# LEARNING FLEXIBLE CONCEPTS USING A TWO-TIERED REPRESENTATION

by

*R. S. Michalski*
*F. Bergadano*
*S. Matwin*
*J. Zhang*

Foundations of Knowledge Acquisition, Vol. 2: Machine Learning, pp. 145-202,
A. L. Meyrowitz and S. Chipman (Eds.), 1993.

# FOUNDATIONS OF KNOWLEDGE ACQUISITION: Machine Learning

*edited*
*by*

**Alan L. Meyrowitz**
*Naval Research Laboratory*

**Susan Chipman**
*Office of Naval Research*

**KLUWER ACADEMIC PUBLISHERS**
**Boston/Dordrecht/London**

# TABLE OF CONTENTS

# Chapter 5

## LEARNING FLEXIBLE CONCEPTS USING A TWO-TIERED REPRESENTATION

R. S. Michalski, F. Bergadano[1], S. Matwin[2] and J. Zhang

Center for Artificial Intelligence
George Mason University
Fairfax, VA 22030

### ABSTRACT

Most human concepts are *flexible* in the sense that they inherently lack precise boundaries, and these boundaries are often context-dependent. This chapter describes a method for representing and inductively learning flexible concepts from examples. The basic idea is to represent such concepts using a *two-tiered representation*. Such a representation consists of two structures ("tiers"): the *Base Concept Representation* (BCR), which captures explicitly the basic and context-independent concept properties, and *Inferential Concept Interpretation* (ICI), which characterizes allowable concept modifications and context-dependency. The proposed method has been implemented in the POSEIDON[3] system (also called AQ16), and tested on various practical problems, such as learning the concept of "Acceptable union contracts" and "Voting patterns of Republicans and Democrats in the U.S. Congress." In the experiments, the system generated concept descriptions that were both, more accurate and simpler than those produced by other methods tested, such as methods employing simple exemplar-based representations, decision tree learning, and some previous methods for rule learning.

---

1 On leave of absence from the University of Torino, Italy

2 On leave of absence from the Univerity of Ottawa, Canada.

3 The system is named after POSEIDON, the Greek god of the sea, water and waves, which represent fluidity and changing aspects of nature.

## INTRODUCTION

Typical assumptions underlying a large part of machine learning research are that concepts have precise boundaries, are context-independent, and are representable by a single symbolic description. An important consequence of this assumption is that recognizing instances of such concepts, which we call *crisp*, is very simple: if an instance satisfies a given concept description, then it belongs to the concept, otherwise it does not. Another common assumption is that concept instances are equally representative, that is there is no distinction in the typicality among instances.

In some methods, these assumptions are partially relaxed by assigning to a concept a fuzzy set membership function (e.g., Zadeh, 1974), or a probability distribution (e.g., Cheeseman et al., 1988; Fisher, 1987). However, once such a measure is defined explicitly for a given concept, the concept has a fixed, well-defined meaning. Moreover, these methods remain unsatisfactory for coping with context-dependency, handling exceptional cases, or for capturing gradual changes of knowledge about the concept properties.

When one looks at human concepts, one can see that most of them inherently lack precisely defined boundaries, and that their meaning is often context-dependent. Although on the surface these properties can be viewed as undesirable, one can argue that they contribute to a cognitive economy of human knowledge representations (Michalski, 1987). Our view is that this imprecision and context-dependency can be more adequately captured by rules of inference and flexible concept matching than by a probability distribution or a numerical set membership function. In other words, we postulate that the imprecision and context-dependency has often a logical, rather than a probabilistic character. This is confirmed by an observation that people usually decide about the concept membership of borderline instances through inference—by reasoning from general knowledge, drawing an analogy, or performing induction, rather than by conducting a statistical analysis.

Examples of human concepts can often be characterized by a *degree*

*of typicality* in representing the concept. For example, a robin is usually viewed as a more typical bird than a penguin or ostrich. The typicality is usually viewed as the degree to which an instance shares the common concept properties. Another property of concepts is that in different contexts they may have different meaning. For example, the concept "bird" may apply to a live, flying bird, a sculpture, a chick hatching out of the egg, or even an airplane. Thus, human concepts are *flexible*, as their boundaries have certain degree of fluidity, and can change with the context in which the concepts are used. It is clear that in order to learn such concepts, machine learning systems need to employ richer concept representations than are currently used.

This chapter describes an approach to learning flexible concepts based on the idea of *two-tiered representation* (TT), proposed by Michalski (1987). In this representation, a concept is described by two structures ("tiers"), the *base concept representation* (BCR), and the *inferential concept interpretation* (ICI). The BCR defines explicitly the basic properties of the concept, while the ICI describes implicitly, through rules and matching procedures, the allowed modifications of the explicit meaning, and its changes or extensions in different contexts. In the general definition of the two-tiered representation, the "distribution" of the meaning between the two tiers is not fixed, but depends on the properties of the reasoning agent, and on the criteria for evaluating the quality of concept descriptions. In the instantiation of the two-tiered approach that applies to modeling human concept representation, the BCR is assumed to describe the most typical, common, and intentional meaning of a concept, while the ICI would handle the exceptional or borderline cases, and context dependency (Michalski, 1990). The ICI for specific concepts is often inherited from more general concepts.

Early ideas, experiments and the first method for learning two-tiered concept representations were presented in (Michalski et al., 1986; Michalski, 1988; and Michalski, 1990). The general idea was to induce, in the first step, a concept description that is a complete and consistent characterization of all training examples. Such a description is often

overly complex and performs poorly on new examples, if the concept has flexible and/or complex borders, or examples are noisy. Therefore, in the second step, such descriptions are simplified or optimized according to some criterion of description "quality." The method employed a simple form of description simplification, called TRUNC, which removes those parts of the description that cover only a small fraction of examples (the so called *light disjuncts*, or *light rules)*. Such a description change can be logically interpreted as a specialization operation. As the ICI, the method applied a *flexible matching* procedure. An intriguing result of that research was that the description's complexity was substantially reduced without affecting its performance on new examples.

The new method, described here, significantly extends these early ideas. One important advance is the development of a heuristic double-level search procedure, called TRUNC-SG, which explores the space of two-tiered descriptions to derive a globally optimized description. The search employs both generalization and specialization operators, and is guided by a new criterion, the *general description quality* measure *(GDQ)*. This measure considers the accuracy of the description, the computational cost of both tiers - Base Concept Representation and Inferential Concept Interpretation, and its cognitive comprehensibility (Bergadano et al., 1988). By introducing such a general description quality measure, any form of concept learning can be viewed as a process of modifying the input concept description in order to maximize a given description quality measure. Initial concept description can be in the form of positive examples only, positive and negative examples, a complete and/or consistent concept description, an initial description supplied by a teacher, an abstract concept definition (as in the explanation-based learning), or a combination of these forms.

Another advance is that flexible matching is used not only in the recognition process, as in (Michalski et al., 1986), but also in the learning process, i.e., in searching for high "quality" concept descriptions. This feature also distinguishes the method from the related work described in (Bergadano and Giordana, 1989), which does not involve deductive

reasoning in the learning phase, and evaluates the performance of generated descriptions solely on the basis of the coverage of examples. These earlier approaches may be compared to using hands in learning how to row a boat, and then using oars in the performance phase.

The idea that learning is more effective if one uses the same instruments for learning and for performance phases was also present in some incremental learning systems (e.g., Fisher, 1987). The work described here represents also an important advance over tree-pruning techniques (e.g., Quinlan, 1987). These techniques apply a much more restrictive description reduction operator (a tree-pruning operator that performs a generalization of the class replacing the pruned subtree, and specialization of other classes), and do not use deductive matching or flexible interpretation of the learned descriptions. Other advances include the ability to take into consideration the typicality of training instances (when it is known), and the use of a rule base for the Inferential Concept Interpretation.

This chapter describes basic ideas of two-tiered representation, the method proposed, and experimental results from comparing it with several other methods, such as variants of exemplar-based learning, decision tree learning, learning complete and consistent descriptions, and the earlier method using two-tiered representation based on the TRUNC procedure. The experiments have shown that the proposed method compares favorably with other methods. The descriptions learned by the method were both simpler and had higher accuracy in classifying testing examples.

## TWO-TIERED CONCEPT REPRESENTATION

### Motivation and Definition

Traditional work on concept representation has assumed that the whole meaning of a concept resides in a single structure, e.g., a semantic network, a logic-based description, or a decision tree. Such a structure is expected to capture all relevant properties of the concept(s) and define the

concept boundary (e.g., Collins and Quillian, 1972; Minsky, 1975; Smith and Medin, 1981; Sowa, 1984). When concepts have flexible boundaries, or the learning examples have a considerable amount of noise, it may be advantageous to construct a concept representation that is partially inconsistent and/or incomplete with regard to the given examples. This idea was confirmed by the work on pruning decision trees (Quinlan, 1987), in HILLARY system (Iba et al., 1988), and in the work on two-tiered representation (Michalski, 1987; and this chapter).

In traditional approaches, the recognition of a concept instance is doen typically by directly matching the instance description with the stored concept representation. Such matching may include comparing feature values in an instance with those in the concept description, or tracing links in a semantic network, but is not assumed to involve any complex inferential processes. More recently, researchers working on exemplar-based reasoning (e.g., Bareiss, 1989; Kolodner, 1988 and Hammond, 1989) have proposed various inference mechanisms in order to classify new instances. In these methods, however, the concept representation consists of stored examples (cases). Such a representation taxes memory, and makes it difficult to compare different concepts.

The two-tiered representation employs a general concept description (BCR), and an inference mechanism (ICI) for matching the description with instances. Such concept representation can be much simpler than the one that stores individual examples, or their independent generalizations. The BCR can be viewed as a characterization of the "central tendency" of a concept; its contains the most relevant properties, and specifies the basic intention behind the concept. The ICI handles special cases, exceptions[4] and context-dependency. It treats them either by extending the base concept representation (concept *extension)*, or by specializing it (concept *contraction*). This process involves the background knowledge and relevant inference rules contained in the ICI. Inference allows the

---

4 The term "exceptions" is used here in its colloquial meaning. Subsection **Types of Match** gives it a precise meaning.

recognition, extension or modification of the concept meaning according to its context.

When an unknown entity is to be recognized, it is first matched against the Base Concept Representation. Then, depending on the outcome, the entity may be related to the concept's inferential extensions or contractions. A simple inferential matching can be merely a probabilistic inference based on some measure of similarity, e.g., the *flexible matching* method (Michalski et al., 1986). Advanced matching may involve any kind of inference —deductive, analogical or inductive. Let us illustrate the idea of two-tiered representation using the concept of "chair."

**BCR:** *Superclass:* A piece of furniture.

*Function*: To seat one person.

*Structure*: A seat supported by legs and a back rest attached from the side.

*Physical properties:* The number of legs is usually four. Often made of wood. The height of the seat is usually about 14-18 inches from the end of the legs, etc.

(BCR may also include a picture of 3D models of typical chairs)

**ICI:** *Possible variations of the properties in BCR:* The number of legs can vary from one to four. The legs may be replaced by any support. The shape of the seat, the legs and the backrest, and the material of which they are made are irrelevant, as long as the function is preserved. The backrest may be very small or missing, etc.

*Context dependency:*

*Context* = museum exhibit --> chair is not used for seating persons any more.

*Context* = toys --> the size can be much smaller than stated in BCR. The chair does not serve for seating persons, but correspondingly small dolls.

*Special cases:*

If legs are replaced by wheels --> type(chair) is wheelchair

Chair without the backrest --> type(chair) = stool

Chair with the armrests --> type(chair) = armchair

This simple example illustrates several important features of two-tiered representation. Commonly occurring cases of chairs match the BCR completely, and the ICI does not need to be involved. For such cases, the recognition time can thus be reduced. The BCR is not the same as a description of a prototype (e.g., Rosch and Mervis, 1975), as it can be a

generalization characterizing different typical cases or be a set of different prototypes. The ICI does not represent only distortions or corruptions of the prototype, but it can describe some radically different cases. When an entity does not satisfy the base representation of any relevant concept (which concepts are relevant is indicated by the context of discourse), or satisfies the base representation of more than one concept, the ICI is involved. The ICI can be changed, upgraded or extended, without any change to Base Concept Representation. While the BCR-based recognition involves just direct matching, the ICI-based recognition can involve a variety of transformations and any type of inference.

The ideas of two-tiered representation are supported by research on the so-called transformational model (Smith and Medin, 1981). In this model, matching object features with concept descriptions may transform object features into those specified in the concept description. Such a matching is inferential. Some recent work in cognitive linguistics also seems to support the ideas of two-tiered representation. For example, Lakoff (1987), in his idealized cognitive models approach, stipulates that humans represent concepts as a structure, which includes a fixed part and mappings that modify it. The fixed part is a propositional structure, defined relative to some idealized model. The mappings are metaphoric or metonymic transformations of the concept's meaning.

As mentioned before, in the general two-tiered model, the distribution of the concept meaning between BCR and ICI can vary, depending on the criterion of the concept description quality. For example, the BCR can be just concept examples, and ICI can be a procedure for inferential matching, as used in the cased-based reasoning approach. Consequently, the case-based reasoning approach can be viewed as a special case of the general two-tiered representation.

### Concept Representation Language

In the proposed method, the formalism used for concept representation is based on the *variable-valued logic system* $VL_1$ (Michalski, 1975). This formalism allows us to express simply and

implemented, **F** maps events from the set **E**, and concept descriptions from the set **D**, into the degree of match from the interval [0..1]:

$$\mathbf{F}: \ \mathbf{E} \times \mathbf{D} \ \text{--> } [0..1]$$

The value of **F** for an event $e$, and a concept description $D$, is defined as the probabilistic sum of **F** for its rules. Thus, if $D$ consists of two rules, $r_1$ and $r_2$, we have:

$$\mathbf{F}(e, D) = \mathbf{F}(e,r_1) + \mathbf{F}(e, r_2) - \mathbf{F}(e, r_1) \times \mathbf{F}(e, r_2)$$

A weakness of the probabilistic sum is that it is biased toward descriptions with many rules. If a concept description $D$ has a large number of rules, the value of $\mathbf{F}(e, D)$ may be close to 1, even if $\mathbf{F}(e,r)$ for each rule $r$, is relatively small (see Table 4). To avoid this effect, if the value of $\mathbf{F}(e,r)$ falls below a certain threshold, then it is assumed to be 0. (In our method this problem does not occur, because concept descriptions are typically reduced to only few rules; see the TRUNC-SG procedure in the subsection *Basic Algorithm*).

The degree of match, $\mathbf{F}(e,r)$ between an event $e$, and a rule $r$, is defined as the average of the degrees of fit for its constituent conditions, weighted by the proportion of positive examples to all examples covered by the rule:

$$\mathbf{F}(e,r) = (\sum_{i} \mathbf{F}(e, c_i / n) \times \#rpos /(\#rpos + \#rneg)$$

where $\mathbf{F}(e, c_i/n)$ is a degree of match between the event $e$ and the condition $c_i$ in the rule r, $n$ is the number of conditions in $r$, and $\#rpos$ and $\#rneg$ are the number of positive examples and the number of negative examples covered by $r$, respectively.

The degree of match between an event and a condition depends on the type of the attribute in the condition. Four types of attributes are distinguished: nominal, structured-nominal, linear and structured-linear (Michalski and Stepp, 1983).

Values of a structured-nominal (linear) attribute are nodes of an unordered (ordered) generalization hierarchy. In an ordered hierarchy, the children nodes of any parent node constitute a totally ordered set.

In a nominal or structured-nominal condition, the referent is a single value or an *internal disjunction* of values, e.g., [color = red v blue v green]. The degree of match is 1, if such a condition is satisfied by an event, and 0 otherwise. In a linear or structured-linear condition, the referent is a range of values, or an internal disjunction of ranges, e.g., [weight = 1..3 v 6..9]. A satisfied condition returns the value of match 1. If the condition is not satisfied, the degree of match is a decreasing function of the distance between the value and the nearest end-point of the interval. If the maximum degree of match between an example and all the candidate concepts is smaller than a preset threshold, the result is "no match."

**Inferential Concept Interpretation: Deductive Rules**

In addition to flexible matching, the Inferential Concept Interpretation includes a set of deductive rules that allow the system to recognize exceptions and context-dependent cases. For example, flexible matching allows an agent to recognize an old sequoia as a tree, although it does not match the typical size requirements. Deductive reasoning is required to recognize a tree without leaves (in the winter time), or to include in the concept of tree its special instance (e.g., a fallen tree). In fact, flexible matching is most useful to cover instances that are close to the typical case, while deductive matching is appropriate to deal with concept transformations necessary to include exceptions, or take into consideration the context-dependency.

The deductive inference rules in the Inferential Concept Interpretation are expressed as Horn clauses. The inference process is implemented using the LOGLISP system (Robinson and Sibert, 1982). Numerical quantifiers and internal connectives are also allowed. They are represented in the annotated predicate calculus (Michalski 1983).

**Types of Match.** The method recognizes three types of match between an event and a two-tiered description:

1. *Strict match*: An event matches the Base Concept Representation exactly, and it said to be S-covered.

2. *Flexible match*: An event is not S-covered, but matches the Base Concept Representation through a flexible matching function. In this case, the event is said to be F-covered.

3. *Deductive match:* the event is not F-covered, but it matches the concept by conducting a deductive inference using the Inferential Concept Interpretation rules. In this case, the event is said to be D-covered. (In general, this category could be extended to include also matching by analogy and induction; Michalski, 1989).

The above concepts provide a basis for proposing a precise definition of classes of concept examples that are usually characterized only informally. Specifically, examples that are S-covered are called *representative* examples; examples that are F-covered are called *nearly-representative* examples; and examples that are D-covered are called *exceptions*.

As mentioned earlier, one of the major advances of the presented method over previous methods using two-tiered representation (e.g., Michalski et al., 1986) is that the Inferential Concept Interpretation includes not only a flexible matching procedure, but also inference rules. Thus, using our newly introduced terminology, we can say that the method can handle not only representative or nearly representative examples, but also exceptions.

## AN OVERVIEW OF THE POSEIDON SYSTEM

### Basic algorithm

The ideas presented above have been implemented in a system called POSEIDON (also called AQ16). Table 1 presents two basic phases in which the system learns the Base Concept Representation.

The first phase generates a general consistent and complete concept description, and the second phase optimizes this description according to a General Description Quality measure. The optimization is done by applying different description modification operators:

**Phase 1**
*Given:*
Concept examples obtained from a some source
Relevant background knowledge
*Determine:*
Complete and consistent description of the concept
**Phase 2**
*Given:*
Complete and consistent description of the concept
A general description quality (GDQ) measure
Typicality of examples (if available)
*Determine:*
The Base Concept Representation that maximizes GDQ.

*Table 1.* Basic Phases in Generating BCR in POSEIDON.

The search process is defined by:

*Search space:* A tree structure, in which nodes are two-tiered concept descriptions (BCR + ICI).

*Operators:* Condition removal, Rule removal, Referent modification.

*Goal:* Determine a description that maximizes the general description quality criterion.

The complete and consistent description is determined by applying the AQ inductive learning algorithm (using program AQ15; Michalski et al., 1986). The second phase improves this description by conducting a "double level" best-first search. This search is implemented by the TRUNC-SG procedure ("SG" symbolizes the fact that the method uses both specialization and generalization operators). In this "double level" search, the first level is guided by a general description quality measure, which ranks candidate descriptions. The second level search is guided by heuristics controlling the search operators to be applied to a given description. The search operators simplify the description by removing some of its components, or by modifying the arguments or referents of

some of its predicates. A general structure of the system is presented in Figure 1.

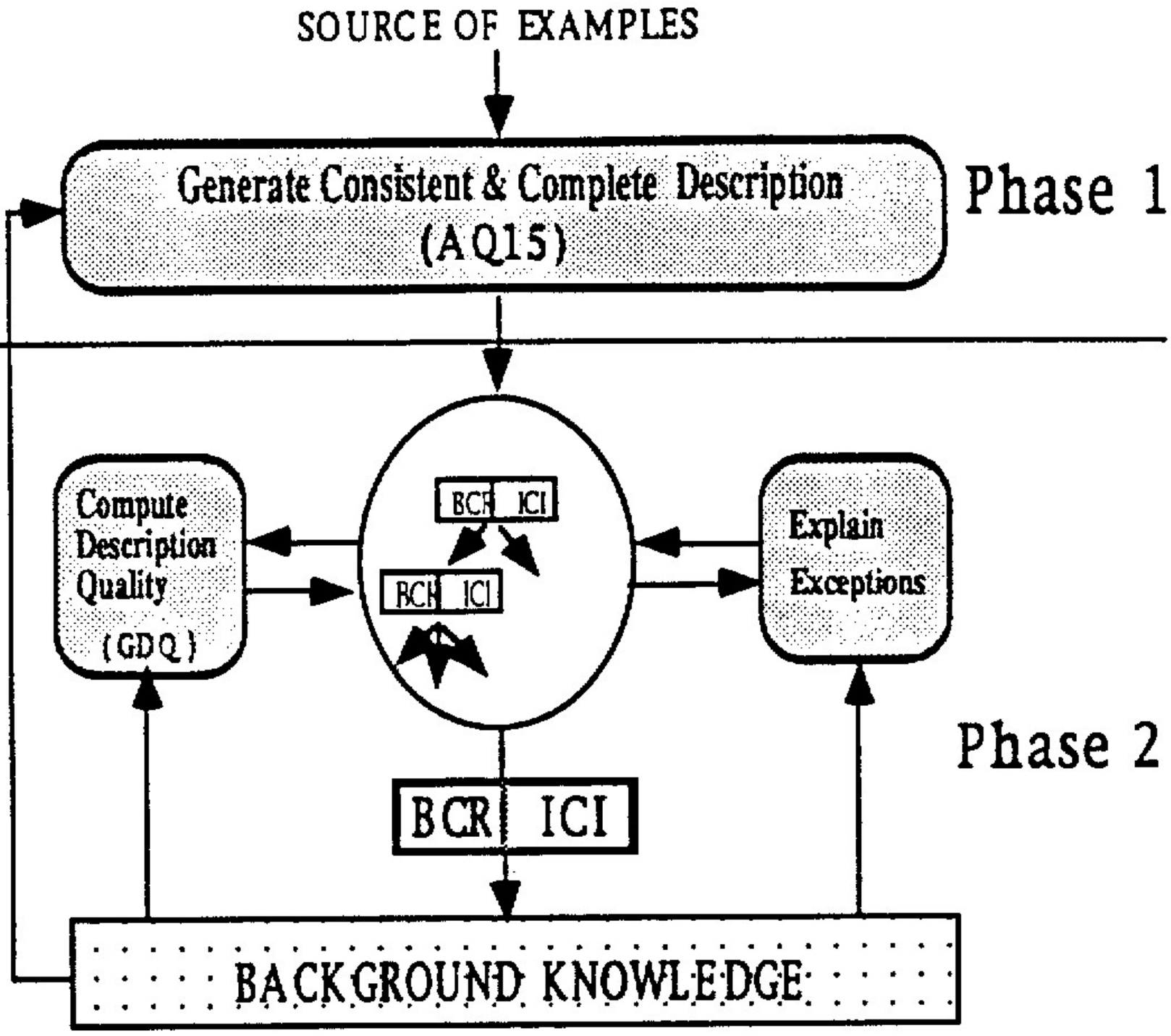


*Figure 1.* Learning Phases in POSEIDON.

The goal of the search is not necessarily to find an optimal solution, as this would require a combinatorial search. Rather, the system tries to maximally improve the given concept description by expanding only a limited number of nodes in the search tree. The nodes to be expanded are suggested by various heuristics discussed before.

The BCR is learned from examples. The Inferential Concept Interpretation contains two parts: a flexible matching function and a rule base. The rule base contains rules that explain exceptional examples, and is acquired through an interaction with an expert.

**Operators for Optimizing Base Concept Representation**

A description can be modified using three general operators: *rule removal, condition removal* and *referent modification.* The rule removal operator removes one or more rules from a ruleset. This is a specialization operator because it leads to "uncovering" some examples. It is the reverse of the "adding an alternative" generalization rule (Michalski, 1983). Condition removal (from a rule) is a generalization operator, as it is equivalent to the "dropping condition" generalization rule.

The referent modification operator changes the referent in a condition (i.e., the set of attribute values stated in a condition). Such changes can either generalize or specialize a description. Consequently, two types of referent modification operators are defined: *condition extension*, which generalizes the description, and *condition contraction*, which specializes the description.

To illustrate these two types of referent modification, consider the condition: [size = 1..5 v 7].

Changing this condition to :

[size = 1..7]

represents a condition extension operator. Changing it to

[size = 1..5]

represents a condition contraction operator. On the other hand, if the initial condition is [size ≠ 1..5 v 7], then changing it to [size ≠ 1..7], represents a condition contraction operator. Similarly, changing it to [size ≠ 1..5] represents a condition extension operator. A summary of the effect of different operators on a description is given in Table 2:

| **Search operator** | | **Type of knowledge modification** |
|---|---|---|
| Rule removal | (RR) | Specialization |
| Condition removal | (CR) | Generalization |
| Condition extension | (CE) | Generalization |
| Condition contraction | (CC) | Specialization |

*Table 2.* Search operators and their effect on the description

Thus, applying the above search operators can either specialize or generalize the given description. A generalized (specialized) description covers potentially a larger (smaller) number of training examples, which can be positive or negative. At any given search step, the algorithm chooses an operator on the basis of an evaluation of the changes in the coverage caused by applying the operator (see *Basic Algorithm* subsection).

**Learning the Inferential Concept Interpretation**

As indicated above, by applying a search operator (RR, CR, CE or CC) to the current Base Concept Representation, one can make it either more general or more specific. If the modified representation is more specific, some positive examples previously covered may cease to be S-covered. These examples may, however, be still covered by the existing Inferential Concept Interpretation (and thus would become F-covered or D-covered). On the other hand, if the modified base representation is more general than the original one, some negative examples, previously uncovered, may now become S-covered. They may, however, remain to be excluded by the existing Inferential Concept Interpretation rules.

Consequently, two types of rules in the Inferential Concept Interpretation can be distinguished: those that cover positive examples left uncovered by the base representation ("positive exceptions"), and rules that eliminate negative examples covered by the base representation ("negative exceptions"). A problem then is how to acquire these rules.

The rules can be supplied by an expert, inherited from higher level concepts, or deduced from other knowledge. If the rules are supplied by an expert, they may not be operationally effective, but they can be made so through analytic learning (e.g., Mitchell et al., 86; Prieditis and Mostow, 1987). If the expert supplied rules are too specific or partially correct, they may be improved inductively (e.g., Michalski and Larson, 1978; Dietterich and Flann 1988; Mooney and Ourston, 1989). Thus, in general, rules for the Inferential Concept Interpretation can be developed by different strategies.

In the implemented method, the system identifies exceptions (i.e., examples not covered by the Base Concept Representation), and asks an expert for a justification. The expert is required to express this justification in the form of rules. The search procedure, shown in Fig. 1, guides the process by determining examples that require justification. This way, the role of the program is to learn the "core" part of the concept from the supplied examples, and to identify the exceptional examples. The role of a teacher is to provide concept examples, and to justify why the examples identified by the learning system as exceptions are also members of the concept class.

## QUALITY OF CONCEPT DESCRIPTIONS

### Factors Influencing the Description Quality

The learning method utilizes a general description quality measure that guides the search for an improved two-tiered description. The General Description Quality measure takes into consideration three basic characteristics of a descritpion: its *accuracy*, *comprehensibility*, and its *cost*. This section discusses these three components, and describes a method for combining them into a single measure.

The accuracy expresses the description's ability to produce correct classifications. Major factors in estimating the description's predictive power are its degree of completeness and consistency with regard to input examples. When learning from noisy examples, however, to achieve a high degree of completeness and consistency may lead to an overly complex and overspecialized description. Such a description may be well tuned to the particular training set, but may perform poorly in classifying future examples. For that reason, when learning from imperfect inputs, it may be better to produce descriptions that are only partially complete and/or consistent.

If an intelligent system is supposed to give advice to humans, knowledge used by such a system should be comprehensible to human experts. A "black box" classifier, even with a high predictive power, is not satisfactory in such situations. To be comprehensible, a description should

involve terms, relations and concepts that are familiar to experts, and be syntactically simple. This requirement is called the *comprehensibility principle* (Michalski, 1983). Since there is no established measure of description's comprehensibility, we approximate it by the *representational simplicity*. Such a measure is based on the number of different operators involved in the description: disjunctions, conjunctions, and the relations embedded in individual conditions. In the case of two-tiered representations, the measure takes into account the operators occurring in both, the BCR and the ICI, and weighs the relative contribution of each part to the comprehensibility of the whole description.

The third criterion, the *description cost*, captures the cost of storing the desription and using it in computations to make a decision. Other things being equal, descriptions which are easier to store and easier to use for recognizing new examples are preferred. When evaluating the description cost, two characteristics are of primary importance. The first is the cost of measuring values of variables occurring in the description. In some application domains, e.g., in medicine, this is a very important factor. The second characteristic is the computational cost (time and space) of evaluating the description. Again, in some real-time applications, e.g., in speech or image recognition, there may be stringent constraints on the evaluation time. The cost and the comprehensibility of a description are frequently mutually dependent, but generally these are different criteria.

The criteria described above need to be combined into a single evaluation measure that can be used to compare different concept descriptions. One solution is to have an algebraic formula that, given numeric evaluations for individual criteria, produces a number that represents their combined value. Such a formula may involve, e.g., a multiplication, weighted sum, maximum/minimum, or t-norm/t-conorm of the component criteria (e.g., Weber, 1983).

Although the above approach is often appropriate, it also has significant disadvantages. First, it combines a set of heterogeneous

evaluations into a single number, and the meaning of this final number is hard to understand for a human expert. Second, it usually forces the system to evaluate all the criteria for each description, even if it is sufficient to compare descriptions on the basis of just one or two most important ones. The latter situation occurs when one description is so much better than the other according to some important criterion, that it is not worth to even consider the alternatives. To overcome these problems, we use a combination of a lexicographic evaluation and a linear function-based evaluation, which is described in the next section.

**Combining Individual Factors Into a Single Preference Criterion**

Given a set of candidate descriptions, we use the General Description Quality criterion to select the "best" description. Such a criterion consists of two measures, the *lexicographic evaluation functional* (LEF), and the *weighed evaluation functional* (WEF). The LEF, which is computationally less expensive than WEF, is used to rapidly focus on a subset of the most promising descriptions. The WEF is used to select the final description. A general form of a LEF (Michalski, 1983) is:

$$\text{LEF: } \langle(\text{Criterion}_1,\tau_1), (\text{Criterion}_2,\tau_2), ..., (\text{Criterion}_k,\tau_k)\rangle$$

where $\text{Criterion}_1$, $\text{Criterion}_2$, ... , $\text{Criterion}_k$ are elementary criteria used to evaluate a description, and $\tau 1, \tau 2, ... ,\tau k$ are corresponding *tolerances*, expressed in %. The criteria are applied to every candidate description in order from the left to right (reflecting their decreasing importance). At each step, all candidate descriptions whose score on a given criterion is within the tolerance range from the best scoring description on this criterion are considered equivalent with respect to this criterion, and are kept on the CANDIDATE LIST; other descriptions are discarded. If only one description remains on the list, it is chosen as the best. If the list is non empty after applying all criteria, a standard solution is to chose the description that scores highest on the first criterion. In POSEIDON, we chose another approach in the latter case (see below).

The LEF evaluation scheme is not affected by the problems of using a linear function evaluation, mentioned above. The importance of a

criterion depends on the order in which it is evaluated in LEF, and on its tolerance. Each application of an elementary criterion reduces the CANDIDATE LIST, and thus the subsequent criterion needs to be applied only to a reduced set. This makes the evaluation process very efficient. In POSEIDON, the default LEF consists of the three elementary criteria discussed above, i.e., accuracy, the representational simplicity and the description cost, specified in that order. The next section describes them in detail.

Tolerances are program parameters, and are set by the user. If the tolerance for some criterion is too small, the chances of using the remaining criteria decrease. If the tolerance is too large, the importance of the criterion is decreased. For this reason, the LEF criteria in POSEIDON are applied with relatively large tolerances, so that all the elementary criteria are taken into account. If after applying the last criterion the CANDIDATE LIST has still several candidates, the final choice is made according to a *weighed evaluation functional* (WEF). The WEF is a standard linear function of elementary criteria. The description with the highest WEF is selected.

Thus, the above approach uses a computationally efficient LEF to obtain a small candidate set, and then applies a more complex measure to select from it the best description.

**Taking the Typicality of Examples into Consideration**

Accuracy is a major criterion to determine the quality of a concept description. In determining accuracy, current machine learning methods usually assume that it depends only on the number of positive and negative examples (training and/or testing) correctly classified by the description. One can argue, however, that in evaluating accuracy one might also take into consideration the *typicality* of examples (Rosch and Mervis, 1975). If two descriptions cover the same number of positive and negative examples, the one that covers more typical positive examples and fewer typical negative examples can be considered more accurate.

For the above reason, we propose a measure of completeness and

consistency of a description that takes into account the typicality of the examples. In POSEIDON, the typicality of examples can be obtained in one of two ways.

The first way is that the system estimates it by the frequency of the occurence of examples in the data (notice that this is different from a usual cognitive measure of typicality, which captures primarily the degree to which an example resembles a prototypical example). The second way is that the typicality of examples is provided by an expert who supplies training examples. If the typicality is not provided, the system makes the standard assumption that the typicality is the same for all examples.

In the measures below, the degree of completeness of a description is proportional to the typicality of the positive events covered, and the consistency is inversely proportional to the typicality of the negative events covered[5]. Since the system is working with a two-tiered description, other factors are taken into account.

One is that according to the idea of two-tiered representation, a "high quality" concept description should cover the typical examples explicitly, and the non-typical ones only implicitly. Thus, the typical examples should be covered the Base Concept Representation, and non-typical, or exceptional ones by the Inferential Concept Interpretation.

In POSEIDON, the Base Concept Representation is inductively learned from examples provided by a teacher. Therefore, the best performance of the system will be achieved if the training set contains mostly typical examples of the concept being learned. For the exceptional examples, the teacher is expected to provide rules that explain them. These rule become part of the Inferential Concept Interpretation. An advantage of such an approach is that the system learns a description of typical examples by itself, and the teacher needs to explain only the special cases.

---

5 When negative examples are instances of another concept, as is often the case, their typicality is understood as the typicality of being members of that other concept.

In view of the above, the examples covered explicitly (*strictly*-covered, or S-COV) are assumed to contribute to the completeness of a description more than flexibly-covered (F-COV) or deductively-covered (D-COV).

**General Description Quality Measure**

This section defines the General Description Quality (GDQ) measure implemented in POSEIDON. As mentioned above, the measure combines the accuracy, representational simplicity and the cost of a description.

The accuracy is based on two factors, the *typicality-based completeness*, T_COM, and the *typicality-based consistency*, T-CON. These two factors are defined for a two-tiered concept description, $\mathcal{D}$, as follows:

$$T\_COM(\mathcal{D}) = \frac{\sum_{e^+ \in S\text{-}cov} w_s * Typ(e+) + \sum_{e^+ \in F\text{-}cov} w_f * Typ(e+) + \sum_{e^+ \in D\text{-}cov} w_d * Typ(e+)}{\sum_{e^+ \in POS} Typ(e^+)}$$

$$T\_CON(\mathcal{D}) = \frac{\sum_{e^- \in S\text{-}cov} w_s * Typ(e^-) + \sum_{e^- \in F\text{-}cov} w_f * Typ(e^-) + \sum_{e^- \in D\text{-}cov} w_d * Typ(e^-)}{\sum_{e^- \in NEG} Typ(e^-)}$$

where POS and NEG are sets of positive and negative examples, respectively, which are covered by the two-tiered concept description $\mathcal{D}$. Typ(e) expresses the degree of typicality of example e of the given concept. Weights $w_s$, $w_f$, and $w_d$ represent different significance of the type of coverage (S-COV, F-COV, and D-COV). Thresholds $t_1$, and $t_2$ reflect the desirability of a given type of coverage for the given degree of typicality:

$w_s$: if Typ(e) $\geq t_2$, then 1, else w
$w_f$: if $t_2 >$ Typ(e) $\geq t_1$, then 1, else w
$w_d$: if $t_1 >$ Typ(e), then 1, else w

where thresholds $t_1$ and $t_2$ satisfy the relation $0 \leq t_1 \leq t_2 \leq 1$, and $0 < w < 1$.

The role of w is to decrease the weight the examples that are covered in a way (S, F or D) that is not compatible with their typicality.

Using the terms of T_COM and T_CON, the description *accuracy* is defined as:

*Accuracy* = $w_1$*T_COMPLETENESS +$w_2$*T_CONSISTENCY

where $w_1+w_2=1$. The weights $w_1$ and $w_2$ reflect the expert's judgment about the relative importance of completeness and consistency for the given problem. The default value of both is 0.5.

A measure of comprehensibility of a concept description is difficult to define. As mentioned earlier, we approximate it by a representational simplicity, defined as:

$$RepSimplicity(\mathcal{D}) = TC - (v_1 * \sum_{op \in BCR(\mathcal{D})} C(op) + v_2 * \sum_{op \in ICI(\mathcal{D})} C(op))$$

where TC is the sum of the complexities of all operators in the description $\mathcal{D}$. BCR($\mathcal{D}$) is the set of all operator occurrences in the BCR of the description, and ICI(dsp) is the set of all operator occurrences in the ICI. C(op), the *complexity* of an operator, is a real function that maps each operator symbol into a real number representing its complexity. The complexities of the operators are chosen by an expert, assuming the following constraints:

C(range) < C(internal v ) < C(=) < C(<>) < C(&) < C(v) < C(⇒).

When the operator is a predicate, C increases with the number of the arguments. Parameters $v_1$ and $v_2$ represent relative weights of the operators in BCR and ICI, respectively, assuming $v_1+v_2 = 1$.

The Base Concept Representation is supposed to describe the general and easy-to-define meaning of the concept, while the Inferential Concept Interpretation is mainly used to handle rare or exceptional events. As a consequence, the Base Concept Representation should be easier to comprehend than the Inferential Concept Interpretation, and thus $v_1$ should be larger than $v_2$. The *cost* of a description $\mathcal{D}$ depends on two factors:

- *Measuring-Cost* (MC) -- the cost of measuring variables used in the concept description

$$MC(\mathcal{D}) = \sum_{e \in Pos+Neg} \sum_{v \in Vars(e)} mc(v) \,/\, (|Pos| + |Neg|)$$

- *Evaluation-Cost* (EC) -- the cost of evaluating the concept description

$$EC(\mathcal{D}) = \sum_{e \in Pos+Neg} ec(e) \,/\, (|Pos| + |Neg|)$$

where $Vars(e)$ is the set of all variables occuring in the concept description, $mc(v)$ is the cost of measuring the value of the variable $v$, and $ec(e)$ is the computational cost of evaluating the concept description to classify the event e. The latter depends on the computing time and/or on the number of operators involved in the evaluation. We now define the cost of a description:

$$Cost(\mathcal{D}) = u_1 * MC(\mathcal{D})) + u_2 * EC(\mathcal{D})$$

where $u_1$ and $u_2$ are weights defining the relative importance of the measuring-cost and the evaluation-cost for a given problem.

The general description quality (GDQ) measure is in the form of a Lexicographic Evaluation Functional (LEF), in which the above defined concepts of accuracy, representational simplicity and the description cost are used as elementary criteria. The tolerances and other parameters defined above can be chosen by a user to reflect the problem domain, or determined experimentally. They also have default values, so that the user does not have to specify them. More details about the general description quality measure are in (Bergadano et al., 1988).

## LEARNING BY MAXIMIZING DESCRIPTION QUALITY

As mentioned before, learning a base concept representation (BCR) of a concept is performed in two phases. In the first phase, a complete and consistent concept description is learned inductively from examples. In the second phase, the obtained complete and consistent description is optimized according to the general description quality criterion.

In POSEIDON, the first phase is done using the AQ15 learning program, described in (Michalski et al., 1986a). The following subsections

describe the second phase (the TRUNC-SG procedure).

### Search Heuristics for Optimizing Base Concept Representation

The task of optimizing BCR by directly applying the General Description Quality measure is computationally expensive. It requires that every newly generated description is matched flexibly against all training examples. To make this process more efficient, a *double-level* search method is employed. The first level uses a simple heuristic to determine which operator, RR, CR, CE or CC, is likely to improve the description, and the second level actually applies the operator, and evaluates the description according to the General Description Quality measure.

The first level applies the so-called *Potential Accuracy Improvement* heuristic (PAI). The PAI is a function of the change in the coverage of positive and negative examples by the description due to an operator application. Specifically:

$$PAI = \Delta P/TP - \Delta N/TN$$

where $\Delta P$ ($\Delta N$) is the *change* in the number of positive (negative) examples that would be covered by the description after applying the operator, and TP (TN) is the total number of positive (negative) examples. For generalizing operators, SR and CE, $\Delta P$ and $\Delta N$ are non-negative, and for specializing operators, CR and CC, $\Delta P$ and $\Delta N$ are non-positive.

The advantage of the Potential Accuracy Improvement measure is that it can be computed much more efficiently than the General Description Quality. For every condition in the current description, a list of examples covered by it is maintained using bit vectors. The sets of examples covered by a ruleset (representing a complete description) is then obtained by intersection and union operations.

The matching time can be improved further by also maintaining bit vectors for the examples covered by rules (the matching time trades off with the memory for storing the bit vectors). Note that computing the General Description Quality requires flexible matching, and thus cannot be done by an intersection and union operations on bit vectors.

The above formula does not take into consideration the degree of

reduction of the description complexity caused by applying an operator. For example, removing a rule reduces complexity more than removing a condition. To account for this, POSEIDON assigns a higher weight (preference) to applying the RR operator (rule removal) than for applying the CR operator (condition removal).

The condition removal operator generalizes the description, therefore, the description (ruleset) resulting from its application may cover some additional examples (positive or negative). Due to this, some rule(s) may become redundant. If the CR operation produces a rule that differs from another rule only in the value of one attribute, the two rules can be merged into one, in which the attribute is related to the internal disjunction of values (this is a case of the so-called "refunion" operation; see Michalski and Stepp, 1983).

For example, the rules [shape = circle]&[size = 2..6] and [shape = square]&[size = 2..6] can be replaced by single rule [shape = circle v square]&[size = 2..6].

It is worth noting that in the case of operators RR and CR, the Potential Accuracy Improvement heuristic can be simplified by using an approximation:

$$PAI' = \#P/TP - \#N/TN$$

where #P (#N) is the number of positive (negative) examples covered by the *component* (rule or condition) to be removed. Such a heuristic is very efficient because it needs to be computed only once for every condition and every rule in the initial description. This computation can be done before the search starts, and does not need to be repeated for every node in the search. The operator that produces the largest Potential Accuracy Improvement is chosen, and applied to the description under consideration. The descriptions so generated are then subjected to an evaluation by the General Description Quality criterion.

The search algorithm (TRUNC-SG) is presented in Table 3. Let us explain the motivation and individual steps of the algorithm. Step 1

chooses the node (description) for expansion on the best-first basis, that is, chooses the node with the highest General Description Quality.

This is not always an optimal choice, because "worse" nodes can sometimes lead to better descriptions after a number of removals. Whether the search will behave in this manner will depend on the adequacy of the General Description Quality as the measure of concept quality.

---

***Search Algorithm (TRUNC-SG***

1. Identify in the search tree the best candidate description D.
   (Initially, D is the complete and consistent description obtained by AQ15 in Phase I. Subsequently, it is the highest ranked description according to the General Description Quality criterion).
2. Apply to D that operator, selected from among the operators

   $RR_i$ - Remove the i-th rule, or

   $CR_{ij}$ - Remove the j-th condition from the i-th rule

   $CC_{ij}$ - Contract the referent of the j-th condition in the i-th rule

   $CE_{ij}$ - Extend the referent of the j-th condition in the i-th rule

   that maximizes the Potential Accuracy Improvement measure.
3. Compute the General Description Quality (GDQ) of the description obtained in step 2. If the GDQ of this description does not exceed the GDQ of the original D by more than D (an experimental threshold), then proceed to step 1. (Computing the description accuracy for GDQ employs flexible matching).
4. Identify exceptional examples that are

   (a) the positive examples that cease to be covered, and

   (b) the negative examples that become covered.

   Ask an expert to provide rules explaining these examples. If such rules are obtained, add them to the Inferential Concept Interpretation; otherwise, add the exceptional example(s) to it.
5. Update the GDQ value of the new node by taking into account the added Inferential Concept Interpretation.
6. If the *stopping criterion* is satisfied, then STOP, otherwise proceed to step 1.

---

*Table 3*. The algorithm for Optimizing a Concept Description.

Step 2 chooses the "best" search operator according to the Potential Accuracy Improvement heuristic, and applies it to the current description. Step 3 computes the General Description Quality of the new node. It should be noted that, in the General Description Quality measure, the typical examples covered directly by the base representation can weigh more than those covered through flexible matching. The examples covered by Inferential Concept Interpretation rules weigh more than the ones covered through flexible matching, but less than the ones covered by the Base Concept Representation. A new description (node) is worth to consider only if it "sufficiently" better (more than $\Delta$) than the previous one, otherwise the control goes to Step 1 (the reason for this is given below).

Step 4 determines exceptional examples, and asks an expert for an explanation of them. If the explanation is provided, appropriate rules are added to the Inferential Concept Interpretation. These rules may extend or contract the Base Concept Representation. For example, the rule removal operator might uncover some positive examples, that were previously covered. In this case, new rules added to the Inferential Concept Interpretation would allow the system to reason about such "special" positive examples, and explain why they should be classified as instances of the concept being learned. On the other hand, the condition removal operator might cause some negative examples to be covered. In this case, new Inferential Concept Interpretation rules would have to be added to contract the Base Concept Representation. An important issue concerning step 4 is when an explanation should be required from an expert ("explainer"). The problem is that in some cases the chosen operator may not be appropriate, because it leads to a very poor description. In such a case, it is not worthwhile to ask an expert for an explanation, and search should continue in other direction.

The method employs the following strategy. Suppose that $N$ is the node (description) to be expanded, and $M$ is the node obtained after applying an operator, e.g., the condition removal. The effort to obtain an explanation is made only if the General Description Quality of $M$ is

"significantly" better than that of $N$ (above a certain threshold T). In this case, the explainer is given the General Description Quality evaluations of both descriptions, $N$ and $M$, and asked for an explanation. These evaluations give the explainer a sense of importance of the request. If the explainer cannot provide an explanation, the exceptional examples are directly added to the Inferential Concept Interpretation. Step 5 updates the General Description Quality of the obtained two-tiered description by taking into consideration the added Inferential Concept Interpretation rules. Step 6 decides whether to stop or continue the search. The *stopping criterion* is satisfied when the number of nodes explored exceeds value *k1*, or when the General Description Quality is not improved after the exploration of *k2* nodes since the last improvement. The search parameters *k1* and *k2* have a default value, which is modifiable by the user. When the search stops, the best node found until this point defines the chosen two-tiered concept description.

In conclusion, let us point out the main difference between the above two-level search and the standard best-first search. The difference is that only one operator is applied to the (best-GDQ) node selected for expansion, rather than all available operators, as in the standard search. The operator applied is the "best" according to the PAI heuristic. Such a procedure helps to avoid generating low quality nodes, and thus makes unnecessary the computation of the General Description Quality for these nodes. Other operators are applied only if the results obtained along this branch of the search tree turn out to be unsatisfactory.

**An Abstract Example**

An abstract example of the search process is given in Figure 2. Individual nodes represent both components of a two-tiered description (BCR and ICI) generated at any given search step, and show the coverage of training examples by the description. The rectangular areas represent the coverage by the Base Concept Representation, and the curved lines denote the coverage by the Inferential Concept Interpretation. In the example, the accuracy is computed according to the formula described before, assuming that all examples have the same typicality.

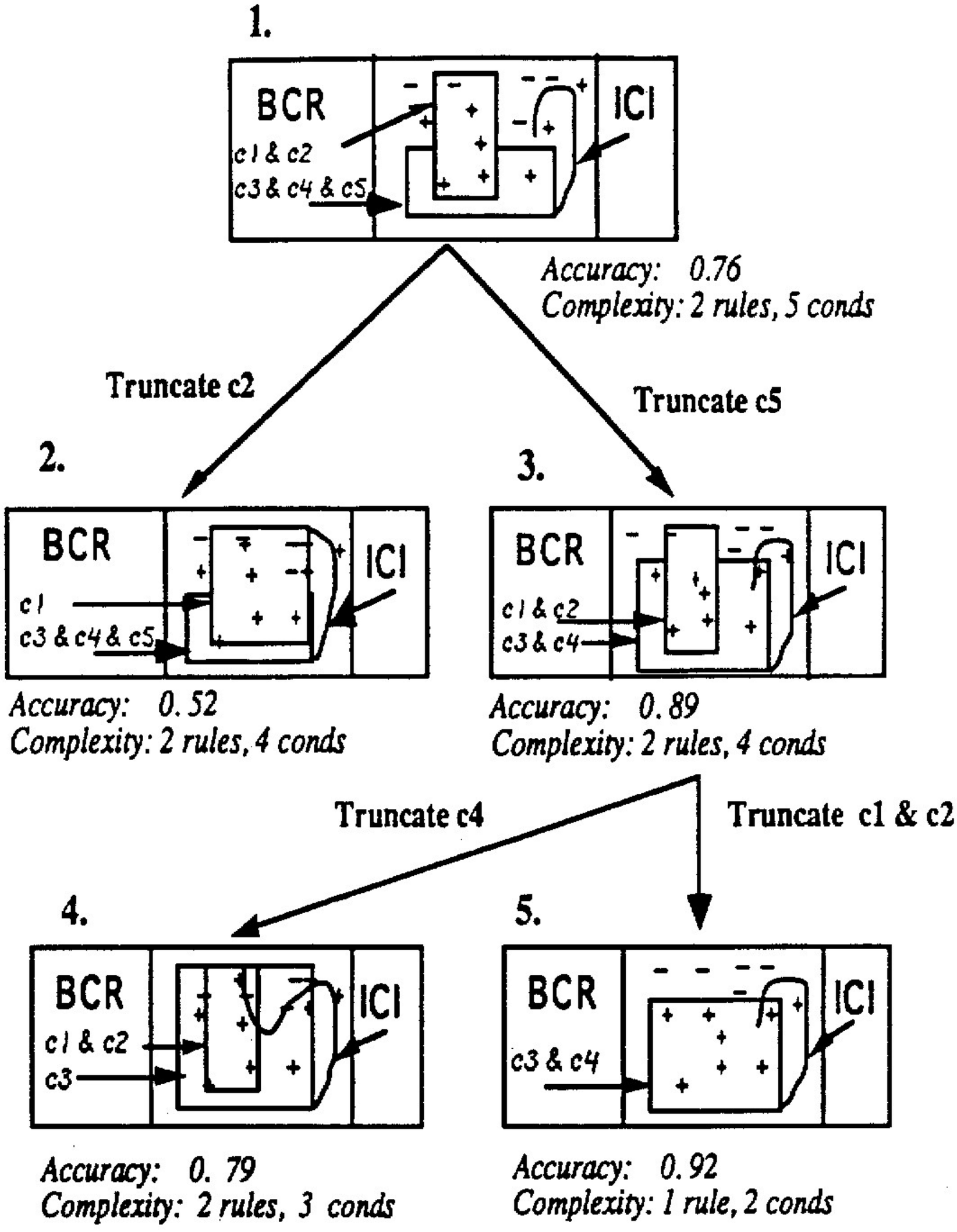


*Figure 2.* An Illustration of the Search Process.

The initial description is represented by node 1. The BCR contains two rules represented by two rectangular areas, which cover five positive examples out of eight, and one negative example out of five. The

Inferential Concept Interpretation extends this coverage by recognizing one more positive example. Next nodes correspond to descriptions obtained by an application of operators marking the branches of the search tree. For example, node 3 is obtained by eliminating condition c5 in the second rule of the initial description. The new description is more accurate because all positive examples are now covered, without changing the coverage of the negative examples.

By truncating the first rule in node 3, node 5 is generated. The description no longer covers negative examples, and is simpler. This node is then accepted as the optimized description resulting from the search. The other nodes lead to inferior concept representations with respect to General Description Quality, and are discarded. The quality has been computed with w1=w2=0.5. For simplicity, the cost is omitted, and the complexity of the Inferential Concept Interpretation is ignored. The complexity of the Base Concept Representation is indicated by the number of rules and the number of conditions.

## EXPERIMENTS

The proposed method was implemented in the POSEIDON system (also called AQ16). To evaluate its performance, it was tested, together with several other methods, in two problem domains. The other methods tested included: simple forms of exemplar-based learning, learning consistent and complete descriptions (implemented in AQ15), generating *top rule* descriptions (described by Michalski et al., 1986), and generating pruned decision trees (implemented in the ASSISTANT program; Cestnik, Kononenko & Bratko, 1987). All these methods were applied to the same training data, and tested on the same testing data from the two problem domains.

The first domain was labor-management contracts, and the problem was to learn a general description that discriminates between acceptable and unacceptable contracts. The second domain was congressional voting, and the problem was to characterize the voting behavior of Republicans and Democrats in the US House of Representatives.

## Experimental Data

**Labor-management contracts.** The data regarding labor-management contracts were obtained from *Collective Bargaining*, a review of current collective bargaining issues published by the Department of Labor of the Government of Canada. The data describe labor-management contracts negotiated between various organizations and labor unions with at least 500 members, and concluded in the second half of 1987 or the first half of 1988. The experiments focused on the personal and business services sector. This sector includes unions representing hospital staff, teachers, university professors, social workers and certain classes of administrative personnel.

The data involved multivalued attributes, and thus the $VL_1$ language was directly applicable. Each contract is described by sixteen attributes, belonging to two main categories. One category concerns issues related to the salaries, e.g., pay increases in each year of the contract, the cost of living allowance, a stand-by pay, etc., and the second category concerns issues related to fringe benefits, e.g., different kinds of pension contributions, holidays, vacation, dental insurance, etc. Positive examples represent contracts that have been accepted by both parties. Negative examples represent contracts deemed unacceptable at least by one of the parties. Here is an example of an acceptable labor-management contract:

*Duration of the contract = 2 years*
*Wage increase in the first year = 1.5%*
*Wage increase in the second year = 3.5%*
*Cost-of-living-allowance = unknown*
*Hours of work/per week = 38*
*Pension offer = none*
*Stand-by pay = $0.12/hr*
*Shift differential = second shift is paid 25% more than first shift*
*Educational allowance is offered*
*Holidays per year= 11 days*
*VacLength = better than average in the industry*
*Long term disability insurance = offered by the employer*
*50% dental insurance cost = covered by the employer*
*Bereavement leave = available*
*Employer-sponsored health plan = not mentioned*

The above description is represented as the following $VL_1$ rule:

[Dur = 2] [Wage1 = 1.5] [Wage2 = 3.5] [Cola = unknown] [Work-hours = 38] [Pension = none] [StbyPay = 12] [ShiftDff = 25] [Educ-allw = yes] [Hlds = 11] [VacLen = better][LngTrmDisbll = true] [Dntl-ins = half] [Bereavement = yes] [EmpHlthPln= unknown] ::> [Contract Class = acceptable]

In the rule above, and next rules, the following abbreviations were used:

SByPay for "Stand-by-pay"
Vacation for "Vacation length"
Hlds for "Holidays per year,"
LngTrmDisbl "Long term disability insurance,"
EmpHlthPln for "Employer-sponsored health plan"
ShiftDff for Shift differential
Contract Class for "Contract classification"

Also, for simplicity, the conjunction is represented by concatenation. The training set consisted of 18 positive and 9 negative examples of contracts; the testing set consisted of 19 positive and 11 negative examples.

**US Congress Voting record.** The data regarding the US Congress voting record were the same as the ones used by Lebowitz (1987) in his experiments on conceptual clustering. The data represent the 1981 voting records of 100 selected representatives (50 in the training set and 50 in the testing set). The problem was to learn descriptions discriminating between the voting record of Democrats and Republicans. Below is an example of the voting record of a Democrat in the US Congress:

*Draft registration = no*
*Ban aid to Nicaragua = no*
*Cut expenditure on MX missiles = yes*
*Federal subsidy to nuclear power stations = yes*
*Subsidy to national parks in Alaska = yes*
*Fair housing bill = yes*
*Limit on PAC contributions = yes*
*Limit on food stamp program = no*
*Federal help to education = no*
*State = north east*
*Population = large*
*Occupation = unknown*
*Cut in Social Security spending = no*
*Federal help to Chrysler Corp. = vote not registered*

## A Description of Experiments

For each problem domain, the experiments involved the following steps:

1. Learn a complete and consistent description from the training examples (by the AQ15 program).
2. Determine the *top rule* description from the above description using the TRUNC method (Michalski et al., 1986).

   The *top rule* description consists of a single rule that covers the maximum number of positive examples among all other rules in the complete and consistent description. Such a description is easy to determine, because the AQ15 generates rules together with measures indicating the number of examples covered *totally* and *uniquely* by each rule, which are denoted the t-weight and u-weight of a rule, respectively (see below). In the experiments, one top rule description was generated for positive concept examples, and one for the negative examples (the latter one from a complete and consistent description of the negative examples). An instance was classified as belonging to a concept, if it matched best the top rule description of positive examples, and was rejected if it matched the top rule description of the negative examples. If both descriptions were matched with roughly the same degree, then the instance was classified as "no match." Learning the *top rule* description, and using it with flexible matching, represents a simple, but important version of the two-tiered concept learning approach (Michalski, 1990).
3. Determine an optimized two-tiered description from the complete and consistent description using the TRUNC-SG procedure.
4. Determine descriptions of the given concepts using other methods, specifically, variants of the exemplar-based learning approach, and the decision tree learning program ASSISTANT.
5. Test the performance of all generated descriptions on the testing examples.

To illustrate the difference between the complete and consistent descriptions, the *top rule*, and the optimized descriptions created by POSEIDON, figures below a sample of these descriptions in the Labor Management domain. Figure 3 shows a complete and consistent description produced by AQ15.

In the Figure, t (t-weight) is the total number of examples covered by a rule, and u (u-weight) is the number of examples uniquely covered by the rule.

{[cntrct-dur>1]&[wage_incr_yr2=>3.0%]&[#hlds >10]: (*t =11, u = 11*)} or
{[wage_incr_yr1 > 4.5%]: (*t = 4, u = 4*)} or
{[wage_incr_yr1 > 4%] & [wage_incr_yr2 > 4.0%]: (*t = 1, u = 1*)} or
{[wage_incr_yr1 > 4.5%] & [#hlds > 9]: (*t = 1, u = 1*)} or
{[wage_incr_yr1 > 2%] & [vacation > average]: (*t = 1, u = 1*)}

**::> [Contract Class = acceptable]**

{[wage_incr_yr1=2..4%]&[#hlds<10]&[vacation=AVG]: (*t = 3, u = 3*)} or
{[wage_incr_yr1≤ 4.5%] & [wage_incr_yr2 ≤ 4.0%] & [hlds = 10] & [vacation ≤ AVG]: (*t = 2, u =2*)} or
{[dur = 1] & [wage_incr_yr1 < 4.0%] & [#hlds < 10]& [vacation ≤ AV G]: (*t =1, u =1*)} or
{[wage_incr_yr1 ≤ 4.0%] & [wage_incr_yr2 ≤ 3.0%] & [vacation =AVG]: (*t =1, u = 1*)} or
{[dur =1] & [wage_incr_yr1 ≤ 4.0%]&[vacation ≤ AVG]: (*t =1, u =1*)} or
{[wage_incr_yr1 = 2.0%] & [wage_incr_yr2 ≤ 3.0%]: (*t =1, u =1*)}

**::> [Contract Class = unacceptable]**

*Figure 3.* Complete and Consistent Descriptions Generated by AQ15.

By selecting from each descripion the rule with the largest t-weight, the following *top rule descriptions* were obtained (Figure 4):

***BCR:***

{[dur >1]&[wg_incr_yr2 >3%]&[#hlds >10]: (*t =11, u = 11*)}

**::> [Contract Class = acceptable]**

{[wg_incr_yr1 = 2..4%]&[#hlds < 10]&[vacation=AVG]: (*t =3, u =3*)}

**::> [Contract Class = unacceptable]**

***ICI:*** Flexible matching

*Figure 4. Top Rule Descriptions* Obtained by the TRUNC Method.

By optimizing the complete and consistent description using the TRUNC-SC method, and acquiring the ICI rules from an expert, the following optimized two-tiered description was obtained (Figure 5).

***BCR:***

[wage_incr_yr1 > 4.5%] or
[wage_incr_yr2 > 3.0%] or
[#hlds > 9] or
[vacation > AVG]
::> **[Contract Class = acceptable]**

[wage_incr_yr1 ≤ 4.0%] & [#hlds < 10] or
[wage_incr_yr2 ≤ 4.0%] & [vacation < average] or
[Dur = 1] & [wage_incr_yr1 ≤ 4.0%] or
[wage_incr_yr2 ≤ 3.0%]
::> **[Contract Class = unacceptable]**

***ICI:***

Flexible matching plus deductive matching using rules:
[wage_incr_yr1≥5.5%]&[vacation < average]
::> **[Contract Class = acceptable]**
[wage_incr_yr1 ≤3%] &[wage_incr_yr2 < wage_incr_yr1]
::> **[ContractClass = unacceptable]**
[wage_incr_yr1≤3%]&[wage_incr_yr2≤3%]&[hours_work≥40]&
[pension= empl_contr]
::> **[Contract Class = unacceptable]**

*Figure 5*. Optimized Two-tiered Descriptions Obtained by POSEIDON.

During the BCR description optimization process, the system determined the training events that were incorrectly classified by the base representation. An expert was asked to formulate rules explaining these examples (the ICI rules in Figure 5). For example, the first ICI rule for an unacceptable contract (Figure 5) describes contracts with the wage increase in the first year lower or equal 3%, and an even lower increase in the second year. In such circumstances, the holiday and vacation time do not matter, and the contract is classified as unacceptable (by the union).

As one can see, the optimized BCR descriptions are significantly simpler than the complete and consistent descriptions generated by AQ15.

They also seem to represent the most important characteristics of the labor management contracts. Specifically, a contract is acceptable when it offers a significant wage increase (the first two rules in Figure 5), or it offers many holiday days, or the vacation time is above average.

## Results From Testing POSEIDON and Other Methods

As mentioned earlier, experiments tested POSEIDON and three other methods, specifically, variants of exemplar-based learning, the method for learning consistent and complete descriptions, a method for generating *top rule* descriptions, and a method for generating pruned decision trees. All of these methods were employed to learn a concept description from the same set of training examples. All the learned descriptions were then applied to the same testing examples. The performance was evaluated by counting the number of examples that were classified correctly, incorrectly, or unclassified.

Tables 4 to 7 present the results of different experiments. A summary of all results is shown in Table 8. In all tables, columns "Correct" and "Incorrect" specify the percentage of the testing events that were correctly and incorrectly classified, respectively. The column *No_Match* specifies the number unclassified examples (i.e., the examples that did not match any description to a sufficient degree). To provide an estimate of the complexity of descriptions learned, the tables also list the number of conditions and rules in each description. In the case of pruned decision trees, the table lists the number of nodes and leaves (the number of leaves corresponds to the number of rules that can be directly determined from the decision tree).

Experiment 1 (Table 4) tested a *factual* description, and variants of the exemplar-based approach (1-, 3- and 5- nearest neighbor match). A factual description is a disjunction of all the training events, and, as such, is obviously complete and consistent with regard to the training set. The first part of Experiment 1 tested the factual description on the testing examples using the *strict match* method. In such a method, a testing example must match exactly one of the training examples to be classified.

In this case, obviously, the description had no predictive power. It produced No_Match answers for all testing examples of the labor contract data, and for 96% testing examples of the congressional voting data (two examples were the same in the training and testing sets).

---

***Simple Exemplar-based Description***

Labor management problem (Labor): 27 rules and 432 conditions
Congress problem (Congress): 51 rules and 969 conditions

| | **Correct** | | **No_Match** | |
|---|---|---|---|---|
| | *Labor* | *Congress* | *Labor* | *Congress* |
| ***Strict Match*** | | | | |
| Training Set | 100% | 100% | 0% | 0% |
| Testing Set | 0% | 4% | 100% | 96% |
| *1-Nearest Neighbor* | | | | |
| Training Set | 100% | 100% | 0% | 0% |
| Testing Set | 77% | **86%** | 0% | 0% |
| *3-Nearest Neighbors* | | | | |
| Training Set | 100% | 100% | 0% | 0% |
| Testing Set | **83%** | 84% | 0% | 0% |
| *5-Nearest Neighbor* | | | | |
| Training Set | 100% | 100% | 0% | 0% |
| Testing Set | 80% | 84% | 0% | 0% |

---

*Table 4.* Results of Experiment 1.

Subsequent parts of Experiment 1 tested the factual description using the *k-nearest neighbor* method with different k. The method involved determining k closest (best "fitting") learning examples to the one being classified, and assigning to it the class of the majority of the closest examples. Such a method is equivalent to simple forms of exemplar-based learning. The *1-Nearest Neighbor* row lists results from applying the factual description with a matching method somewhat similar to the one described in (Kibler and Aha, 1987). The only difference was that Kibler and Aha's method uses the *maximum* function for evaluating a ruleset (disjunction), while our flexible matching uses the *probabilistic sum*. The method was also tested with k=3 and 5.

The second experiment used concept descriptions generated by AQ15 without truncation (Table 5). Such descriptions are consistent and complete with regard to the training examples, i.e., they classify all training examples 100% correct when using the strict matching method. The flexible matching method did not change this result.

*Complete and Consistent Description (No truncation)*

Labor-mgmt problem (Labor): 11 rules and 28 conditions
Congress problem (Congress): 10 rules and 32 conditions

| | Correct | | No_Match | |
|---|---|---|---|---|
| | *Labor* | *Congress* | *Labor* | *Congress* |
| ***Strict Match*** | | | | |
| Training Set | 100% | 100% | 0% | 0% |
| Testing Set | 80% | 86% | 3% | 0% |
| ***Flexible Match*** | | | | |
| Training Set | 100% | 100% | 0% | 0% |
| Testing Set | **80%** | **86%** | 3% | 0% |

*Table 5.* Results of Experiment 2.

For the testing set, the number of correct classifications was relatively high (80-86%), the same for the strict and flexible matching methods. Flexible matching made no difference, probably due to two factors. Firstly, the complete and consistent descriptions include many specific rules, leaving little space for the "no match" cases (3%), in which flexible matching could help. Secondly, the descriptions consisted only of disjoint rules, as the program was run using the "disjoint cover" parameter. In such a situation, the "multiple match" cases do not occur, and flexible matching cannot help.

The above results are similar to those obtained in the previous experiment, which used an exemplar-based approach (Table 4). The main difference is that the AQ descriptions are much simpler in terms of the number of rules and the number of conditions involved (11 vs. 27 rules in the labor management problem, and 10 vs. 51 rules in the congress voting problem). The simpler descriptions allow the system to

be more efficient in the recognition mode.

The third experiment (Table 6) tested the *top rule* descriptions determined from the above complete and consistent descriptions. As shown in Table 5, the performance of these rules using flexible matching was comparable to that of the complete and consistent descriptions, as well as factual descriptions (compare with Tables 4 and 5).

***The Top Rule Description*** ***(the TRUNC method)***

Labor-mgmt problem (Labor): 2 rules and 6 conditions
Congress problem (Congress): 2 rules and 6 conditions

| | Correct | | No_Match | |
|---|---|---|---|---|
| | *Labor* | *Congress* | *Labor* | *Congress* |
| ***Strict Match*** | | | | |
| Training Set | 52% | 62% | 48% | 38% |
| Testing Set | 63% | 69% | 30% | 24% |
| ***Flexible Match*** | | | | |
| Training Set | 81% | 75% | 0% | 0% |
| Testing Set | **83%** | **85%** | 0% | 0% |

*Table 6.* Results of Experiment 3.

It may be surprising that the *top rule* descriptions performed better on the testing set than on the training set. This is due to the fact that the training set contained more exceptions than the testing set. The system used the TRUNC method, in which the truncation process removes rules that cover all except the most typical training examples.

The *top rule* descriptions consist of only one rule per concept, and therefore they are significantly simpler than the factual, and consistent and complete descriptions (they use only 2 vs.11 vs. 27 rules in the Labor Management problem, and 2 vs. 10 vs. 51 rules in the Congress Voting problem). It is quite revealing that such simple rules performed as well as much more complex descriptions generated in previous methods. The

fourth experiment (Table 7) tested optimized descriptions generated by POSEIDON, i.e., derived by the TRUNC-SG method. The descriptions were tested using flexible matching alone (*Flexible Match*), and in combination with deductive matching (*Deductive Match*).

*Optimized Description (POSEIDON)*

Labor-mgmt problem (Labor): 9 rules and 12 conditions
Congress problem (Congress): 10 rules and 21 conditions

| | **Correct** | | **No_Match** | |
|---|---|---|---|---|
| | *Labor* | *Congress* | *Labor* | *Congress* |
| ***Strict Match*** | | | | |
| Training Set | 63% | 84% | 37% | 16% |
| Testing Set | 43% | 73% | 54% | 23% |
| ***Flexible Match*** | | | | |
| Training Set | 85% | 100% | 15% | 0% |
| Testing Set | 83% | 92% | 4% | 0% |
| ***Deductive Match*** | | | | |
| Training Set | 96% | 96% | 4% | 0% |
| Testing Set | **90%** | **92%** | 0% | 0% |

*Table 7.* Results of Experiment 4.

For comparison, the performance of these descriptions was also tested using strict match. The latter is rather an impractical combination. As expected, these descriptions used with strict matching gave relatively poor performance.

The optimized descriptions (BCR) combined with deductive matching (ICI) gave the best performance (90-92% correct). When used with only flexible matching, the performance was slightly lower. The descriptions are simpler than complete and consistent descriptions, although they include the Inferential Concept Interpretation rules. They are, of course, more complex than the top rule descriptions, which do not use any interpretation rules.

For the Labor data, descriptions applied with deductive matching produced higher performance than when used with flexible matching only (90 vs. 83%)[6]. For the Congress data problem, the performance was the same for the two matching methods. This is because deductive rules were acquired on the training set; in the specific testing set, the D-covered events were the same as F-covered ones.

Table 8 summarizes the results of experiments, specifically, it compares the performance and complexity of descriptions generated by simple exemplar-based methods, the two-tiered descriptions generated by POSEIDON, and pruned decision trees generated by ASSISTANT (a descendant of the Quinlan's ID3 program; Cestnik et al., 1987).

ASSISTANT was applied to the same learning and training data, which were used in the previous experiments (whose results were presented in Tables 4, 5, 6 and 7.) The decision trees obtained by ASSISTANT were optimized using a tree-pruning mechanism (Cestnik et al., 1987). This mechanism is compared with the TRUNC-SG method in the next section.

The factual description was applied with the flexible matching function. The complexity of a rule-based description was measured by stating the number of rules (#Rules) and the number of conditions (#Conds). The complexity of a decision tree was measured by the number of leaves (#Leaves) and the number of nodes (#Nodes).

---

6 This difference, for the Labor Contract data, is not $\chi 2$ significant. Nevertheless, we think that there are other reasons to prefer deductive matching over flexible matching. Deductive classification is based on rules and knowledge-based inference, and is therefore easier to understand by humans. The rules may be modified locally, while changing the flexible matching function is difficult and produces uncontrolled, global consequences. In other words, examples that are correctly recognized through ICI deductive rules are also explained *ipso facto* in terms of domain knowledge. The same cannot be said of examples correctly recognized by flexible matching, which is a knolwedge-independent distance measure. To reflect this, the GDQ measure assigns a higher score to a description with deductive matching than with flexible matching.

| | *Labor Contract* | *Congress Voting* |
|---|---|---|
| **Simple exemplar-based method** | | |
| ***Performance*** **(%Correct)** | | |
| *1-nearest neighbor* | 77% | 86% |
| *3-nearest neighbor* | 83% | 84% |
| *5-nearest neighbor* | 80% | 84% |
| ***Complexity*** **(#Rules / #Conds)** | 27 / 432 | 51 / 96 |
| **Pruned decision tree (ASSISTANT + PRUNING)** | | |
| ***Performance*** **(%Correct)** | 86% | 86% |
| ***Complexity*** **(#Leaves/ #Nodes)** | 29 / 53 | 19 / 28 |
| **Complete and consistent description (AQ15 without rule truncation)** | | |
| ***Performance*** **(%Correct)** | 80% | 86% |
| ***Complexity*** **(#Rules / #Conds)** | 11 / 29 | 10 / 32 |
| **Top rule two-tiered description (AQ15 with rule truncation )** | | |
| ***Performance*** **(%Correct)** | 83% | 85% |
| ***Complexity*** **(#Rules / #Conds)** | 2 / 6 | 2 / 6 |
| **Optimized two-tiered description (POSEIDON)** | | |
| ***Performance*** **(%Correct)** | 90% | 92% |
| ***Complexity*** **(#Rules / #Conds)** | 9 / 12 | 10 / 21 |

*Table 8*. Summary of the Results of Testing Descriptions Generated by Different Methods.

In the above experiments, for both domain problems, the learning method implemented in POSEIDON produced descriptions that are simpler (except for the *top rule* descriptions), and also perform better on the testing data than other tested methods. Being simpler, these descriptions are also easier to understand, and have a lower evaluation cost. The meaning of the concept defined by such descriptions depends on the base representation (i.e., a TRUNC-SG optimized description learned from examples), and the inferential concept interpretation (consisting of an apriori defined flexible matching procedure and a set of deductive rules, formulated by the expert).

Using rules in the inferential concept interpretation has an advantage that exceptional cases are easy to explain. In the current method, the system determines which examples are exceptional (those that are misclassified by the base representation). The expert analyzes them, and determines the rules for ICI. The top rule descriptions were significantly simpler than any other descriptions, but performed somewhat worse than the optimized description and the decision tree. Depending on the desired trade-off between the accuracy and simplicity, the *top rule* or the optimized description can be taken as the base representation of the concept being defined.

**The Role of Parameters and Related Issues**

POSEIDON has many parameters which can be controlled by a user. On the surface, this might be considered as a disadvantage. In our view, a learning system that allows the user to explicitly modify parameters that affect learning processes (but which are not just method-dependent), is to be preferred over a system that does not explicitly define such parameters. The point is that in the latter systems these parameters are defined only *implicitly*, by the assumptions and the structure of the method. For example, many systems do not take into consideration the typicality of examples. In POSEIDON, this is equivalent to an assumption that the typicality of all examples is equal to the default value 1. As another example, consider the cost of measuring the value of attributes. If a learning program does not have parameters representing such costs, then

this is equivalent to an assumption that all costs are the same (which in reality is often not true). By being able to control such learning parameters, the user can produce results that better fit the task at hand. For example, for some tasks, the accuracy of descriptions may be decisive criterion, while for others the description simplicity may be of equal concern.

An important problem to be investigated is the sensitivity of POSEIDON to its various parameters. While a comprehensive answer to this problem goes beyond the scope of this paper, we report below a preliminary sensitivity analysis regarding the parameters controlling the trade-off between the description accuracy and simplicity. Such parameters are considered to have the most important effect on the performance of learned descriptions. Specifically, they are the *tolerances* in the lexicographic evaluation functional measuring the description quality. To explain their role, let us briefly review the description quality measure. This measure combines several criteria, such as the accuracy, the simplicity, and the cost. Each criterion is associated with a tolerance interval such that differences within this interval are not considered unimportant. Thus, if the tolerance interval of accuracy is very narrow, then the accuracy becomes the prevailing criterion in quality evaluation. On the other hand, if this tolerance interval is wide, the remaining criteria become more significant.

An experiment was performed using the same Congress voting data, as used in experiments reported in Tables 4-7. The training set had 51 examples, while the testing set had 49 examples. The concept to be learned was the voting record of Republicans in the US Congress. The description tested in Table 7, had 10 rules and 21 conditions, and yielded the accuracy of 100% on the training set, and 92% on the testing set. The description was obtained using the accuracy tolerance ($\tau_1$) value equal 0.05. To determine the method's sensitivity to this parameter, the accuracy tolerance $\tau_1$ was set to values 0.55, 0.35, 0.02, 0.005, and for each value the description accuracy was measured. For the above accuracy tolerances, the system's performance on the testing set was 88%, 88%, 90%, and 92%,

respectively. Thus, this experiment seems to indicate that the accuracy of the descriptions slowly grows with the narrowing of the tolerance interval on the accuracy in the description quality measure, which completely confirms an intuitive expectation.

In general, when the accuracy tolerance interval is wide, the simplicity of the description assumes an important role, yielding performances close to the performance of the *top rule* in the two-tiered description. Intermediate values, such as the one used in the experiments presented in Table 7 ($\tau_1 = 0.05$) produced the best results, e.g., the performance of 92% on the testing set from the Congress data. In the case of the narrow tolerance interval for accuracy, the simplicity has a lower impact on the quality of the description. An interesting topic for future research is to systematically investigate the influence of such parameter changes on the performance of the descriptions[7].

Another issue that should be explored more in the future is the role of example typicality of learning examples. In the presented method, if the input examples are assigned typicality values, the generated base concept representation will tend to cover the most typical examples, while the inferential concept interpretation will tend to cover less typical examples. A problem for future investigation is to determine the effect of the typicality on the overall quality of generated concept descriptions. When the typicality information is unavailable, the system itself will assign examples to different classes of typicality. The examples covered by the base representation are classified as typical, those covered by flexible

---

7 In our experiment, for small values of $\tau_1 = 0.02$ and .005, which emphasize the role of accuracy in the measured quality of a description, the performance on the testing set was close or equal to the performance obtained for $\tau_1 = 0.05$, and higher than performance of 86% for AQ15 in Table 7. The reason is that in the last two experiments, as well as in the original experiment in Table 7, it was always possible to find a description that was simpler than the one produced by AQ15, but still 100% correct on the training data. Therefore, by giving more importance to accuracy, the simpler description was preferred, and better performance on the test set was obtained.

matching as nearly-typical, and those covered by the deductive rules as non-typical.[8] An interesting experiment would be to compare such classifications with human classifications. Another interesting issue relates to the noise in the data. The preliminary analysis indicates that the proposed method has a significant ability for handling noisy data. Experiments show that noisy examples are usually covered by the "light" rules, i.e., rules that cover few examples. By removing such rules from the description, the effect of noise can be significantly minimized (Zhang and Michalski, 1989). Future research should investigate these aspects of the method in greater detail.

## RELATED WORK

The research presented here relates to various efforts on learning imprecise concepts, in particular, to learning methods generating pruned decision trees (e.g., Quinlan, 1987; Cestnik, Kononenko & Bratko, 1987; Fisher and Schlimmer, 1988). In these methods, a concept description (or a set of descriptions) is represented as a single tree structure ("one tier") that is supposed to account for all concept instances. An unknown instance is classified by following the nodes of the decision tree from the root to the leaf indicating the class. To avoid overfitting, some parts (subtrees) of the originally generated decision tree are pruned away. As a result, such decision trees do not cover some training examples. Since the recognition process does not use flexible matching, such pruned trees must always produce some error on the training examples, although the overall performance on new examples may increase.

The two-tiered method avoids overfitting by simplifying original descriptions, yielding base concept representations that, in the formal logical sense, are usually also incomplete and inconsistent. The two-tiered method, however, can compensate for the lack of coverage or for an

---

8 This three-way classification of the examples can be viewed as a simple method of learning typicality. A similar feature is available in Cobweb (Fisher, 1987). On the other hand, if the typicality information is available, it is used by POSEIDON to improve the quality of the learned description.

excessive coverage of the first tier (BCR), by the application of the second tier (ICI). This can be done by flexible matching and/or deductive inference rules. The latter ones are normally unaffected by noise, because they depend on a deeper understanding of the domain. In addition, the presented method takes into consideration the typicality of the examples (if it is available). This feature gives the method an additional help for handling noisy examples.

The method presented in (Quinlan, 1987) is based on a hill-climbing approach that first truncates conditions, and then rules. No search is performed, only one alternative truncation is tried at every step. The final result might possibly be far from optimal. By avoiding the search, such a procedure should, however, be significantly faster than the one implemented in POSEIDON. If the speed of learning and the simplicity of descriptions are of central importance, then the TRUNC method (that determines the top rule descriptions without search) should be applied rather than TRUNC-SG. In the same paper (Quinlan, 1987), other methods for pruning decision trees are also described. Some of these methods require a separate testing set for the simplification phase, and others use the same training set that was used in creating the tree. The simplification phase in POSEIDON can also be done either using the original training set, or using a separate set of examples.

The experiments by Fisher and Schlimmer (1988) on pruning decision trees use a statistical measure to determine the attributes to be pruned. Such measures require a rather large data sample, and thus do not apply well to small training sets. In the two-tiered approach, training events are analyzed logically, rather than statistically, both in the phase creating a complete and consistent description, and in the optimization phase. Consequently, the two-tiered approach seems to be more suited for learning from a relatively small number of examples. An interesting possibility for future research is to integrate a statistical measure, such as used by Fisher and Schlimmer, or other, in the process of rule learning and truncating with large data sets.

The system developed by (Iba et al. 1988) uses a trade-off measure

and truncating with large data sets.

The system developed by (Iba et al. 1988) uses a trade-off measure that is somewhat similar to the general description quality (GDQ) measure proposed in this paper. Our GDQ measure considers more factors. Besides taking into account the typicality of the instances covered by the description, it considers different types of matching between an instance and a description. Moreover, the simplicity measured by GDQ depends not only on the number of rules in the description as in (Iba et al., 1988), but also on the different syntactic features in the description.

The inductive algorithm implemented in CN2 uses a heuristic function to terminate search during rule construction (Clark & Niblett, 1989). The heuristic is based on an estimate of the noise present in the data. Such pruning of the search space of inductive hypotheses results in rules that may not classify all the training examples correctly, but that perform well on testing data. CN2 can be viewed as an induction algorithm that includes pre-truncation, while the algorithm reported here is based on post-truncation. CN2 applies truncation during rule generation, and POSEIDON applies truncation after rule generation. The advantage of pre-truncation is efficiency of the learning process. On the other hand, such an approach has difficulty with identifying irrelevant conditions and redundant rules.

The two-tiered method described here can also be viewed as a kind of constructive induction in the sense of (Michalski, 1983). In fact, the whole learned description may include new terms, absent from the examples used for learning. This behavior is also encountered in several other systems (e.g., Sammut and Banerji, 1986; Drastal, Czako & Raatz, 1989). However, constructive learning in POSEIDON is due to the second tier based on domain knowledge characterizing non-typical examples. This is different from using domain knowledge to rewrite or augment the whole training set (e.g., Rouveirol, 1991), or to generate new attributes by a data-driven approach (Bloedorn & Michalski, 1992), or a hypothesis-driven approach (Wnek and Michalski, 1991).

The exemplar-based learning system PROTOS (Bareiss, 1989) is

concept description and acquiring the matching knowledge via explanations of training events provided by a teacher. There are, however, major differences: 1) PROTOS stores exemplars as base concept descriptions, whereas POSEIDON generates simple and easy-to-understand generalizations as base concept descriptions, 2) PROTOS uses domain knowledge in classifying all new cases, whereas POSEIDON uses Inferential Concept Interpretation rules only for classifying exceptions, 3) During the learning process, PROTOS asks the teacher for explanations for all exemplars, whereas POSEIDON only asks for explanations of exceptions.

The problem of using some typicality measure of examples has not so far been given much attention in machine learning, although there were attempts in this direction. For example, Michalski and Larson (1978) introduced the idea of "outstanding representatives" of a concept to focus the learning process on the most significant examples. In cognitive science, the concept of typicality of examples has been studied extensively (e.g., Rosch and Mervis, 1975; Smith and Medin 1981). The concept of two-tiered representation has naturally led us to the proposition of a precise definition of representative, nearly-representative and exceptional examples, namely, as those that are covered by the first tier, the second tier's procedure for flexible matching, and the second tier's inference rules, respectively. The ideas of two-tiered representation are also consistent with recent research on two-stage category construction (Ahn and Medin, 1992).

To summarize, there are several major differences between the method presented and related research described in the literature. First, the method has the ability to recover from the loss of coverage due to the description truncation by using the second tier. Specifically, the procedure of flexible matching or deductive rules are used to cover examples not covered explicitly. As has been demonstrated experimentally, this ability often leads to a significant reduction of concept descriptions, and at the same time, to an improvement of their predictive power. Second, the description reduction is done by independently performing both generalization and

specialization operators. Third, any part of the description may be truncated in the simplification process, not just only specific parts (as, e.g., in decision tree truncation). Fourth, the method is able to take into account the typicality of the examples. Finally, the method uses a general description quality measure, which takes into consideration a number of different aspects of a description.

To relate the presented two-tiered approach to other basic machine learning approaches, Table 9 characterizes them in terms of the type of concept representation used and the kind of matching applied for classification.

| | Simple Induction | Exemplar-based | Two-tiered |
|---|---|---|---|
| **Representation** | General | Specific | General |
| **Matching** | Precise | Inferential | Inferential |

*Table 9.* A comparison of the two-tiered method with simple inductive and exemplar-based methods.

## SUMMARY AND OPEN PROBLEMS

The most significant aspect of the presented method is that it represents concepts in a two-tiered fashion, in contrast to traditional learning methods that represent concepts by a monolithical structure. In this representation, the first tier, the base concept representation (BCR), captures the explicit and common concept meaning, and the second tier, the inferential concept interpretation (ICI) defines allowable modifications of the base meaning and exceptions. Thus, typical concept instances match the BCR, and thus can be recognized efficiently. Such a two-tiered representation is particularly suitable for learning flexible concepts, i.e., concepts that lack precise definition and are context-dependent.

In the POSEIDON system that implements the method, the BCR is learned in two steps. First, a complete and consistent description is learned by a conventional learning program (AQ15). Next, this description is optimized according to a general description quality measure. This is done by a double-level search process that uses both generalization and

optimized according to a general description quality measure. This is done by a double-level search process that uses both generalization and specialization operators. The General Description Quality takes into account not only properties of BCR, but also of ICI. This is done by measuring the complexity and accuracy of the total description.

The ICI has two components: one specifies a flexible matching function, and the second specifies inference rules for handling exceptions and context-dependency. The ICI rules can be of two types. The rules of the first type extend the meaning of the concept, while the rules of the second type contract this meaning. The first type rules are employed when an instance is neither covered by the BCR (not S-covered), nor by the flexible matching function (not F-covered). The second type of rules are used when an unknown instance covers a base representation of more than one concept, or when concept membership has to be confirmed. In both cases, the rules are used deductively. An advantage of using rules for matching over other matching methods is that they can serve as an explanation why a given instance does or does not belong to the concept.

The experimental results have strongly supported the hypothesis that two-tiered concept descriptions can be simpler and easier to understand than "single-tier" descriptions. In the experiments, these descriptions also had greater prediction accuracy, i.e., performed better on new examples. For example, the two-tiered descriptions obtained for the acceptable labor management contracts gave a performance of over 90% correct using only about 9 rules. In contrast, the best performance of a simple exemplar based method gave the 80% correct predictions on new examples, and used 27 rules, and the corresponding pruned decision tree performed at 86%, and had 29 leaves (each of which may be viewed as corresponding to one rule). The system also performed better than the previous method based on the TRUNC procedure in terms of the prediction accuracy (80%), but at the cost of a more complex concept description. In addition, two-tiered descriptions are relatively easy to understand, and can easily represent an explicit domain knowledge.

The presented method is different in several significant ways from the

earlier method of learning two-tiered representations (Michalski et al., 1986). The flexible matching procedure is used not only in the testing phase, but also in the learning phase. In addition to a flexible matching function, the method employs rules for extending or contracting the concept meaning. The earlier TRUNC method used only one specialization operator (rule removal), while TRUNC-SG employed in POSEIDON uses two generalization and two specialization operators. The price for that is that the new method is significantly more complex.

There are many interesting problems for future research. An especially interesting problem is how to integrate the description optimization phase with the initial description generation phase (done by AQ). Another interesting problem is how to learn second tier rules from examples. In the initial method developed by (Plante and Matwin, 1990), the inferential concept interpretation rules are learned by a chunking process in the situations when multiple explanations of positive or negative training events are provided. Future research should also address the application of constructive induction (Michalski, 1983) in the process of learning flexible concepts. In constructive induction, background knowledge is used to construct new attributes and/or higher level descriptors. As a result, produced descriptions can capture the salient features of the concept, and can be simpler and more comprehensible. The ideas of constructive induction seem to be very relevant to the method proposed. For example, through constructive induction the system may be able to fold several rules into a single one, or prevent the removal of relevant rules.

The current system does not address the problem of dynamically emerging hierarchies of concepts. The system only learns one concept at a time, and concepts do not change or split as new examples become available. Another open issue is the ability of the system to reorganize itself. The distribution of knowledge between the Base Concept Representation and the Inferential Concept Interpretation should be determined by the performance of the system on large testing sets. If it turns out, for instance, that some inferential concept interpretation rules

are used very often, then they could be compiled into the base representation. Further research is needed on the role and the importance of different parameters used in the method, and on the trade-offs that they can control.

This paper has focused on learning attributional descriptions that characterize entities by attributes, and ignore their structural properties. Although such descriptions are quite powerful and sufficient for many practical problems, there are applications that require structural descriptions that characterize entities as systems of components, and the relationships among these components. Developing a method for learning two-tiered structural descriptions is therefore an important topic for future research. A relatively solution to the above problem would be to replace the AQ15 program by a version of INDUCE (e.g., Michalski, 1983) for learning the initial complete and consistent description. The basic search procedure would essentially be the same, but would deal with a more complex knowledge representation. A structural representation would allow additional description modification operators, so the descriptions could be modified in more ways, so this would increase the flexibility and the complexity of the search process. Also, the computation of the general quality of descriptions would require proper modification, and flexible matching would need to be extended to handle structural concept descriptions.

As practical problems frequently require only attributional descriptions, and the method is domain-independent, POSEIDON has a potential to be useful for concept learning and knowledge acquisition in a wide range of real-world applications.

## ACKNOWLEDGMENTS

The authors thank Hugo de Garis, Attilio Giordana, Ken Kaufman, Elizabeth Marchut-Michalski, Doug Medin, Franz Oppacher, Lorenza Saitta, Gail Thornburg, and Gheorghe Tecuci for useful comments and criticisms. The authors express special gratitude to Alan Meyrowitz and Susan Chipman for their support of the research described in this chapter.

In addition, Alan Meyrowitz provided detailed comments on this chapter that helped in the prepartion of the final version. The authors thank Zbig Koperczak for his help in acquiring the data used in the experiments.

This research was done in the Artificial Intelligence Center of George Mason University. The research activities of the Center are supported in part by the Defense Advanced Research Projects Agency under the grants administered by the Office of Naval Research, No. N00014-87-K-0874 and No. N00014-91-J-1854, in part by the Office of Naval Research under grants No. N00014-88-K-0397, No. N00014-88-K-0226, No. N00014-90-J-4059, and No. N00014-91-J-1351, and in part by the National Science Foundation under the grant No. IRI-9020266. The second author was supported in part by the Italian Ministry of Education (ASSI), and the third author was supported in part by the Natural Sciences and Engineering Research Council of Canada.

## REFERENCES

Ahn, W. & Medin, D.L. (1992). A Two-stage Model of Category Construction, *Cognitive Science*, 16, pp. 81-121.

Bareiss, R. (1989). *An exemplar-based knowledge acquisition*. Academic Press.

Bergadano, F. & Giordana, A. (1989). Pattern classification: an approximate reasoning framework.*International Journal of Intelligent Systems*.

Bergadano, F., Matwin, S., Michalski, R.S. & Zhang, J. (1988a). Learning flexible concept descriptions using a two-tiered knowledge representation: Part 1 - ideas and a method. *Reports of Machine Learning and Inference Laboratory*, MLI-88-4. Center for Artificial Intelligence, George Mason University.

Bergadano, F., Matwin, S., Michalski, R. S. & Zhang, J. (1988b). Measuring quality of concept descriptions, *Proceedings Third European Working Sessions on Learning*. Glasgow, 1-14.

Bloedorn, E. & Michalski, R.S. (1992). Data-driven constructive induction AQ17: A method and experiments. *Reports of Machine Learning and Inference Laboratory*, Center for Artificial Intelligence, George Mason University (to appear).

Cheeseman, P., Kelly, J., Self, M., Stutz, J., Taylor, W. & Freeman, D.(1988). AutoClass: A Bayesian classification system. *Proceedings of the Fifth International Conf. on Machine Learning*, Ann Arbor, 54-64.

Cestnik, B., Kononenko, I., Bratko, I. (1987). ASSISTANT 86: A knowledge-elicitation tool for sophisticated users. *Proceedings of the 2nd European Workshop on Learning*. 31-45.

Clark, P. Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning Journal*, Vol. 3, No.4, 261-183.

Collins, A. M., Quillian, M. R. (1972).Experiments on semantic memory and language comprehension" in *Cognition, Learning and Memory*, L. W. Gregg (Ed.), John Wiley.

DeJong, G., Mooney, R. (1986) Explanation-based learning: An alternative view. *Machine Learning*, Vol. 1. No. 2.

Dietterich, T.. (1986). Learning at the knowledge level. *Machine Learning*, Vol. 1. No. 3. 287-315.

Dietterich., T., Flann, N. (1988). An inductive approach to solving the imperfect theory problem. *Proceedings of the Explanation-based Learning Workshop*, Stanford University. 42-46.

Drastal, G., Czako, G., Raatz, S. (1989). Induction in an abstraction space: A form of constructive induction. *Proceedings of IJCAI 89*, Detroit. 708-712.

Fisher, D. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*. Vol. 2, 139-172.

Fisher, D. H. & Schlimmer, J. C. (1988). Concept simplification and prediction accuracy. *Proceedings of the Fifth Int'l. Conf. On Machine Learning*. Ann Arbor, 22-28.

Hammond, K. (1989). Case-based planning: Viewing planning as a memory task. *Academic Press.*

Iba, W., Wogulis, J. & Langley, P. (1988). Trading off simplicity and coverage in incremental concept learning. *Proceedings of the Fifth Int'l. Conf. on Machine Learning*. Ann Arbor, 73-79.

Kedar-Cabelli, S. T. & McCarthy, L. T. (1987).Explanation-based generalization as resolution theorem proving. *Proceedings of the 4th Int. Workshop on Machine Learning*, Irvine.

Kibler, D. & Aha, D. (1987). Learning representative exemplars of concepts. *Proceedings of the 4th International Workshop on Machine Learning*, Irvine.

Kolodner, J. (Ed.) (1988). *Proceedings of the Case-based Reasoning Workshop*, DARPA, Clearwater Beach, FL.

Lakoff, G. (1987).*Women, Fire, and Dangerous Things: What Categories Reveal about Mind*, University of Chicago Press.

Lebowitz, M. (1987). Experiments with incremental concept formation: UNIMEM, *Machine Learning Journal*, Vol. 2, No. 2.

Michalski, R.S. (1975). Variable-valued logic and its applications to pattern recognition and machine learning. In D. C. Rine (Ed.), *Computer science and multiple-valued logic theory and applications*, North-Holland Publishing Co. 506-534.

Michalski, R.S. & Larson, J. B. (1978). Selection of most representative training examples and incremental generation of VL1 hypotheses: The underlying methodology and the description of programs ESEL and AQ11. *Reports of the Department of Computer Science*, TR 867, University of Illinois at Urbana-Champaign.

Michalski, R.S. (1983). A theory and methodology of inductive learning. In R.S. Michalski J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach*. Palo Alto, CA: Tioga (now Morgan Kufmann).

Michalski, R.S. & Stepp, R.E. (1983). Learning from observation: conceptual clustering. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach . Palo Alto, CA:* Tioga (now Morgan Kaufmann).

Michalski, R. S., Mozetic, I., Hong J. & Lavrac, N. (1986). The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. *Proceedings of the 5th AAAI*. 1041-1045.

Michalski, R. S.(1989). Two-tiered concept meaning, inferential matching and conceptual cohesiveness. In S. Vosniadou & A. Ortony (Eds.),*Similarity and analogy*, Cambridge: Cambridge University Press.

Michalski, R. S. & Ko, H. (1988). On the nature of explanation, or why did the wine bottle shatter? *AAAI Symposium: Explanation-Based Learning*, Stanford University. 12-16.

Michalski, R. S. (1987). How to learn imprecise concepts: A method employing a two-tiered knowledge representation for learning. *Proceedings of the Fourth International Workshop on Machine Learning*, Irvine, CA. 50-58.

Michalski, R. S. (1990). Learning flexible concepts: fundamental ideas and a methodology in Y. Kodratoff and R. S. Michalski (Eds.) *Machine Learning: An artificial intelligence approach, Vol. III.* San Mateo, CA: Morgan Kaufmann Publishers.

Mooney, R. & Ourston, D. (1989). Induction over the unexplained: integrated learning of concepts with both explainable and conventional aspects. *Proceedings of 6th Int'l Workshop on Machine Learning*, Ithaca, NY, 5-7.

Minsky, M. (1975). A framework for representing knowledge. In P. Winston (Ed.), *The Psychology of computer vision.*

Mitchell, T. M., Keller, R. & Kedar-Cabelli, S. (1986) Explanation-based generalization: A unifying view. *Machine Learning Journal,* Vol. 1. No. 1, 11-46.

Mitchell, T. M. (1977). Version spaces: an approach to concept learning. Ph.D. Dissertation, Stanford University.

Plante, B. & Matwin, S. (1990). Learning second tier rules by chunking of multiple explanations. *Research Report*, Department of Computer Science, University of Ottawa.

Prieditis, A. E. & Mostow, J. ( 1987). PROLEARN: Towards a Prolog interpreter that learns. *Proceedings. of IJCAI 87*, Milan. 494-498.

Quinlan, J. R.. (1987) Simplifying decision trees. *Int. Journal of Man-Machine Studies*. Vol. 27, 221-234.

Robinson J. A. & Sibert E. E. (1982). LOGLISP: An aternative to Prolog. *Machine Intelligence,* Vol. 10, J. E. Hayes & D. Michie (Eds.), 399-419.

Rosch, E. & Mervis, C. B. (1975). Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology*, Vol. 7, 573-605.

Rouveirol, C. (1991). Deduction and semantic bias for inverse resolution. *Proceedings of IJCAI 91*. Sydney, Australia.

Sammut, C. & Banerji, R.B. (1986). Learning concepts by asking question. in R.S. Michalski, J. G. Carbonell & T.M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.* Palo Alto, CA: Tioga (now Morgan Kaufmann Publishers).

Smith, E. E. & Medin, D. L. (1981). *Categories and concepts.* Harvard University Press.

Sowa, J. F. (1984). *Conceptual sructures.* Addison Wesley.

Sturt, E. (1981). Computerized construction in Fortran of a discriminant function for categorical data. *Applied statistics*, Vol. 30, 213-222.

Watanabe, S. (1969). *Knowing and guessing, a formal and quantitative study*. John Wiley.

Weber, S. (1983). A general concept of fuzzy connectives, negations and implications based on t-norms and t-conorms," *Fuzzy sets and systems*, Vol. 11, 115-134.

Winston, P. H. (1975). Learning structural descriptions from examples," in P. Winston (Ed.) *The Psychology of computer vsion.*, McGraw-Hill.

Wnek, J. & Michalski, R.S. (1991). Hypothesis-driven constructive induction in AQ17: Method and experiments. *Reports of Machine Learning and Inference Laboratory*, Center for Artificial Intelligence, George Mason University.

Zadeh, L. A. (1974). Fuzzy logic and its applications to approximate reasoning. *Information Processing*. North Holland, 591-594.

Zhang, J. & Michalski, R. S. (1989). Rule optimization via SG-trunc method. *Proceedings of the Fourth European Working Sessions on Learning*. Glasgow.