MCC Technical Report Number AI-445-86

USING A TMS FOR EBG

Shiuh-li Liuh and Michael N. Huhns

December 31, 1986

Non-Confidential

Microelectronics and Computer Technology Corporation
AI/KBS Program
9500 West Balcones Center Drive
Austin, TX 78759
(512) 343-0978

Abstract

This paper describes a method for implementing Explanation-Based Generalization (EBG) in a system which contains a Truth Maintenance System (TMS). It is shown that the TMS data dependency network provides the explanation structures needed by the EBG method. An EBG learning capability can therefore be easily added to a system with a justification-based TMS, without modifying the TMS. The paper discusses some interesting results of incorporating EBG in a TMS environment, one of which is nonmonotonic learning. Future research topics related to learning in general are also discussed.

1 Introduction

An intelligent system must be able to reason with whatever knowledge it has available, and then learn from the results of its reasoning. Moreover, the reasoning and learning processes must be robust and effective enough to be able to operate with incomplete knowledge in a changing world. A justification-based TMS provides a nonmonotonic reasoning capability that can maintain a consistent set of beliefs in such a world [7]. It allows inferences to be retracted by maintaining a cache, in the form of a dependency network, for all of the inferences that the reasoning system has made. EBG is a knowledge intensive approach for learning from examples of reasoning that has recently received extensive coverage in the research literature [3,4,5,6]. It derives a generalization from a single example by deductively justifying and explaining the example in terms of an underlying domain theory. We show that the TMS data dependency network constitutes the explanation structures required by EBG for generalization. Therefore, not only does a TMS allow a straightforward and efficient implementation of EBG, but also it enables learning to proceed nonmonotonically.

1.1 Explanation-Based Generalization

The ability to generalize from examples is essential for any machine learning system [1,2]. Explanation-Based Generalization is a method that enables a system to learn a generalized concept, or in terms of problem solving [4], a generalized sequence of operators or a macrorule that achieves a goal state, from a single training example. Based on a formulation by Mitchell, et al. [3], it requires four types of information:

1. Goal concept or goal state
2. Training example: a specific example of the goal concept, or a sequence of operators which transforms a specific initial state to the goal state
3. Domain theory: a set of inference rules and facts that can be used to explain the training example

4. Operationality criterion: a specification of how a learned concept definition must be expressed so it is "operational", or in problem solving terms, a specification of the types of the operators allowed in the generalized solution sequence so it may be readily applied to solve new problems.

The EBG method involves two steps:

1. Explain: use the domain theory to construct an explanation structure which provides an operational account for how the training example satisfies the goal.

2. Generalize: use a goal regression technique [9] to determine a set of sufficient conditions under which the general explanation structure holds.

1.2 Truth Maintenance Systems

A truth maintenance system is a subsystem of a problem solver that maintains logical justifications for program beliefs [7]. Whenever a deduction is made, a justification for the deduced conclusion is constructed which records the dependency links between the conclusion and the data that support its deduction. The primary function of the TMS is to assign belief statuses to data in accordance with the logical dependencies, preserving overall consistency and providing a well-founded basis for beliefs [7,8]. The belief status of a datum may be IN or OUT, which means the datum is currently believed or not believed, respectively.

Keeping track of the dependencies among data enables the system to efficiently revise its belief set when assumptions are retracted or new information is added; a TMS thus supports default reasoning and nonmonotonic reasoning. It also allows dependency directed backtracking, and facilitates the generation of comprehensive explanations.

2 TMS and EBG

In a TMS, every datum which the system can reason about or reason with is represented as a node. A node's justifications represent dependency links

between the node and other nodes which contribute to its belief. A node is IN if and only if it has at least one valid justification; it is OUT if and only if it does not have any valid justification. A justification consists of two sets of nodes: an IN-list and an OUT-list. A valid justification is one for which each node in its IN-list is IN and each node in its OUT-list is OUT. For example [10], if *Assertion g1 : (gray Clyde)* is derived from the following three data:

Rule r1: (believed)
((elephant X)
(unless (albino X))[1]
$\longrightarrow$
(gray X)),

Assertion e1: (believed) *(elephant Clyde)*,

Assertion a1: (NOT believed) *(albino Clyde)*,

then the TMS will construct a justification for $g1$ which has $r1$ and $e1$ in its IN-list and $a1$ in its OUT-list, represented by the pair: $((r1\ e1)\ (a1))$.

As described above, a node in a TMS is connected to others by justifications. The reason for a node's IN status can be obtained by tracing the dependency network that a TMS creates among nodes. The key observation of this paper is that the TMS data dependency network constitutes the explanation structures required by EBG for generalization. Further, the EBG goal concept is simply a justified datum, the training example is the situation that resulted in the datum and its justification being recorded in the TMS, the domain theory is the rule-based system that generated the justification or data-dependency network, and the operationality criterion is the set of predicates allowed at the leaves of the data-dependency network in order for EBG to succeed. This straightforward mapping means that it is relatively easy to add an EBG learning capability to a reasoning system already equipped with a TMS. An algorithm that accomplishes EBG using the information stored in the TMS is described next.

[1]Preceding an assertion by *unless* denotes negation-by-failure, i.e., the system is unable to establish a valid justification for the assertion.

3 The Algorithm

The purpose of the generalization step in EBG is two-fold: 1) obtaining a more general statement about the goal predicate by substituting variables for constants in the training example, and 2) compiling knowledge by combining several related rules into one more efficient macrorule. This is achieved by regressing the most general form of the goal predicate through the rules in the example's justification network completely (or until the operational criterion is met). To avoid over-generalization, it is necessary to maintain a substitution list for variables in the goal predicate throughout the regression process. Our method is very similar to the technique described in [4].

The generalization procedure is presented in terms of a rule-based representation of knowledge. Data in the system are either assertions or rules. Data have justifications; a datum with a justification of empty IN-list and empty OUT-list represents an axiom. The form X/S denotes the result of applying the substitution list S to the form X. A substitution list is a list of variable and binding pairs.

Given a specific example g and a valid justification j, the algorithm computes a macrorule for a general form of g as follows:

1. G := the form obtained by replacing all arguments of g with distinct variables;
   J := a justification with an empty IN-list and an empty OUT-list;
   S := null; (the substitution list for G)
   Antecedents := null; (the antecedents of the macrorule)
   Subgoals := a sequence containing G;
   Explanations := the IN-list of the justification for g;

2. If *Subgoals* is null, output the macrorule

   $$Antecedents \longrightarrow G/S$$

   with the justification J.

3. Let s be the first element of *Subgoals*. If s meets the operational criterion, is an *unless* clause (negation by failure), or unifies with an

assertion in *Explanations* which is an axiom, remove s from *Subgoals* and include it in *Antecedents*, then go to step 2.

4. If there is a rule R in *Explanations*, one of whose consequents unifies with s,

   add the resultant binding to S,
   remove s from *Subgoals*,
   let the antecedents of R be R_A and include R_A/S in *Subgoals*,
   include R in the IN-list of J,
   go to step 2.

   Else

   find an assertion in *Explanations* that unifies with s,
   add to *Explanations* the IN-list of one of the valid justifications for that assertion,
   go to step 4.

4 Discussion

There are several interesting implications of incorporating EBG into a system with a TMS involving the nonmonotonicity of the domain theory and the learning, and the flexibility of formulating the learning task. These constitute capabilities that the mapping of EBG onto the Soar problem-solving architecture does not provide [6].

4.1 Nonmonotonic Learning

Since part of the explanation structure (data-dependency network) from which the generalized concept or macrorule is constructed may become OUT (disbelieved) in a nonmonotonic system, it is necessary to place the learned concept or macrorule under the realm of the TMS by assigning it a nontrivial justification. A learned rule will then be unlearned if any part of the domain theory which supports its derivation loses validity. This means the system exhibits nonmonotonic learning. Those facts which are derived from the currently disbelieved rule will also become disbelieved

automatically. The justification for the learned concept or the macrorule contains the general domain rules from the explanation structure, but not facts pertaining to the specific training example. This means that the learned macrorule does not necessarily become OUT when the training example becomes OUT.

We will illustrate this point by extending an example from [4]. The domain theory includes the following rules:

k1: $((hate\ X\ Y)(possess\ X\ Z)(weapon\ Z) \longrightarrow (kill\ X\ Y))$

h1: $((depressed\ X) \longrightarrow (hate\ X\ X))$

p1: $((buy\ X\ Y) \longrightarrow (possess\ X\ Y))$

w1: $((gun\ X) \longrightarrow (weapon\ X))$

It also includes the following facts about John:

d1: $(depressed\ John)$

b1: $(buy\ John\ obj1)$

g1: $(gun\ obj1)$

While deriving the assertion

k2: $(kill\ John\ John)$

the system establishes the dependency network shown in Figure 1. Given the goal predicate $(kill\ X\ Y)$ and the justification $((k1\ h2\ p2\ w2)\ (\ ))$ for the example assertion $k2$, the generalization procedure will compute the macrorule

k3: $((depressed\ X)(buy\ X\ Y)(gun\ Y) \longrightarrow (kill\ X\ X))$

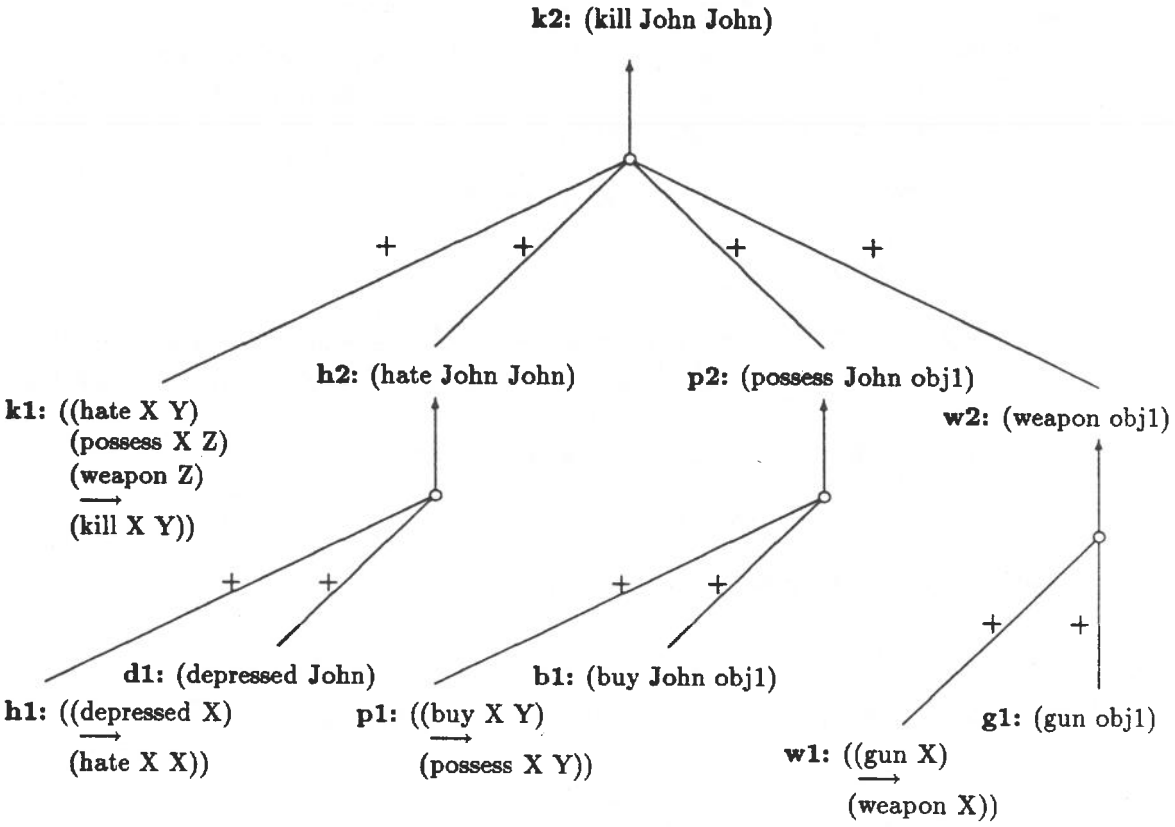

Figure 1: TMS data-dependency network / EBG explanation structure.

and assign it the justification $((k1\ h1\ p1\ w1)(\ ))$.

However, if later we discover that our theory of psychology is not correct, we may invalidate the rule $h1$ (making it OUT) and insert the following new rule into the knowledge base:

h2: $((depressed\ X)(introspective\ X)$
$\longrightarrow$
$(hate\ X\ X))$

The derived rule $k3$ will then become OUT because $h1$, one of its in-supportors (an element of IN-list), is no longer IN. Any assertion that is based on $k3$ will also become OUT.

If the knowledge base also contains the fact

$(introspective\ John)$

the system will be able to derive

$(kill\ John\ John)$

again and, if initiated, learn the following new macrorule:

k4: $((depressed\ X)$
$(introspective\ X)$
$(buy\ X\ Y)$
$(gun\ Y)$
$\longrightarrow$
$(kill\ X\ X))$

This suggests a way to tackle the problem of learning with an incomplete domain theory or a theory with defeasible rules. We may still apply EBG and risk over-generalization in such circumstances, but, as we (or the system itself if it is sufficiently intelligent) learn more about the domain and start to revise the knowledge base by making some existing rules OUT, the system will automatically retract over-generalizations which are based on those defeasible rules. Note that if new rules are introduced which enable another derivation of the example, the system will be able to relearn by invoking the generalization procedure again.

Another point about nonmonotonicity is that a TMS allows a system to maintain consistency under incremental revision of its knowledge base. This is accomplished by the TMS constantly updating and propagating changes throughout the dependency network. Combining EBG with TMS means the learning task does not have to assume a static view of the world. Facts about the domain and the operational criterion need not be fixed inputs to the EBG method. They may be modified as the requirements for the learning task change, as well as when the system's understanding of the world changes.

The domain theory might contain rules which are defeasible, due to the use of default reasoning or the presence of negated conditions (antecedents preceded by *unless*). Because the data-dependency network maintains an OUT-list (as well as an IN-list), corresponding to the use of defeasible rules for the explanation of a goal concept, the resultant macrorule might also contain defaults and negated conditions. The conclusions of the macrorule would thus be defeasible, as well as the rule itself.

4.2 Flexible Learning

The availability of a TMS allows a structuring of when learning occurs and how the information which is learned might be organized. The data dependency links are created and maintained by the TMS independently of the learning task. This means that the generalization task may be carried out sometime after the proof for the example is derived without the need to recalculate the explanation. The decoupling of the explanation and generalization steps, plus the fact that a TMS dependency network potentially encompasses the entire knowledge base, will allow more flexibility in formulating and scheduling the learning task.

This flexibility is beneficial, but it makes the still open issue of scheduling the generalization task even more complex. For example, based on previous problem solving activities, the TMS may have multiple valid justifications for an example that the learning task decides is worth generalizing. Which justification should it use? When there are several ways to explain a past experience, the system needs heuristics to guide it in deciding which explanation is the best candidate for generalization.

Another open question is how to integrate the learned macrorule into

the problem solving subsystem so that the resultant performance of the system will be improved. As the system continues learning, it may become burdened with a large number of similar rules of varying generality. How should the knowledge base be reorganized as rules are accumulated in order for the system to take advantage of the newly-acquired rules most effectively? After all, EBG only computes generalizations that are within the deductive closure of the domain theory. The question is not what assertions the system is able to derive, but how efficiently it can derive these assertions or whether it can do this more efficiently after seeing one example of the same derivation. The justifications maintained by the TMS for each learned rule are being investigated for use in a rule-ordering mechanism that appears to yield the desired efficiency.

5 Conclusions

A unification of EBG and TMS means that the TMS data dependency network is useful for learning, as well as for providing explanations and maintaining a consistent set of beliefs. The TMS can also maintain justifications for the rules derived by EBG. Thus, this unification yields a system with both nonmonotonic reasoning and nonmonotonic learning. The algorithm presented in this paper has been implemented in Proteus, a rule- and frame-based, nonmonotonic inference system being developed at MCC.

6 Acknowledgements

We are especially grateful to Tom Mitchell for many very interesting and stimulating discussions. We also thank members of the Proteus project at MCC for their help and support.

References

[1] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, eds., *Machine Learning, An Artificial Intelligence Approach, Vol. I*, Tioga Press, Palo Alto, CA, 1983.

[2] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, eds., *Machine Learning, An Artificial Intelligence Approach, Vol. II*, Morgan Kaufmann, Los Altos, CA, 1986.

[3] T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli, "Explanation-Based Generalization: A Unifying View", *Machine Learning*, Vol. 1, No. 1, 1986, pp. 47-80.

[4] G. DeJong and R. Mooney, "Explanation-Based Learning: An Alternative View", *Machine Learning*, Vol. 1, No. 2, 1986, pp. 145-176.

[5] R. J. Mooney, S. W. Bennett, "A Domain Independent Explanation-Based Generalizer", *Preceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, PA, August 1986, pp. 551-555.

[6] P. S. Rosenbloom, J. E. Laird, "Mapping Explanation-Based Generalization onto Soar", *Preceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, PA, August 1986, pp. 561-567.

[7] J. Doyle, "A Truth Maintenance System", *Artificial Intelligence*, Vol. 12, No. 3, 1979, pp. 231-272.

[8] D. Russinoff, *An Algorithm for Truth Maintenance*, MCC Technical Report AI-062-85, 1985.

[9] N. J. Nilsson, *Principles of Artificial Intelligence*, Tioga Press, Palo Alto, CA, 1980.

[10] C. J. Petrie, D. M. Russinoff, D. D. Steiner, *PROTEUS: A Default Reasoning Perspective*, MCC Technical Report AI-352-86, 1986.