + 4.5\ x3 + 3.6\ x4}{x1 + x2 + x3 + x4} <= 2.5$$

or

$$-0.9\ x1 - 0.3\ x2 + 2.0\ x3 + 1.1\ x4 <= 0$$

Figure 4 shows the mathematical constraint development that occurs from a typical product mix characteristics restriction. Confirmation of the constraint takes place by a procedure similar to that described earlier.

GOAL STRUCTURE DEVELOPMENT

Goal programming problems are characterized by multiple objectives or goals that often conflict. The relative importance of each is indicated by its ordinal or preemptive weight. Each objective may consist of one or more goal or subgoal within an objective as indicated by a cardinal or non-preemptive weight. The concept of goal structures is concerned with the determination of ordinal and cardinal weights. It decides the order of importance of different objectives, as well as the relative importance of subgoals within an objective.

An important concept in goal structure development is consumerability of objectives. With consumerable objectives it is possible to convert the objectives to a common measure such as profit, cost, or utility. With inconsumerable objectives it is not possible to convert all the objectives to a common measure such as profit, cost, or utility. In these cases ordinal weights or preemptive weights are used to attribute a hierarchial structure to the objectives. The objectives are then pursued in hierarchial order. The lower level objectives are completely ignored when higher level objectives are pursued. Ordering of objectives or goals into different priority levels is called preemptive or ordinal ordering. For a given goal programming problem with some system and goal constraints, objectives attempted at the expense of higher priority goals are not allowed.

Figure 5. A Goal Programming Problem

$\text{Min } z = P_1 d_1 + P_2 (3d_2n+5d_3p) + P_3d_4p$

st

$3x_1 + 4x_2 + d_1n - d_1p = 50$

$x_1 + d_2n - d_2p = 12$

$2x_1 - x_2 + d_3n\ d_3p = 4$

$x_1, x_2, d_1n, d_1p, d_2n, d_2p, d_3n, d_3p > 0$

Figure 5 shows three levels of priority -- p1, p2, and p3, with p1 being the highest. Thus minimization is attempted first. the value obtained for d_1n is included as a constraint in the goal program at p_2 and p_3 level. The minimization of $3d_2n + 5d_3n$ is the second level of priority, and is attempted next. The value is added to the constraint to the problem before pursuing the minimization of d_4p at level p_3.

The Analytical Hierarchy Process (AHP) expert system helps a decision maker rank different options for a particular task or goal in hierarchial order based upon multiple criteria.

Steps involved in the AHP:

1. A decision maker enters the names of different alternatives and criteria used to evaluate them into the program..
2. The decision maker ranks the relative importance of each criterion used to evaluate different alternatives using verbal or numeric comparisons. A numeric translation is sued by the program if verbal scale is sued. The responses are used to compute criteria weights, and an inconsistency ratio. The former sum up to one, and are used by the program in step 4 to weigh the relative weights assigned to each alternative for different criteria. The later is a measure of decision maker's internal inconsistent in the comparison ranking process on a scale of 0 to 1.
3. The decision maker makes numerical or verbal pairwise comparison of criterion values for different alternatives. If verbal comparison are used, they are converted to a numeric scale as in Step 3. The program calculates relative weights for different alternatives for each criterion. It also has the provision for deriving the relative weights for each criterion using raw data for each alternative for all criteria in conjunction with the decision maker's utility curve.
4. The decision maker initiates the synthesis procedure. The criteria weights (computed in Step 2), and the relative weights of different alternatives for each criteria are combined together to compute the priority weight for each alternative, an overall inconsistency ratio, and a sensitive utility analysis. A higher priority weight reflects a higher priority. If the number of alternatives to be examined is large, AHP system recommends the use of a rating utility scale. After Step 2, this approach queries the decision maker for rating intensities for different values of each criterion.

The goal structure program assumes that the decision maker can objectively compare any pair of goals, and rank their relative importance to one another. The working of program can be explained in three steps.

The first step: program takes the file containing all model constraints as input for every constraint. One goal is based upon minimization of negative deviational variable and other on the minimization positive deviational variable. All goals constructed are presented to the decision maker. A record of all goals selected by the decision maker, and their description are stored in files.

The second step is analogous to step 2 of the AHP approach. The program takes the output file created above, and constructs all non-repetitive pairs. For a file containing n goals, the number of such goal pairs is n(n-1)/2.

REFERENCES

Binbasioglu, M. & Jarke, M. (1986). Domain specific DSS tools for knowledge based model building. *Decisions Support Systems, 2,* 213-223.

Greenberg, H. J. (1983). A functional descriptoin of analyze: A computer assisted analysis system for linear programming models. *ACM TOMS, 9,* 18-56.

Greenberg, H. J. (1987). A natural language discource model to explain linear programming models and solutions. *Decisions Support Systems, 3,* 333-342.

Krishman, R. (1988). *PM: A knowledge based tool for model construction. Proceedings of the Twenty Second Annual Hawaii Conference on the System Science.* Kono, HI.

Lee, R. M. & Krishman, R. (1988). *Logic as an integrated modeling framework.* Decision Systems Research Institute, School of Urban and Public Affairs, Carnegie Mellan University.

McDermott, J. (1982). R1: A rule based configurer of computer systems. *Artificial Intelligency, 10*(1), 38-88.

Murphy, F. H. & Stohr, E.A. (1985). *An intelligent system for formulating linear programs.* New York University working paper series CRIS#95, GBA, pp. 40-85.