Breaking the Primitive Concept Barrier

Robert Kass (kass@cis.upenn.edu)*
Ron Katriel (katriel@cis.upenn.edu)
Tim Finin (tim@cis.upenn.edu)

Department of Computer and Information Science/D2
Moore School of Electrical Engineering
University of Pennsylvania
Philadelphia, PA 19104

Abstract

This paper addresses a fundamental trade-off which exists in knowledge representation languages between maintaining the integrity of knowledge bases, and ease of knowledge acquisition. Systems which use a *classifier* (such as KL-ONE) do a good job of maintaining knowledge base integrity, but make it difficult to add new knowledge. An *interactive classifier* is a means of easing the knowledge acquisition process while still maintaining the integrity of the knowledge base. However, the presence of primitive concepts in the knowledge base makes automatic classification error-prone and interactive classification tedious. In this paper we discuss adding a *definitional* component to a KL-ONE-like knowledge base which greatly reduces the number of primitive concepts in the knowledge base and significantly enhances interactive classification.

1 Introduction

A conflict exists among knowledge representation systems. Systems in which it is easy to add new knowledge tend to be overly accepting—they allow new knowledge which is inconsistent with the current knowledge in the knowledge base (KB). On the other hand, adding new concepts using a knowledge representation system which does ensure the integrity of the knowledge base can be very tedious and error-prone for the knowledge engineer. Our work has focused on methods for reducing this conflict. Our goal is a knowledge representation which guarantees its own integrity, yet facilitates the knowledge engineering process.

*This research is supported in part by U.S. Army Research Office grant DAAG-29-84-K-0061, Defense Advanced Research Projects Agency grant ONR-N00014-85-K-0807 and a grant from the Digital Equipment Corporation.

A group of knowledge representation systems exist which maintain the consistency of the knowledge base via a *classification* process which is applied to each new concept added to a knowledge base. The first such system was KL-ONE [4], while later systems such as NIKL [9], KL-TWO [12] and KRYPTON [3] have evolved from the initial ideas of KL-ONE. In these systems, a new concept is added to the KB by providing a *definition* of the concept. The classifier ensures that the new concept is consistent with the existing knowledge in the KB before the new concept is added to the KB.

Unfortunately, constructing full definitions for each new concept can be quite tedious. One means of escaping this knowledge acquisition problem is to use an *interactive classifier* which was first proposed by Finin and Silverman [5,6,11]. The basic idea is to allow the knowledge engineer to provide an incomplete definition of new concepts to the system. The classifier will begin the process of classification as usual, but as it encounters possible attributes which may be missing from the definition it will stop and ask the user whether these attributes should be included in the definition. Thus the system and the user actually *negotiate* the definition of new concepts in the KB.

Interactive classification has a problem with *primitive* concepts, however. Primitive concepts are concepts in the knowledge base which do not have complete definitions. Thus any classifier (whether interactive or automatic) cannot deal with the concept solely on the basis of the definition. The interactive classifier must resort in most cases to asking the user whether the new concept is consistent with existing primitive concepts in the knowledge base. As the knowledge base grows the number of questions posed to the user during the classification of one new concept quickly becomes excessive. Thus primitive concepts form a barrier to classification.

In this paper we discuss an extension to KL-ONE which greatly reduces the burden on the user when

CH2408-3/87/0000/0066$01.00 © 1987 IEEE

adding new concepts to a knowledge base, while maintaining the soundness of the knowledge representation language. This extension consists of adding an explicit definitional component to concepts in the knowledge base. Within this component the strictness of concept definitions is itself relaxed. The benefits of this modification are threefold:

- The relaxed form of definitions reduces the number of primitive concepts in a knowledge base.
- The explicit definitional component can help the classifier deal with concepts that do not have complete definitions.
- The definitional component enables interactive classification to be an effective knowledge acquisition technique for KL-ONE-like systems.

In the remainder of this paper we shall first look in more detail at KL-ONE (as a representative of classification-based knowledge representation systems) and at interactive classification. We then proceed to discuss the extension to KL-ONE, why it is warranted, and consider its effects on both automatic and interactive classification.

2 An Overview of KL-ONE

KL-ONE really consists of two sublanguages — a *description language* and an *assertion language*. The description language is intended to capture the *intension* of a concept, without reference to actual entities which may exist. Thus concepts which would be represented in the description language include "elephant", "unicorn" and "debt security". On the other hand, the assertion language represents extentional facts, facts about entities in the world. Thus statements like "Clyde is an elephant in the circus" would be represented in the assertional language of KL-ONE. This paper will only be concerned with the description language portion of KL-ONE.

2.1 The Description Language

The description language of KL-ONE allows one to form a variety of descriptive terms out of other descriptive terms using a small set of description-forming operators. This seemingly endless recursion is avoided by the introduction of *primitive concepts*. Primitive concepts are used to represent concepts for which we cannot provide complete definitions (such as "person"), or for concepts for which we may not wish to provide a complete definition in a particular domain (such as "integer").

KL-ONE organizes concepts defined in the description language into a taxonomic structure defined by the *subsumption* relationship.[1] Informally, concept A subsumes concept B if every instance of B is also an instance of A. For example, the concept "equity" subsumes the concept "stock" because all stocks are equities. A concept's meaning is strictly determined by its subsuming concepts (*superConcepts*) and its local structure. The taxonomy usually indicates only direct subsumption relations. This makes the notation more readable, while permitting one to "read off" the implicit relations using the transitivity of subsumption. This mechanism is called *inheritance*. Each concept may have local information consisting of *Roles*, which describe potential relationships between instances of the Concept and other closely associated Concepts, and *RoleSet Relations*, which express the interrelations among the functional roles [2].

2.2 Classification

New concepts are inserted in the taxonomy by the *classifier*. The classification process consists of finding the correct location in the taxonomy for the new concept. (If a location cannot be found, the new concept must be inconsistent with the knowledge already defined and will be rejected.) The classification procedure consists of two steps. First, the classifier finds the *most specific subsumers* of the new concept. This is done by checking which existing concepts in the knowledge base subsume the new concept. Rather than compare every concept in the knowledge base with the new concept, the classifier starts by checking if the most general concept in the KB (usually called "THING") subsumes the new concept. If so, the new concept is compared with each immediate subsumee of "THING". This process is applied recursively, "pushing down" the new concept in the taxonomy until it can go no further.

The second stage in the classification process is to determine which existing concepts in the KB are subsumed by the new concept. Since the new concept has already been pushed down as far as it can go, the only candidates which may be subsumed by the new concept are its siblings[2] or their subsumees. Once again the process is recursive. If a sibling is subsumed by the new concept the sibling is pushed down to reflect that subsumption, otherwise the new concept is compared with the immediate subsumees of the sibling, and so on.

[1] The taxonomy is actually a *lattice*, with the subsumption relation defining a partial ordering on the concepts.

[2] Two concepts are siblings if they share a common immediate subsumer in the taxonomy.

2.3 Expressibility and Primitive Concepts

KL-ONE uses a formal notion of a definition when specifying defined concepts. Thus all conditions in the set of sufficient conditions defining a concept C are necessary conditions. This notion of a definition is very restrictive when attempting to define "real world" or "natural kind" terms. The problem with such terms is that the attributes which we associate with a given natural kind concept (such as "chair" or "elephant") are seldom necessary.

The notion of a primitive concept was introduced in KL-ONE to handle the problem of defining natural kind concepts. Primitive concepts are ones in which the conditions specified are necessary, but taken together are not sufficient to give a definition. Thus primitive concepts can be viewed as concepts whose conditions are not completely specified.

Unfortunately, primitive concepts introduce a classification problem. An automatic classifier can never push a new concept X below an existing primitive concept P unless X is explicitly defined to be a kind of P, since the incomplete specification of P may include attributes of P which X does not have. In realistic knowledge bases as many as half or more of all concepts may have to be primitive. When defining new concepts for the knowledge base, the user must know the relationship of each new concept to all primitive concepts already defined. But again, this means the user is *manually* classifying the concepts, rather than relying on the automatic classifier to discover the subsumption relationships. This failure of automatic classification in KL-ONE is contrary to the whole spirit of its design.

3 Interactive Classification

3.1 Motivation

Usually, for a classifier to correctly place a new description in a knowledge base taxonomy, the description must be completely specified. As it turns out, this requirement is not very practical. It is likely that the designers, creators and maintainers of a knowledge-based system will be different people. Each may need to be reminded of the contents of portions of a large knowledge base, and prevented from introducing inconsistencies. A user might actually want to provide an incomplete description of a concept to the classifier. He or she might want to avoid having to specify all of the description's attributes. It would be helpful if the classifier could then "negotiate" a definition for the concept with the user. This is the motivation for *interactive classification* [6,5].

3.2 An Overview of Interactive Classification

The idea behind interactive classification is to provide the user of the classifier with immediate feedback, while enabling the system to assume as much of the burden of classification as possible. It is important that the interactive classifier does not ask unnecessary questions of the users of the system, lest they lose their patience and stop using the system. To achieve this last goal the classifier should take advantage of the structure of the taxonomy. Especially useful is disjointness and cover information. The original work on interactive classification [11] was done in the context of a simple tree-structured taxonomy. Given any two concepts in this taxonomy, either one subsumes the other or they are mutually exclusive.

Two very useful complementary heuristics were developed to minimize asking unnecessary questions of the user: classification using *exclusion* and classification using *attribute profiles*. The first technique eliminates potential subsumers by letting the user reject all candidates except possibly one (in a tree, every node, except the root, has exactly one immediate subsumer). The second approach is to make a guess for the best candidate among all the possible consistent subsumers and verify it with the user.

3.3 Problems with Interactive Classification

One of us (Katriel) has attempted to extend the interactive classifier to a full KL-ONE-like language called KLASSIC (KL-one Associated Interactive Classifier), only to encounter significant problems. With the disjointness restriction on the structure of the taxonomy removed, the interactive classifier in KLASSIC is forced to check subsumption between the new description and *most* of the concepts in the knowledge base. Since in most cases the subsumption will fail, the interactive classifier has to choose between two lines of action: it can give up, backtracking up the taxonomy, or it might assume that the reason for this failure is the incompleteness of the new description. If the interactive classifier chooses to backtrack, it could be wrong—the user may simply have failed to give a complete description for the concept. If the classifier chooses the latter, it will have to query the user about each of the missing features of the new description (unless an inconsistency is found first). KLASSIC used the second approach and produced questions like "Does an integer have wheels" when

attempting to classify a small knowledge base about kinds of bicycles. Needless to say, such behavior is not what we are after.

Primitive concepts also present a problem for interactive classification. Since primitive concepts have only necessary conditions, the user must sanction *every* subsumption relation that the classifier finds between a subsuming primitive concept and the new description. Actual knowledge bases built using a KL-ONE-like language tend to have a large number of primitive concepts. In building a knowledge base about financial securities here at the University of Pennsylvania we have found that over half of the concepts defined in the knowledge base are primitive concepts. Clearly, for interactive classification to be effective in a full KL-ONE-like language, additional information is needed about the concepts to guide the classifier.

4 An Extension to KL-ONE

4.1 The Definitional Component

The additional information we propose to add is an explicit *definitional* component for each concept. This definitional component has the form

$$def(X) = N_1 \wedge N_2 \wedge \cdots \wedge N_k \wedge D_1 \wedge D_2 \wedge \cdots \wedge D_l$$

The N_i are necessary conditions for something being an X. The D_i-terms[3] represent disjunctions of sets of contingent conditions (i.e., non-necessary conditions) and have the form

$$D_i = S_{i1} \vee S_{i2} \vee \cdots \vee S_{in}$$

The S-terms consist of conjunctions of contingent conditions or C-terms. An N-term or C-term may be either a simple term or a reference to another concept's definition[4]. A simple term is akin to a role and its value restriction in KL-ONE. Our notation also allows for a negated simple term, whether it is a necessary or contingent attribute. Finally, a D-term can be a covering disjunction for the concept it defines. This last important notion will be described in detail in section 4.2. For the rest of this section it is assumed that all disjunctions are coverings.

Consider the following hypothetical example of the definition of a concept X:

$$def(X) = N_1 \wedge N_2 \wedge N_3 \wedge ((C_1 \wedge C_2) \vee (\neg C_1 \wedge C_3))$$

In this case N_1, N_2 and N_3 are necessary conditions for X. In addition to these conditions C_1, C_2 and C_3 play a role in the definition of X, but are not in themselves necessary. In fact (together with N_1, N_2 and N_3) the clauses $(C_1 \wedge C_2)$ and $(\neg C_1 \wedge C_3)$ form two sets of sufficient conditions for being an X.

The addition of a definitional component to each concept is a departure from the traditional notion of a definition used in knowledge base systems such as KL-ONE. In these representations, all members of the set of sufficient conditions for being a concept are also necessary. We are relaxing this restriction so that contingent conditions can play a role in the definition of a concept. This extension has both a practical and an epistemic basis.

4.2 Practical Justification

The presence of a definitional component for each concept greatly enhances the ability of a classifier and increases the usefulness of the knowledge base for reasoning. A definition provides a complete characterization for a concept, hence classification can be based on the definitions themselves. Since the requirements for what constitutes a definition have been relaxed, concepts which are considered primitive in a KL-ONE type formalism can now be defined to include sets of sufficient conditions in which the individual conditions are contingent rather than necessary.

Consider an example from the financial investments domain:

$$\begin{aligned} &def(DebtSecurity) = \\ &\quad Security \wedge \\ &\quad FaceValue : Currency \wedge Maturity : Time \wedge \cdots \wedge \\ &\quad (CorporateSecurity \vee GovernmentSecurity) \wedge \\ &\quad (Bond \vee MoneyMarketSecurity) \end{aligned}$$

The fact that a debt security is either a bond or a money market security is important in defining the concept of a debt security. Such information would be excluded from a KL-ONE type definition.

The addition of a definitional component to concepts can greatly reduce the number of primitive concepts in a knowledge base. This in turn means the classifier is more capable of doing its job. Moreover, as we shall see in the next section, the definitional

[3] Note that a definition may have several D-terms, each representing a range of possible attributes. For example:

$$\begin{aligned} &def(employee) = \\ &\quad person \wedge \cdots \wedge \\ &\quad (EmplStatus : FullTime \vee EmplStatus : PartTime) \wedge \\ &\quad (pay : salaried \vee pay : hourly \vee pay : commissioned) \end{aligned}$$

[4] Referencing another definition is simply a matter of convenience. The fully expanded form could be substituted for the reference.

component will in some cases allow the classifier to place new concepts below a primitive concept without the user making the relationship explicit in the definition.

External reasoning processes can also take advantage of the explicit definitional component. For example, our proposal facilitates generating natural language descriptions of concepts. The information stored in a concept's definition can in some cases be represented in KL-ONE by defining additional subconcepts and using *cover* and *disjointness* information. However, this distributes the definition over a number of concepts in the taxonomy. To generate a concept description the system must circumscribe the knowledge relevant to that concept [8]. The presence of a definitional component greatly reduces the computation required to perform this task.

4.3 Epistemic Justification

The epistemic justification for relaxing the definitional component has a philosophical origin. Since the earliest Greeks, philosophers have been concerned with expressing the true essence or "meaning" of things which exist in our world such as trees, chairs or people. Attempts to give these *natural kind* terms definitions, which have the same form as definitions for mathematical concepts, do not seem to work, since natural kind terms seem to have very few properties which are essential to them. Current opinion among philosophers generally follows two basic lines[10]. One group claims that natural kind terms do not have definitions at all, and the meaning of such concepts must be found in another way. The other group still believes some form of definition for natural kind terms is possible, but the exact form of such a definition is a matter for debate.

Any knowledge representation formalism which hopes to represent "real world" concepts must take a stand on how natural kind concepts are handled. KL-ONE follows those who believe natural kind terms do not have definitions, and hence introduces the notion of a primitive (undefined) concept in its network to represent such terms.

Our formalism leans towards the second approach. One popular idea among those who believe that natural kind terms do have definitions is that of a "cluster concept". This idea stems from the observation that individual properties of a given natural kind concept X may not be essential to that concept, but taken as a group *most* of those properties must hold for us to call something an X. Our formalism implements a version of this idea by allowing the user to define which sets of properties are sufficient for us to call something an X. The practical benefits noted above make it a very attractive approach for knowledge based systems.

5 Extension Implications

5.1 Subsumption and Inheritance

Determining subsumption relations between concepts having definitional components is somewhat more difficult than it is in KL-ONE. Concept A subsumes concept B if and only if $def(B) \Rightarrow def(A)$[5]. In KL-ONE this reduces to checking that B has every role and constraint that A has, and that any further information on B is consistent with A. Since our definitional component allows concepts to be defined using sets of contingent conditions, checking subsumption relations between concepts in our scheme requires more work—a theorem prover will be needed.

Inheritance is straightforward. The definitions themselves are inherited in the hierarchy. Thus the definition of a given concept is the conjunction of the local information about the concept and the definitions of all ancestors of that concept. This potentially large logical expression can be simplified and converted into the form presented in the previous section. Properties are necessary if they occur in the top level conjunction of the definition, otherwise they are contingent properties. Thus inheritance of necessary properties behaves just as it would in KL-ONE, while additional information can be inherited as well. In fact, through inheritance of definitions some properties which had been contingent properties to all ancestors may become necessary properties for some subsumee. For example, suppose that the concepts A and B are defined as follows

$$\begin{aligned} def(A) &= N_1 \wedge (C_1 \vee C_2) \\ def(B) &= N_2 \wedge (C_1 \vee \neg C_2) \end{aligned}$$

and we know that both A and B subsume X. Then

$$def(X) = N_1 \wedge N_2 \wedge (C_1 \vee C_2) \wedge (C_1 \vee \neg C_2)$$

which reduces to

$$def(X) = N_1 \wedge N_2 \wedge C_1$$

Thus the property C_1 is a necessary property for the concept X, although it was a contingent property on both of X's subsumers. Note that in standard KL-ONE it would be more difficult for a classifier to deduce that X necessarily has attribute C_1 from the knowledge that X is subsumed by both A and B (See figure 1).

[5] "$\Rightarrow$" stands for logical implication

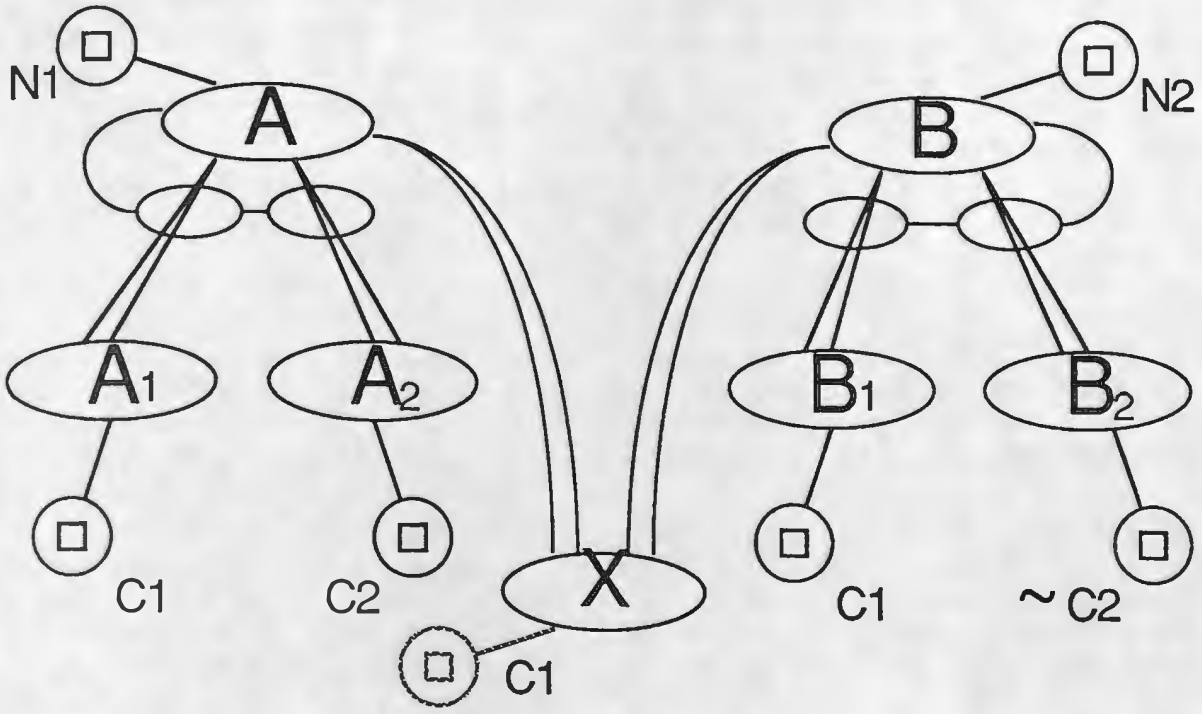

Figure 1: $def(X) = N_1 \wedge N_2 \wedge (C_1 \vee C_2) \wedge (C_1 \vee \neg C_2)$

5.2 Completeness and Covers

Up to this point the examples have assumed that the definition given has been a *complete definition*. For a concept X which has a complete definition, every instance of a thing in the world which we would call an X satisfies the definition of X. In a complete definition, the sets of contingent conditions expressed in the definition (the D-terms) have *covered* the concept. This means any individual satisfying the definition satisfies at least one of the members in each of the sets of covering properties. In the remainder of this section we will explore the use of definitions in which not all of the D-terms are coverings. We shall call such definitions *incomplete definitions*. A concept which has an incomplete definition is an *incomplete concept*.

A definition is incomplete if one or more sets of contingent conditions used to obtain a sufficient characterization of the concept are missing. Thus if we had specified to a knowledge base that

$$
\begin{aligned}
&def(elephant) = \\
&\quad mammal \wedge \\
&\quad ((nose : trunk \wedge size : huge \wedge color : gray) \vee \\
&\quad (nose : trunk \wedge ears : large \wedge tusks : ivory) \vee \\
&\quad (tusks : ivory \wedge size : huge \wedge color : gray))
\end{aligned}
$$

then the disjunction of conditions would not be complete. For example, we would expect a huge, pink mammal with a trunk to be an elephant, but such creatures are not accounted for in the definition.

Another way to view an incomplete concept is to consider it a special case of a complete concept. The incompleteness of a disjunction would be represented by including a special primitive concept[6] in the disjunction. This concept can be thought of as representing all the exceptions which we didn't account for in the disjunction.

Classification in a knowledge base hierarchy containing incomplete concepts has a peculiarity beyond those described earlier. The pink elephant example above illustrates this. When checking to see if a new concept X is subsumed by an existing incomplete concept Y, the subsumption may fail for two reasons.

1. X is inconsistent with one or more of the necessary properties or covered sets of contingent properties (covered D-terms) in the definition of Y.

2. X does not meet any of the criteria in a non-covering set of contingent attributes in the definition of Y.

In the first case we know that Y does not subsume X. In the second case, X may simply not be a kind of Y

[6] in the sense of KL-ONE, that is, a concept which hasn't a definition even in our extended formalism

or X may satisfy a constraint which is missing from the definition of Y. With an automatic classifier such a subsumption will fail. On the other hand, using an interactive classifier, the system and the user may explore the possibility that Y does subsume X and even extend the set of conditions in the definition of Y to reflect this fact. Classifying a new, incomplete concept is no different than classifying a completely defined concept.

Incomplete concepts in our system play a role analogous to primitive concepts in KL-ONE, but with significant differences. In KL-ONE, the primitive concepts have definitions which are not complete. The implication is that these concepts have other (necessary) properties which have not been specified. Definitions for incomplete concepts in our system also lack some information, but what is lacking is some combination of contingent conditions—***not*** necessary conditions. Thus the classifier is still able to automatically classify items below an incomplete concept in many cases, something which cannot be done in KL-ONE without the user explicitly sanctioning the subsumption.

Incomplete concepts are in a sense a concession to those who believe natural kind terms do not have definitions. Even if one holds this view, natural kind concepts may still be specified using incomplete definitions. In practice it is not hard to give characterizations of most natural kind concepts with a fairly small set of conditions. Doing so will allow the classifier to attempt classification of new concepts below these natural kind concepts. This will suffice in most cases to enable the classifier and any external inferencing mechanism to treat incomplete concepts in the same way complete concepts are.

5.3 Defaults and Exceptions

Much attention has been given to ways to represent default and exceptional information in a hierarchical knowledge base and still keep a consistent semantics for subsumption and inheritance. Our formalism is not intended to deal with this problem, but does affect it in some ways. Default and exceptional properties of concepts by their nature are not necessary. Consequently they cannot be inherited in the lattice in the same way that necessary properties are. Our formalism, on the other hand, bases subsumption and inheritance on concept definitions which can include contingent properties. Thus default or exceptional properties which are used in the definition can be inherited *via the definition*. However, the formalism makes no distinction among the contingent properties as to which are defaults and which are exceptions—nor should it. Information about whether certain properties are typical or unusual may help guide classification of concepts or reasoning with concepts in the knowledge base, but it should not be a factor in the definition of those concepts.

5.4 Benefits for Interactive Classification

The proposed extension makes it much easier for an interactive classifier to determine subsumption on its own. Assuming the creators of the knowledge base take full advantage of the extended definitional capability for concepts, the classifier should be able, in many cases, to find a sufficiency set which fits the new description. Even if a perfect fit cannot be found, the classifier can look for the best matching set.

In the case of primitive concepts, the gain is even greater. Whereas the interactive classifier previously had to check *every* primitive concept with the user, it can now autonomously decide about subsumption in many of the cases. The user will be called upon only in cases where the new description could be a *new exception*. If the user sanctions the exception, it will be reflected as a change to the contingent features of the definition of the existing subsuming concept.

6 Future Work

A classifier which includes our proposed extension has not been built yet. We intend initially to modify the classifier of NIKL (the New Implementation of Kl-ONE) [9] to take advantage of the definitional component of concepts. Even without interactive classification this modification should make knowledge acquisition for NIKL an easier process. We plan to reimplement the financial securities knowledge base in this revised NIKL. Using the proposed extension we will take advantage of disjointness and cover information in implementing an interactive classifier for NIKL. Cover information occurs in the definitions as covering disjunctions, while disjointness allows the classifier to stop exploring the subsumees of a concept once one of them is found to subsume the new description being classified.

Viewing the disjunction in the definition as a complementary operator to conjunction (*join* vs. *meet*), it seems natural to introduce a *bottom concept* into the taxonomy (which might be named NOTHING), and perform classification top-down from THING and bottom-up from NOTHING *simultaneously*. Alternating between the two will enable the interactive classifier to prune the search space (this is akin to an

expert system which combines data driven and goal driven inferencing), thus minimizing the necessary interaction with the user.

Finally, we are exploring the notion of a programming language based on the calculus of type subsumption [1]. Initial research indicates that our extension to KL-ONE has many similarities to Ait-Kaci's work. We would like to pursue these connections.

7 Conclusion

We have presented an extension to KL-ONE which maintains its soundness and greatly reduces the burden on the user during knowledge acquisition. We have discussed some of the limitations of KL-ONE-type languages, with emphasis on the primitive concept problem. As a solution, we have proposed adding an explicit definitional component to concepts and relaxing the strictness of concept definitions themselves. This extension has both practical and epistemic justification. Using this extension enhances automatic classification and enables effective interactive classification.

References

[1] H. Ait-Kaci. *A New Model of Computation Based on a Calculus of Type Subsumption.* MS-CIS 83-40, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, Pennsylvania, 1983.

[2] R. Bobrow. *RSRs, Role Orderings and Quantification.* Technical Discussion Report 4842, Bolt Beranek and Newman, Inc., 1982.

[3] R. J. Brachman. *A Structural Paradigm for Representing Knowledge.* Technical Report 3605, Bolt Beranek and Newman, Inc., May 1978.

[4] R. J. Brachman and J. G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216, 1985.

[5] T. Finin. Interactive classification: a technique for acquiring and maintaining knowledge bases. *Proceedings of the IEEE*, 74(10):1414–1420, October 1986.

[6] T. Finin and D. Silverman. Interactive classification: a technique for building and maintaining knowledge bases. *Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems*, 107–114, August 1984.

[7] Robert Kass, Ron Katriel, and Tim Finin. *Breaking the Primitive Concept Barrier.* Technical Report MS-CIS-86-36, Department of Computer and Information Science, University of Pennsylvania, 1986.

[8] K. R. McKeown. *Text Generation—Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.* Cambridge University Press, 1985.

[9] M. G. Moser. *An Overview of NIKL, The New Implementation of KL-ONE.* Technical Report 5421, Bolt Beranek and Newman, Inc., 1983.

[10] S. P Schwartz. *Naming, Necessity, and Natural Kinds.* Cornell University Press, Ithaca, New York, 1981.

[11] D. L. Silverman. *An Interactive, Incremental Classifier.* MS-CIS 84-10, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, Pennsylvania, 1984.

[12] M. Vilain. The restricted language architecture of a hybrid representation system. In 9^{th} *Int. Joint Conf. on Artificial Intelligence*, pages 547–551, William Kaufmann, Inc., Los Altos, California, 1985.