PERGAMON

Expert Systems with Applications 22 (2002) 39–50

Expert Systems with Applications

www.elsevier.com/locate/eswa

Mining linear programming models from databases using means ends analysis and artificial neural network

O.B. Kwon[a,*], K.C. Lee[b]

[a]Instituto Handong de Informacion Tecnologica, Av. Avellaneda 3071, Piso 4, CAP FED Buenos Aires, Argentina
[b]School of Business Administration, Sung Kyun Kwan University, Seoul, 110–745, South Korea

Abstract

Since formulating linear programming models from scratch is knowledge-intensive and, hence, very costly, knowledge-based formulation support systems have been proposed. The drawback of knowledge-based formulation support systems, however, is that they require that sufficient domain knowledge be captured in advance. Hence, the purpose of this paper is to propose a methodology that automatically recognizes and captures relevant knowledge on formulating linear programming models from a relational database. Our methodology has two components. First, first-cut models are recognized from a data dictionary via means-ends analysis (MEA). Second, valid first-cut models are isolated through the application of an artificial neural network technique. To demonstrate the integrity of our methodology, Model Miner, a prototype system, is described and tested. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Decision support system; Linear programming; Model formulation; Means ends analysis; General problem solver; Artificial neural network

1. Introduction

Formulating linear program (LP) models from scratch is labor-intensive and costly. Formulation efforts include having to select decision variables and find linear dependencies between the left-hand side (LHS) and the right-hand side (RHS) of the model. Discerning complex relationships between database fields remains a research challenge (Fayyad, Piatetsky-Shapiro & Smyth, 1996). Because linear programming models are applicable to this case, relying on conventional discovery techniques is not efficient. For example, a module of a linear programming model consists of a set of modelling constructs that are related to each other, either by an objective function or by a constraint. These constructs offer a methodology by which linear programming models can be exacted from a database with both domain-independent and domain-specific knowledge.

In this paper, we propose a two-phased LP model discovery method: a first-cut model discovery and a second-cut model discovery. The first-cut model discovery phase recognizes well-formed candidate modules from the data dictionary of a relational database. Then, a means-ends analysis (MEA) algorithm is applied. The second-cut model discovery phase selects a valid LP model among the candidate modules. In this phase, correct coefficient values are found by adopting a back propagation method. Model Miner, a prototype system, which is described in detail, tests the integrity of our methodology.

The remainder of this paper is as follows. In Section 2, a literature survey on model formulation and MEA is described briefly. Overall framework for the LP model discovery is illustrated in Section 3. The first part, the first-cut model discovery from data dictionary, is delineated in Section 4. Section 5 shows the second-cut model discovery and how ANN is applied. In Section 6, a prototype system is given to show the feasibility of the proposed methodology. Finally, concluding remarks and future research issues are put in Section 7.

2. Mathematical programming model formulation support systems

A mathematical programming project of this kind involves a number of steps, including gathering and organizing data, creating the mathematical programming model, optimizing that model, and reporting and analyzing the optimal solution. Once the model has been created, it can be optimized with little or no effort on the part of the analyst (Vinze, Sen & Liou, 1993).

The general approaches to the computer supported formulation of mathematical programming models involve proposing specific modelling languages and knowledge-based

* Corresponding author. Fax: +54-11-4613-4952.
E-mail address: kob@handong.edu (O.B. Kwon).

PII: S0957-4174(01)00048-3

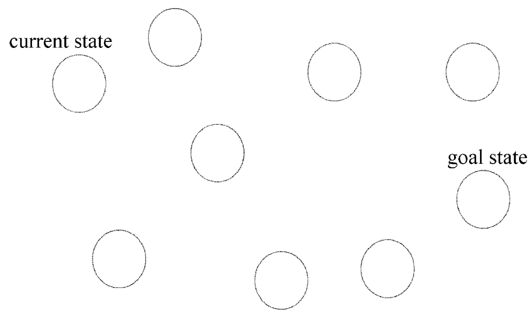

Fig. 1. MEA state space.

formulation systems. PM*, a logic-based modelling language, is used to divide knowledge into domain-specific knowledge according to logic forms. Structured modeling language (SML) is to make executable code for structured modeling (SM), a systematic way of thinking about models and their implementations, based on the idea that every model can be viewed as a collection of distinct elements, each of which has a definition that is either primitive or based on the definition of other elements in the model (Geoffrion, 1989). LINDO, LINGO, AMPL, GAMS are well-known modeling tools that have their own modelling languages.

Knowledge-based formulation systems have aimed to alleviate modeler's formulation efforts. A block-link approach has been successfully proposed (Murphy & Stohr, 1986), and the design of a software system interface, LPFORM, in graphic forms, has also been described (Ma, Murphy & Stohr, 1989). Procedural modelling knowledge treats the domain-specific knowledge as an instance of modelling knowledge (Bhargava, 1990; Bhargava & Krishinan, 1990). A prototype system in IMMPS supports the formulation, discourse, and analysis of the modelling knowledge (Greenberg, 1991), as does DATAFORM, a model database from Ketron Management Science (Ketron Management Science, 2000). Case-based reasoning techniques are also applied to model formulation supports (Liang, 1991, 1993; Vellore, Vinze & Sen, 1990). A control blackboard approach to the simulation of control features observed in the expert's model formulation protocols has been proposed (Tseng, 1997). However, they still require

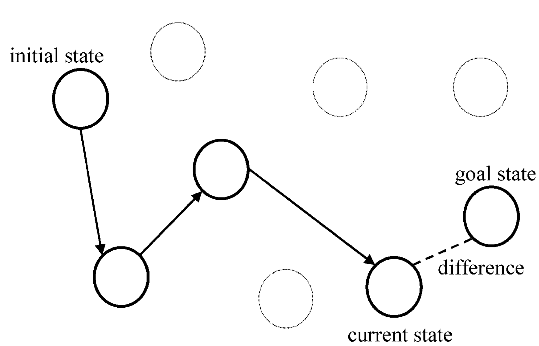

Fig. 2. Finding a path from initial state to goal state.

that sufficient domain knowledge for model formulation should be prepared beforehand.

Similarly, a variety of knowledge-based models have been constructed (Krishnan, Li & Steier, 1992; Li, Jobling & Grant, 1997). In particular, one promising object-oriented approach to model formulation implements reusable modelling constructs in large scale support systems (Becker & Bodart, 1995; Kwon & Park, 1996). Recently, knowledge-based model formulations like this one have become applicable to adjacent areas of large scale support systems such as supply chain management. LOPTIS, for example, is a database that considers raw material, production, direct shipment to markets, shipment to warehouses, tran-shipment between warehouses, warehouse shipments to markets and finished goods inventory. LOPTIS is an excellent example of how data-driven model formulations can be applicable to the supply chain.

3. Means-ends analysis

MEA, which is applied to general problem solver (GPS), is one of the problem solving methodologies in cognitive science area (Laird, Newell & Resenbloom, 1987; Newell & Simon, 1963). The search process over the problem space combines aspects of both forward and backward reasoning in that both the condition and action portions of rules are looked at when considering which rule to apply. Differences between the current and goal states may be reduced by operators. The correspondence between operators and differences may be provided as knowledge in the system or may be determined through some inspection of the operators if the operator action is penetrable. This later case, which is true of STRIPS (STanford Research Institute Problem Solver)-like operators allows task-independent correlation of differences to the operators which reduce them. STRIPS was one of the earliest robot planning programs. STRIPS was attached to a robot called SHAKEY, which had simple vision capabilities as well as tactile sensors. When knowledge is available concerning the importance of differences, the most important difference is selected first to further improve the average performance of MEA over other brute-force search strategies. However, even without the ordering of differences according to importance, MEA improves over other search heuristics (again in the average case) by focusing the problem solving on the actual differences between the current state and that of the goal.

Problem spaces, such as the type introduced by STRIPS (Fikes & Nilsson, 1971), are commonly composed of a set of goals, a state or set of states, and a set of valid operators which contain the constraints under which the operator can be applied. Fig. 1 shows that the purpose of means-ends analysis is to identify a procedure that cause a transition from the current state to the goal state, or at least to an intermediate state.

Then means-ends analysis produces a path through state