TAXI: A Taxonomic Assistant

Thomas Y. Galloway
USC-Information Sciences Institute[1]
4676 Admiralty Way
Marina del Rey, CA 90292-6695
Telephone: 213-822-1511
GALLOWAY @ VAXA.ISI.EDU

Abstract

The task of constructing knowledge bases is a difficult one due to their size and complexity. A useful aid for this task would be a system which has both knowledge about a particular knowledge representation scheme and tools with which to manipulate the representation's components. Such a system would be a *knowledge maniupulation system (KMS)*.

This paper describes a KMS called TAXI which is used to manipulate knowledge in the form of a taxonomic knowledge representation scheme. The particular taxonomic representation used is discussed, along with support for the usefulness of a KMS for this particular representation scheme. Tools provided in the TAXI system are described, as are possible applications for the system.

1 The Problem

This paper describes TAXI, a first version of a *Knowledge Manipulation System (KMS)* for taxonomic representation. A KMS is a system which contains knowledge about some representation scheme and its components, along with tools allowing users to manipulate and explore domain knowledge represented by that scheme.

In order to most effectively utilize large amounts of domain knowledge, it is essential that the knowledge be organized in some manner. In addition, the organization process enables a person to better understand the domain and relationships among the knowledge being organized. The usefulness of this meta-knowledge and experimentation within a domain while trying to organize its information is shown in the work of Swartout and Papert [Swartout, 1981] [Papert, 1979].

The remainder of the paper discusses taxonomic representation, TAXI's design goals, its component tools, and possible applications.

2 Taxonomic Representation and a KMS

The taxonomic representation scheme was selected for several reasons. Taxonomies are useful for organizing declarative knowledge, which is used in many knowledge based systems, particularly diagnostic systems.

Constructing a taxonomy creates a classification process for a domain so that each domain object is uniquely defined by its attributes. A set of objects representing a domain are defined by associating a value with each of a set of attributes for each object. The domain is further subdivided into classes of objects which have the same value for each of a given set of attributes.

The final taxonomy will be the result of numerous decisions about attributes, values, application order of attributes, etc. Only by participating in these decisions will a user, whether human or machine, be fully aware of the implicit knowledge behind the final taxonomic structure. Thus, the construction process will provide the constructor with a better understanding of the utilized domain.

Except for already well defined and understood domains, it is currently not possible to fully automate taxonomy construction [Swartout, 1981]. This is due to taxonomy construction being a meta-classification problem, requiring that large amounts of domain knowledge be possessed before construction begins. Automatic construction is also not practical since users will not obtain the better understanding of the domain from performing the construction.

However, due to the needs of detecting generalities among objects, keeping track of dependencies, determin-

[1]This work was done while the author was at the University of Pennsylvania. It was supported by the author while on a teaching assistantship and by Penn via use of machines. No grants or contracts at all.

ing useful classification attributes, and constantly reforming the taxonomy, large taxonomies are difficult for people to construct.

Thus, the hybrid approach of a KMS, used as a taxonomic assistant in this case, appears appropriate. A person can contribute the domain knowledge and control the construction process while the assistant can keep track of dependencies, perform the grunge work of bookkeeping, and detect generalities in the knowledge uncovered by use of the taxonomic representation. Taxonomy construction is thus easier, more efficient, and allows for more experimentation during the design phase. Taxonomy editing, which is required unless dealing with a static domain, will also be easier.

Finally, a taxonomic KMS is useful for learning about a domain. By using the provided tools to alter the structure of an existing taxonomy, users can examine a domain from many perspectives, discovering the consequences of different classifications and obtaining a better understanding of how objects and classes are interrelated than can be obtained via a static taxonomy such as the biological taxonomy used in schools.

3 TAXI's Representation Scheme

Taxonomies can be represented in several ways. For example, a powerful representation results from the use of a directed acyclic graph or lattice as the base data structure. This representation allows nodes to have several direct subsumers, and is used by the KL-ONE and NIKL representation languages [Brachman, 1985].

A simpler data structure for taxonomic representation is the discrimination tree; a set of nodes partially ordered by a subsumption relation to form a tree structure. Each node can have only one subsumer (parent). Attributes and their values are used as discriminators to divide sets of objects into subsets.

There are four reasons why TAXI uses the discrimination tree structure.

1. Computational simplicity: For the first version of a taxonomic assistant, it was desirable to have an easily implemented representation. Many design decisions involving user interface design, tool selection, and which taxonomic components are important would be the same no matter what representation was used.

2. Conceptual simplicity: Among other aims, TAXI is intended to be used as a learning tool. The discrimination tree representation is easier to comprehend for most users, making it easier to perceive taxonomy component interrelationships and dependencies.

3. Familiarity: Many people are already familiar with the discrimination tree structure from their exposure to the biological taxonomy in school, and thus will already understand the underlying data structure. Having to explain a more complex representation scheme to a user will serve as one more obstacle to system use.

4. Graphical representation: Discrimination trees are simple to represent graphically. A graphic based user interface is important because it conveys information in a direct and easily understood manner. Using a discrimination tree, most users already possess the intuitive understanding that objects/classes which are close together are relatively similar, and that the further down a tree a class is, the less general it is.

TAXI's design paradigm is similar to a good text editor's. It contains knowledge about discrimination tree structure, taxonomy components, their interrelationships and dependencies, and tools for structure and domain manipulation.

The final result of a taxonomy construction should be a tree where each object in the domain set occupies a unique leaf node. Objects are distinguished by defining them in terms of attributes and associated values. Attributes are used as discriminators at levels in the tree.

Taxonomy construction is essentially a trial and error process, and is both incremental and non-modular. Decisions must be made as to which attributes are "best" for the desired taxonomy, and in what order they are applied as discriminators. In order to experiment with these decisions, the taxonomy must be reformed with each new experiment. However, since TAXI automatically reforms the taxonomy for users, experimentation is much easier.

4 TAXI's Components and Tools

TAXI possesses knowledge of six taxonomic components; *Objects, Attributes, Types, Classes, Discriminators,* and the *Taxonomy* itself. TAXI has tools for the creation and editing of each component, as well as other tools for manipulating overall taxonomic structure and interrelationships. While individual tools are not very powerful, the

entire set (currently about 40) provides users with control over the construction, editing, exploration, and experimental processes for discrimination tree taxonomies. In this section, I discuss the structure of a TAXI taxonomy, and some of the more powerful and useful tools.

A taxonomy's initial state is a single *class*, containing all domain *objects*, which is the root of the tree. This inital class/node is divided into subclasses which are the nodes at the next tree level of the tree by selecting an *attribute* and creating a subclass for each possible value of the attribute. In the case of attributes which can have an large, or even infinite number of possible values (ones which take numeric values, for example), the subclasses are limited to values currently used in the taxonomy's objects.

Objects consist of a name and a definition. The definition is the set of defined attributes and their values for that object. Classes contain objects as members, and are defined as the set of attributes and associated values which are the same for all members.

Attributes and their values define classes and objects. For computational efficiency and to let users easily determine possible values, each attribute has a *type* which defines all the possible values for the attribute. Types can be associated with more than one attribute.

Untyped attributes have been suggested due to difficulty in anticipating all possible values when initially defining an attribute [Silverman, 1984]. However, TAXI provides a tool to easily edit type definitions. The usefulness of types for formalization of attribute definitions and detection of invalid values overcomes the minor inconvenience of editing or examining a type definition.

TAXI currently has two *meta-types, numeric* and *non-numeric*, of which all types are one or the other. Non-numeric types may contain numeric values, but numeric types can only contain numbers.

Meta-types exist only for implementation reasons. It is difficult to represent in a menu infinite or very large numbers of possible values, such as are possible for attributes which possess numeric values. By limiting the possible values for an attribute to numbers, it is possible to ask the user to merely type in a number when assigning a value, as opposed to trying to display an infinite number of possible values in a menu. Numeric types may be restricted to allow values only between user-defined ranges; for example a type might allow only values between 1 and 100 as well as between 3000 and 4000.

Types automatically have **NONE** included as one of their values. *NONE* is used when an attribute is not applicable to an object, such as the attribute *hair-color* for the object *Yul-Brenner*. Of course, TAXI has knowledge about the meaning of the *NONE* value. The current implementation of TAXI requires that every object in the taxonomy have a value for each attribute used as a discriminator in the tree structure.

Changes to type definitions can have large impacts on taxonomies. For example, if a value is deleted from a type, all objects with that value for an attribute of that type will be affected, as will all classes based on that attribute/value. Users are prompted for a new value for each instance of an affected attribute, and affected objects are automatically reclassified.

If a type is deleted, the effects can propagate. Users are asked whether to also delete the attributes of that type, and if not, to provide a new type(s). This will further affect the taxonomy, as discussed later.

Types can be defined using other types by taking a subset of values from an existing type, or via an intersection or union operation on a set of existing types. New values can be added to the result of these methods as well.

Types are assigned during attribute definition. A new type may be declared when the first attribute of the type is declared. Immediately after an attribute is declared, users are queried about the attribute's value for each existing object. The attribute/value is incorporated into the existing object definitions.

When a new attribute is defined, users are asked if it should be incorporated immediately into the taxonomy as a discriminator. A user is free to decline. A list of defined attributes which are not used as discriminators is maintained by TAXI and can be examined at any time.

If an attribute is used in the taxonomy as a discriminator, changes to attribute values will cause TAXI to automatically reclassify affected objects, which will alter the membership of the appropriate classes.

Whether an attribute is used as a discriminator, and at what position in the tree it is used, is usually left to users, unless assistance is requested as discussed later. When an attribute is used, the user must specify the level it is to be used at in the tree. TAXI then splits all classes on the chosen level into subclasses based on the member objects' associated value for that attribute. The taxonomy is then reformed by applying the discriminators used at levels below the newly added discriminator to these subclasses in the same order in which they were previously applied.

The order in which attributes are used as discriminators determines whether objects will be contained in the same class(es) at points other than at the top of the tree. For example, if the objects *man* and *ostrich* are both in a taxonomy, and the first discrimination is *Number-of-Legs*, then they'll be in the same class on level 2, while if the discrimination is *Has-Feathers*, they would not. Since ideally each object in the taxonomy should be the sole class member in a leaf node of the tree, and since people intuitively believe that the closer two discrimination tree nodes are together, the more similar they are, the choice of what order to apply discriminators can have have a large effect on how users will percieve the relative similarity of represented objects.

Therefore, TAXI allows users to change the level at which an attribute is used as a discriminator. By altering the discriminator levels, users can observe taxonomies from several perspectives and notice which objects tend to share class membership (or which don't), learning which objects are relatively similar.

Since attributes define objects and classes, their selection is important. TAXI provides a tool for attribute selection called *Attribute Aid* based on Personal Construct Theory [Boose, 1984]. Attribute Aid attempts to encourages users to consider small subsets of the object set. In one form, users are presented with a randomly selected set of three objects on which Attribute Aid has not been previously used. They are then asked to define an attribute which would distinguish one object from the others; one of the objects would have a different value for that attribute than the other two. By remaining in Attribute Aid, such an attribute can be determined for every object. It is also possible to continue the process until all objects are uniquely defined, and as such are all represented as the sole member of a class which is a terminal node in the discrimination tree.

Attribute Aid can also present users with a class consisting of more than one object which is a leaf node in the tree. In other words, a class consisting of objects which have yet to be uniquely defined. Until all objects in the class are so distinguished, Attribute Aid will query the user for an attibute which will distinguish an object from the other members of that class.

TAXI can also suggest a "best" attribute for a level. Users may request either an *even* or *skewed* distribution for the next level. For even distribution, TAXI determines which attribute which has not been used above that level as a discriminator will cause the least standard deviation among subclass size if so used. For skewed, TAXI finds the attribute which would produce the largest standard deviation. Even distributions create relatively balanced trees, while skewed distributions are a quick method to cause singular terminal nodes to form quickly, but which form an unbalanced tree.

TAXI also allows users to begin with a set of objects, and then order TAXI to continue to find either even or skewed discriminations until either all objects are uniquely defined, or all attributes have been used as discriminators.

It is also possible to define dependencies among objects or attributes such that changes to the definition of one will affect the definition of the other. Users are informed of such changes, although they occur automatically.

The preceding has been an abbreviated description of some of the tools provided by TAXI for manipulation of the taxonomic structure supplied by a discrimination tree. While each tool's power is limited, the combined power of the entire set allows users to quickly and easily construct, modify, or experiment with a taxonomy represented in the form of a discrimination tree.

5 Applications

TAXI's research goals were to design and build a knowledge manipulation system which could serve as a taxonomic assistant. The system would help users increase their knowledge about a domain by enabling them to manipulate elements of the taxonomic representation, in addition to assisting in the taxonomic construction process. In this section of the paper, we discuss possible applications for a taxonomic assistant.

TAXI has several practical applications. The most obvious would be for it to be used as a tool in the construction or reformulation of large taxonomies. One such area in which it would be quite useful would be the current effort to redefine the biological taxonomy in terms of genetic material and differences [Francis, 1985].

TAXI could also be used to determine previously unrealized similarities between objects in a domain. This information could then be incorporated into knowledge-based diagnostic systems for the domain. For example, a taxonomy of the intended domain for a diagnostic system could be constructed using TAXI. Then, by reforming the taxonomy by using different orderings or combinations of attributes, those diseases which are often located close together in the taxonomy are those which are likely to be confused with each other. A knowledge engineer could incorporate this information in the system in the form of special procedures for distinguishing between the discovered similar cases.

Finally, TAXI's exploratory tools could improve knowledge engineers' domain understanding. A domain expert could use TAXI to create a taxonomy of various domain concepts. A knowledge engineer could then use TAXI's tools to examine, explore, and play with that taxonomy, enabling him/her to obtain a better intuitive understanding of the domain and how its objects interact before beginning work of the knowledge-based system.

6 Future Work

TAXI is a first version taxonomy assistant/KMS. There are several improvements planned for future versions, including the removal of the Numeric/Non-Numeric type distinction, the ability to compare two similar taxonomies, the ability to define and explore taxonomies for domain subsets in depth and then incorporate findings into the main taxonomy, and many other relatively minor refinements to TAXI's tools.

Most importantly, a second generation TAXI should be able to handle more sophisticated representation schemes such as the KL-ONE representation and an extended discrimination tree which allows objects to have multiple values for its attributes.

7 Conclusion

This paper has described TAXI, a knowledge manipulation system for taxonomic knowledge representations which use a discrimination tree as their data structure. TAXI's KMS paradigm appears successful in allowing users to obtain the benefits of constructing a taxonomy without having to deal with many of the difficulties inherent to the task which are irrelevant to the goals of constructing a usable taxonomy, or obtaining a better understanding of the domain. It is also successful in allowing users to experiment with taxonomies, allowing them to view domain knowledge from several perspectives by means of altering the definitions of objects and classes.

It is hoped that TAXI will prove to be of use to knowledge engineers, biologists, and perhaps most importantly, to students seeking to improve their understanding of various domains.

8 References

1. Boose, J.,
   Personal Construct Theory and the Transfer of Human Expertise, AAAI Conference Proceedings, 1984.

2. Brachman, R.,
   On the Epistemological Status of Semantic Networks, in **Readings in Knowledge Representation**, Brachman R. & Levesque H. (eds), Morgan Kaufman Publishing Company, 1985.

3. Francis, K.,
   Personal communication, 1985.

4. Papert, S.,
   Mindstorms, Houghton-Mifflen, 1979.

5. Silverman, D.,
   An Interactive, Incremental Classifier, Technical Report MS-CIS-84-10, University of Pennsylvania, 1984.

6. Swartout W.,
   Explaining and Justifying Expert Consulting Programs, IJCAI Conference Proceedings, 1981.