A Reduction Methodology for a Differential Diagnosis Expert System

Akram Salah
Department of Computer Science, Cairo University
Kevin D. Reilly
Departments of Computer and Information Sciences and Biostatistics and Biomathematics, University of Alabama at Birmingham

ABSTRACT

The simple production rule representation is generalized by adding programs to a management system that manipulate rules in a rule-based system. By adapting this methodology, a single generalized rule can represent a group of simple ones. Then programs are employed to satisfy the general rule in a partial way while recursively reducing a decision problem into smaller ones of the same nature until a decision is made. It is shown that the reduction method is more efficient than the simple rule approach and that it minimizes the number of rules used to express a problem. The concept of using a management program to manipulate a set of rules is emphasized through solving a problem in a differential diagnosis expert system. A comparison between the number of rules employed to express a problem is made to show advantages of the reduction methodology over the simple rule representation.

KEYWORDS: ***production systems, expert systems, reduction algorithm, prolog, decision tables, relational databases, medical diagnosis***

INTRODUCTION

Much recent research focuses on computer systems that facilitate methodologies simulating experts' knowledge-based decision-making strategies (see, for example, [1–4]). The most popular current way to create such systems is to incorporate large amounts of "domain-dependent" knowledge acquired from experts. Because experts often express their decision-making processes in sets of

Address correspondence to Kevin D. Reilly, Departments of Computer and Information Sciences and Biostatistics and Biomathematics, University of Alabama at Birmingham, Birmingham, Alabama 35294.

International Journal of Approximate Reasoning 1987; 1:131–139

52 Vanderbilt Ave., New York, NY 10017 0888-613X/87/$3.50

if-then rules, rule bases are devised (Hayes-Roth [5]). Such systems are usually called knowledge-intensive rule-based systems.

A knowledge-intensive rule-based system consists mainly of a set of rules describing different decision situations in the problem under question, together with actions to be taken in each case. A rule in such a system is represented as a structure typically in the form

$$A_1 \ \& \ A_2 \ \& \ \ldots \ \& \ A_n \rightarrow \mathrm{C}$$

Such a rule is interpreted as follows: if A_1 and A_2 and . . . and A_n are true simultaneously, then consequently C is true. The left side of a rule contains a conjunction of atoms called *conditions* and the right side is called a *conclusion* or a *consequence* (Salah and Yang [4]).

If an expert expresses his or her decision-making process as a collection of simple if-then rules, each of them can be represented directly as stated above. A problem arises if an expert expresses rules in a less explicit way or in some form such as a function over a set of rules. Then it is the responsibility of the rule acquirer, whether person or machine, to decide upon a representation or to provide some kind of control on processing such rules.

In this article we show an approach that can facilitate a generalization of simple rule representation. This article is part of a larger study that has borrowed concepts from relational database systems (RDBSs), such that rules are stored in a rule base and then a management system retrieves the rule under question. The system exploits a number of features studied previously (Reilly et al. [6], Yang [7], Reilly et al. [8]), where key concerns have been incorporation into a Prolog framework (Kowalski [9], Clocksin and Mellish [10]) of knowledge representations and RDBSs (Bruynooghe [11]). It is shown that the approach increases the efficiency of a rule-based system.

THE PROBLEM

The problem arises in a differential diagnosis expert system where conditions are either symptoms, observations, or test results gathered by a physician to be used to derive a conclusion, which in this case is a disease or a class of diseases. Rules used to derive such a conclusion would typically be in the form

$$O_1 \ \& \ O_2 \ \& \ \ldots \ \& \ O_n \rightarrow \mathrm{D}$$

where each O_i for $1 \leq i \leq n$ is an observation, and D is a disease or class of diseases. (From here on, we refer to any atom on the left side of a rule as an observation (an observation can be a test result or a symptom) and on the right side as a disease. Thus, this rule is read as follows: if all observations 1 through n exist simultaneously in a patient, then this case can be diagnosed as D.)

Our problem arises when a group of rules is expressed within a single if-then

statement. Our particular concern is this: given a set of n observations and a conclusion D such that if any k-subset of observations of n holds, $k \leq n$, then D can be concluded. That is,

$$\text{any } k \text{ observations of } \{O_1 \,\&\, O_2 \,\&\, \ldots \,\&\, O_n\} \rightarrow \mathrm{D}$$

A simple example of such a case is the common cold, where there are about 12 observations and any 3 of them (existing simultaneously) establish the diagnosis. Examples of a similar nature occur in rheumatic diseases (more discussion is provided below).

Actually, this is a generalization of a rule application. The special case in which $k = n$ defines the "normal" rule structure, that is, the case in which all the conditions have to be satisfied to derive the consequence. A more formal view of this problem is as follows. In a rule system there is a set of conditions for each decision situation, each condition having a domain of values. The left side of any rule represents an element in the Cartesian product of the domains of these conditions. The case here may be conceptualized as having one condition with one domain of observations, say, O with length n, such that if any k-subset of O with $k < n$ occurs simultaneously, then the diagnosis is established. This expresses a set of rules, each one having a condition part $c \in O^k$, where $k < n$, and the same consequence D, which is the disease under consideration.

To represent this situation within an expert system, we examine two alternatives to set the stage for subsequent comparison.

1. The single if-then statement is re-expressed as a set of simple rules. Each such simple rule contains k observations on its left side and D on its right side. Needless to say, the resulting number of rules consumes much memory space, complicates the search when the system is applied, and reduces the efficiency of the system.
2. The production system is extended such that if the "any k out of n" formulation is expressed, it can be handled automatically.

Note that this problem differs from those representations of "inexact reasoning" or uncertainty (Prade [12], Rosenbloom et al. [13]) in which subsets of conditions are used to derive a consequence—for instance, probabilistically, fuzzily, or using weighting schemes.

REDUCTION METHOD

The reduction method is based on viewing rule-base systems as a set of rules together with programs that manage such rules. This view enables us to add program code to the management system such that it can extend the simple representation of production rules. Here, we apply this methodology to enable a direct representation of the generalized form discussed above.

We denote a problem as "a k/n diagnostic problem" when the diagnosis is

dependent on n observations such that if any k of them are found to exist in a case, then the diagnosis is established. For example, the case discussed by Weiss and Kulikowski ([14], p. 119 ff.) in diagnosis of rheumatic diseases such as mixed connective tissue disease (MCTD) involves 10 observations. If any 4 of these 10 exist in a patient, then he or she definitely has a rheumatic disease. Using our terminology, we say that this is a 4/10 diagnostic problem.

A Reduction Algorithm

To solve a k/n diagnostic problem using the reduction methodology, we perform the following:

1. Pick any k symptoms.
2. Name them temporarily $T1, \ldots, Tk$.
3. Check decision table (k) with results for the k symptoms.
4. The output of the decision table is δ.
5. The problem now is δ/R, where $R = n - k$.
6. For any δ/R diagnostic problem:
   (a) if $\delta = 0$ diagnosis is POSITIVE; terminate.
   (b) if $\delta > R$ diagnosis is NEGATIVE; terminate.
   (c) if $\delta \leq R$ go to step 1 (with $k = \delta$, $n = R$) for further reduction.

The set of tables in Table 1 depicts the situation in a simplified form to make it easier to focus on the steps of the reduction method. In realistic cases, actions may involve reports back to the user on the rules that are fired, auxiliary calculations (for instance, of a statistical nature), or other options. In such cases, a table action portion would include additional information along with the number of remaining tests that are depicted in this set of tables. It should be noted that the use of tables to describe the algorithm does not necessarily imply that implementation by tables is mandated. If tables are used in the implementation, they need not always be stored; that is, there are cases in which they can be generated.

An Example

To illustrate the method, we use a specific example of a 4/7 diagnostic problem. To solve the diagnostic problem:

1. Pick any 4 observations.
2. Name them temporarily $T1$, $T2$, $T3$, and $T4$.
3. Check the first table in Table 1 with the results of these 4 observations:

   (P = positive or N = negative)

4. The possible cases are as follows:
   (a) If the results of all 4 are P, then the diagnosis is definitely established.
   (b) If only 3 are P, then we need to check 1 more of the remaining 3.

Table 1. Decision Tables Used for $4/n$, $3/n$, $2/n$, and $1/n$ Problems

$T1$	P	P	P	P	P	P	P	P	N	N	N	N	N	N	N	N
$T2$	P	P	P	P	N	N	N	N	P	P	P	P	N	N	N	N
$T3$	P	P	N	N	P	P	N	N	P	P	N	N	P	P	N	N
$T4$	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
δ	0	1	1	2	1	2	2	3	1	2	2	3	2	3	3	4

For any $4/n$ problem

$T1$	P	P	P	P	N	N	N	N
$T2$	P	P	N	N	P	P	N	N
$T3$	P	N	P	N	P	N	P	N
δ	0	1	1	2	1	2	2	3

For any $3/n$ problem

$T1$	P	P	N	N
$T2$	P	N	P	N
δ	0	1	1	2

For any $2/n$ problem

$T1$	P	N
δ	0	1

For any $1/n$ problem

(c) If only 2 are P, then we need to check 2 more of the remaining 3.
(d) If only 1 is P, then we need to check 3 more of the remaining 3.
(e) If all are N, then hypothetically we need to check 4 of the remaining 3, which is impossible; thus, we reject the diagnosis.

In case (a) there are 4 observations; all of them hold, and the diagnosis is established (further checks are 0 of 3). In cases (b), (c), or (d), a diagnosis is not

established because there is insufficient input information. Instead of restarting the problem, we can define a new *reduced* problem such that we check only the remaining observations. Now we need to check either 1/3 in case (b), 2/3 in case (c), or 3/3 in case (d). In case (e) we can reject the diagnosis because the total number of observations is 7, 4 of them have already been checked and failed, and the remaining observations are 3 in number. To establish a diagnosis, 4 observations need to exist; therefore, it is impossible to establish a diagnosis from this situation (4/3).

Thus, the method either establishes a diagnosis from the information provided or uses the information to reduce the problem to a smaller problem of the same nature. The new problem can be solved recursively by the same methodology.

Commentary

We can cite several advantages of the reduction methodology: (1) there is guaranteed recursive reduction until a solution is reached; (2) the number of rules to be checked is less than using the simple rule approach (see Table 2); (3) the tables given in Table 1 can be used for any diagnostic problem k/n, regardless of the value for n; and (4) the growth of the number of rules is limited, as all the decision tables are complete (Welland [15]), and thus there is no possibility of adding rules to any of them.

As can be seen, the number of rules in the reduction method depends on the length of the subset that establishes the diagnosis, k, rather than the length of the domain of observations, n. This is an important property of the reduction methodology, as in simple rule approaches the number of rules grows exponentially with the length of the set of observations, assuming that simple rule generation uses either combinations or permutations.

According to Weiss and Kulikowski ([14], pp. 118–119), a problem similar to what we have been intimating was detected while an expert system for diagnosis for rheumatic diseases was being implemented. As the expert system evolved, the number of its (physician) users increased; consequently, the number of observations known to the system increased. The expert system started with a 4/10 diagnostic problem and was extended to 4/18 and eventually to 4/35. A

Table 2. Number of Rules Used to Build a Knowledge Base

Problem ID	Using Permutation	Using Combination	Using Reduction
4/10	5020	210	31
4/18	73440	3060	31
4/35	1256640	52360	31

simple treatment of the problem (see Table 2) would make the number of rules increase exponentially with any increase in the number of observations.

A final point to be noted about the reduction method is that it does not depend on any particular application. The algorithm was developed for an expert system for differential diagnosis of rheumatic diseases, but it can be used in any other rule representation of the same nature.

ENVIRONMENT

The system that we employ for representing the methodology of this article is based on an extension of a previously defined system called EXPRD (EXtended Prolog Rule Data system), an integration of a Prolog, a relational database, and a decision table system (Salah [16]). This system is used to store or generate decision tables such as those appearing here. Prolog programs expressing the reduction algorithm are added as a part of the management program. Sets of observations are stored in the system as database relations. An interactive dialogue prompts the user to provide the proper information for the diagnostic problem and invokes the reduction algorithm. If a decision is reached, the program advises the user whether the diagnosis is established or rejected. If the information is not sufficient to establish the diagnosis, the program prompts the user to provide more information. An example dialogue in a session is as follows:

Give me a test you performed: Arthralgia

. . .

What is the result of arthralgia (p = positive, n = negative): p

. . .

Diagnosis is POSITIVE

Chronic polyarthritis > 6 weeks is a significant factor

. . .

What is the result of synovial fluid inflammatory (p = positive, n = negative): p

. . .

These results are not sufficient to establish a diagnosis

. . .

What is the result of subcutaneous nodules (p = positive, n = negative): p

. . .

Diagnosis is NEGATIVE

Two positive symptoms are noted

. . .

Do you wish a trace of this dialog? No

. . .

A general philosophy in dealing with rule systems emerges from our methodology: a management system is employed in which rules are dealt with as one of the components. Such a management system can be viewed as a meta-rule program that helps a user (or an expert-system administrator) to build, manipulate, query, and analyze a rule system.

CONCLUSION

Although the reduction methodology for differential diagnosis expert systems is self-contained in the sense that it solves a well-defined problem, if we take a broader view of the situation, we see this method as part of the overall rule-management environment. The environment conceptualization emphasizes use of meta-level processing to manipulate rule-like representations. Given a *k/n* diagnostic problem, an extended form of rule, the management program is designed to generate a set of simple rules or employ the reduction methodology to reduce the problem to a smaller problem of the same nature. Employing management programs on the meta-level facilitates a global view for expert systems, allowing operations such as generation, reduction, or analysis of rules.

References

1. CODASYL Decision Table Task Group, *A Modern Appraisal of Decision Tables,* Association for Computing Machinery, New York, 1982.

2. Dahl, V., Logic programming as a representation of knowledge, *IEEE Computer,* 16(10), 106–111, 1983.

3. Fikes, R., and Kehler, T., The role of frame-based representation in reasoning, *Comm. of the Assn. for Computing Machinery* 28, 904–920, 1985.

4. Salah, A., and Yang, C. C., Rule-based systems: A set-theoretic approach, *Proceedings of the 3rd Annual Computer Science Symposium on Knowledge-Based Systems: Theory and Application,* Columbia, SC, 1986.

5. Hayes-Roth, F., Rule-based systems, *Comm. of the Assn. for Computing Machinery* 28, 921–932, 1985.

6. Reilly, K., Salah, A., Morgan, P., and Rowe, P., Multiple representations in a language-driven memory model, in *Papers on Computational and Cognitive Science* (E. Battistella, Ed.), Indiana Univ. Linguistic Club, Bloomington, Ind., 87–94, 1984.

7. Yang, C. C., *Relational Databases,* Prentice-Hall, Englewood Cliffs, N.J., 1986.

8. Reilly, K. D., Salah, A., and Yang, C. C., *A Logic Programming Perspective on Decision Table Theory and Practice,* University of Alabama at Birmingham Tech. Report, 1986.

9. Kowalski, R., *Logic for Problem Solving,* Elsevier-North Holland, New York, 1979.

10. Clocksin, W., and Mellish, C., *Programming in Prolog,* Springer-Verlag, New York, 1981.

11. Bruynooghe, M., *Prolog in C for Unix Version 7: A Reference Manual,* Katholieke Univ., Leuven, Belgium, 1980.

12. Prade, H., A computational approach to approximate and plausible reasoning with applications to expert systems, *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-7, 260–283, 1985.

13. Rosenbloom, P., Laird, J., McDermott, J., Newell, A., and Orciuch, E., R1-Soar: An experiment in knowledge-intensive programming in a problem-solving architecture, *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-7, 561–569, 1985.

14. Weiss, S., and Kulikowski, C., *A Practical Guide to Designing Expert Systems,* Rowman & Allanheld, Philadelphia, 1984.

15. Welland, R., *Decision Tables and Computer Programming,* Heyden & Son, London, 1981.

16. Salah, A., *An Integration of Decision Tables and a Relational Database System into a Prolog Environment,* PhD Thesis, Univ. of Alabama at Birmingham, Birmingham, Ala., 1986.