ON THE AUTOMATIC TRANSFORMATION OF CLASS MEMBERSHIP CRITERIA

Barbara C. Sangster
Rutgers University

This paper addresses a problem that may arise in classification tasks: the design of procedures for matching an instance with a set of criteria for class membership in such a way as to permit the intelligent handling of inexact, as well as exact matches. An inexact match is a comparison between an instance and a set of criteria (or a second instance) which has the result that some, but not all, of the criteria described (or exemplified) in the second are found to be satisfied in the first. An exact match is such a comparison for which all of the criteria of the second are found to be satisfied in the first. The approach presented in this paper is to transform the set of criteria for class membership into an exemplary instance of a member of the class, which exhibits a set of characteristics whose presence is necessary and sufficient for membership in that class. Use of this exemplary instance during the matching process appears to permit important functions associated with inexact matching to be easily performed, and also to have a beneficial effect on the overall efficiency of the matching process.

1. INTRODUCTION

An important common element of many projects in Artificial Intelligence is the determination of whether a particular instance satisfies the criteria for membership in a particular class. Frequently, this task is a component of a larger one involving a set of instances, or a set of classes, or both. This determination need not necessarily call for an exact match between an instance and a set of criteria, but only for the "best," or "closest," match, by some definition of goodness or closeness. One important specification for such tasks is the capability for efficient matching procedures; another is the ability to perform inexact, as well as exact matches.

One step towards achieving efficient matching procedures is to represent criteria for class membership in the same way as descriptions of instances. This may be done by transforming the set of criteria, through a process of symbolic instantiation, into a kind of prototypical instance, or exemplary member of the class. This permits the use of a simple matching algorithm, such as one that merely checks whether required components of the definition of the class are also present in the description of the instance. This also permits easy representation of modifications to the definition, whenever the capability of inexact matching is desired.

Other ways of representing definitions of classes might be needed for other purposes, however. For example, the knowledge-representation language AIMDS would normally be expected to represent definitions in a more complex manner, involving the use of pattern-directed inference rules. These rules may be used, e.g., to identify inconsistencies and fill in unknown values. A representation of a definition derived through symbolic instantiation does not have this wide a range of capabilities, but it does appear to offer advantages over the other representation for efficient matching and for easy handling of inexact matches. We might, therefore, like to be able to translate back and forth between the two forms of representation as our needs require.

An algorithm has been devised for automatically translating a definition in one of the two directions -- from the form using the pattern-directed inference rules into a simpler, symbolically instantiated form [11]. This algorithm has been shown to work correctly for any well-formed definition in a clearly-defined syntactic class [10]. The use of the symbolically instantiated form for both exact and inexact matches is outlined here; using a hand-created symbolic instantiation, a run demonstrating an exact match is presented. The paper concludes with a discussion of some implications of this approach.

2. INEXACT MATCHING

The research project presented in this paper was motivated by the need for determining automatically whether a set of facts comprising the description of a legal case satisfies the conditions expressed in a legal definition, and, if not, in what respects it fails to satisfy those conditions [8], [9], [10], [11], [13]. The need to perform this task is central to a larger project whose purpose is the representation of the definitions of certain legal concepts, and of decisions based on those concepts.

Inexact matching arises in the legal/judicial domain when a legal class must be assigned to the facts of the case at hand, but when an exact match cannot be found between those facts and any of the definitions of possible legal classes. In that situation, a reasonable first-order approximation to the way real decisions are made may be to say that the class whose definition offers the "best" or " closest" match to the facts of the case at hand is the class that should be assigned to the facts in question. That is the approach taken in the current project.

In addition to the application discussed here (the assignment of an instance of a knowledge structure to one of a set of classes), inexact matching and close relatives thereof are also found in several other domains within computational linguistics. Inexact matching to a knowledge structure may also come into play in updating a knowledge base, or in responding to queries over a knowledge base [5], [6]. In the domain of syntax, an inexact matching capability makes possible the correct interpretation of utterances that are not fully grammatical with respect to the grammar being used [7]. In the domains of speech understanding and character recognition, the ability to perform inexact matching makes it possible to disregard errors caused by such factors as noise or carelessness of the speaker or writer.

When an inexact match of an instance has been identified, the first step is to attempt to deal with any criteria which were not found to be satisfied in the instance, but were not found not to be satisfied either -- i.e., the unknowns. At that point, if an exact match still has not been achieved, two modes of action are possible: the modification of the instance whose characterization is being sought, or the modification of the criteria by means of which a characterization is found. The choice between these two responses (or of the way in which they are combined) appears to be a function of the domain and sometimes also of the particular item in question. In general, in the

The research reported in this paper was partially supported by the National Science Foundation under Grant #SOC-7811408 and by the Research Foundation of the State University of New York under Grant #150-2197-A.

legal/judicial domain, the facts of the case, once determined, are fixed (unless new evidence is introduced), but the criteria for assigning a legal characterization to those facts may be modified.

3. THE MATCHING OF LEGAL DEFINITIONS: A PRELIMINARY TRANSFORMATION

Because of the importance of inexact matching in the legal/judicial domain, it is desirable to utilize a matching procedure that permits useful functions related to inexact matching to be performed conveniently. Such functions include a way of easily determining all the respects in which attempted exact matches to a particular definition might fail, a way of easily determining what changes to a definition would be sufficient for an exact match with a particular case to be permitted, and a way of ensuring that a contemplated modification to a definition will not introduce inconsistencies.

Two features of a representational scheme that would appear to help in performing these functions conveniently are

> SPEC1) that the scheme permit a distinction to be made between those propositions that must be found to be true of any instance satisfying the definition and any other propositions that might also be true of the instance, and
>
> SPEC2) that the scheme permit the former set of propositions to be expressed in a simple, unified way, so as to reduce or even eliminate the need for inferencing and other processing activities when the functions outlined above are performed.

By satisfying SPEC1, we permit the propositions which are central to the matching process to be distinguished from any others; by satisfying SPEC2, we permit those propositions to be accessed and manipulated (e.g., for the inexact matching functions listed above) in an efficient and straightforward manner. Thus, the fulfillment of SPEC1 and SPEC2 significantly strengthens our ability to perform functions central to the inexact matching process.

A representational scheme that meets these specifications has been designed, and an experimental implementation performed. The approach used is to precede the matching activity proper with a one-time preprocessing phase, during which the definition is automatically transformed from the form in which it is originally expressed into a representational scheme which appears to be more suitable to the matching task at hand. The transformation algorithm makes use of a distinction between those components of the definition which must be found to be true and those whose truth either may be inferred or else is irrelevant to the matching process. The transformation is performed by means of a process of symbolic instantiation of the definition -- the translation of the definition from a set of criteria for satisfying the definition into an exemplary instance of the concept itself. The transformed definition resulting from this process appears to meet the specifications given above.

The input to the transformation process is a definition expressed in two parts:

> COMPONENT1) a set of propositions consisting of relations between typed variables organized in frame form, and
>
> COMPONENT2) a set of pattern-directed inference rules expressing constraints on how the propositions in COMPONENT1 may be instantiated.

The propositions in COMPONENT1 include propositions that must be found to be true of any instance satisfying the

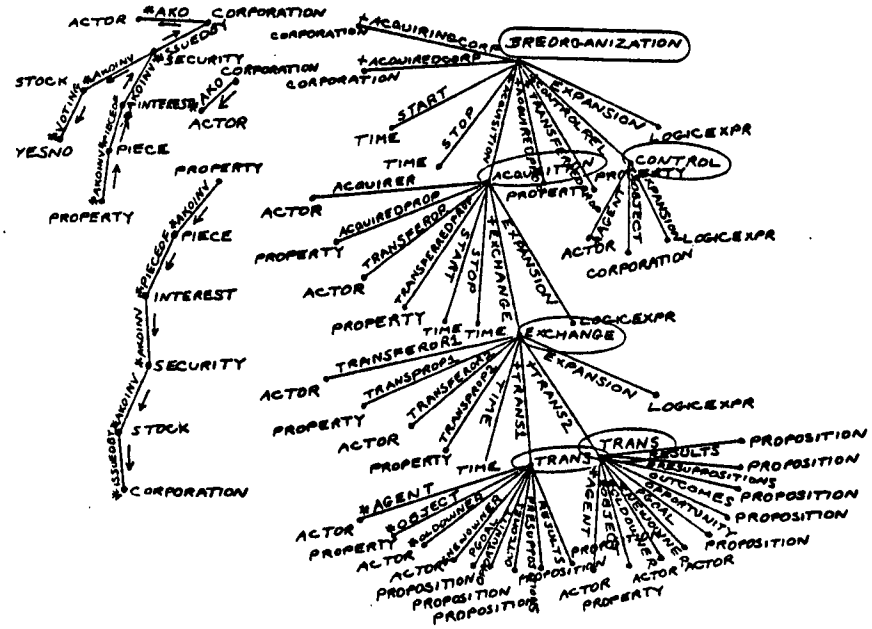

Figure 1: COMPONENT1 for a sample definition.

definition, as well as other propositions that do not have this quality.

The output from the transformation process that is used for matching with an instance is a symbolically instantiated form of the definition called the KERNEL structure for the definition. It consists solely of a set of propositions expressing relations between instances. These are precisely those propositions whose truth must be observed in any instance satisfying the definition. Constraints on instantiation (COMPONENT2 above) are reflected in the choice of values for the instances in these propositions. Thus the KERNEL structure has the properties set forth in SPEC1 and SPEC2 above, and its use during the matching process may consequently be expected to help in working with inexact matches. For similar reasons, use of the KERNEL structure appears also to permit a significant improvement in efficiency of the overall matching process [10], [11].

The propositions input to the transformation process (i.e., COMPONENT1) are illustrated, for the definition of a kind of corporate reorganization called a BREORGANIZATION, in Figure 1; the arcs represent relations, and the nodes represent the types of the instances between which the relations may hold. Several of the pattern-directed inference rules input to the transformation process (COMPONENT2) for part of the same definition are illustrated in Figure 2. The KERNEL structure for that definition output by the transformation process is illustrated in Figure 3. The propositions shown there are the ones whose truth is necessary and sufficient for the definition to have been met. Bindings constraints between nodes are reflected in the labels of the nodes; the nodes in Figure 3 represent instances. Thus, the two components represented in Figures 1 and 2 are transformed, for the purposes of matching, into the structure represented in Figure 3.

The transformation process is described in more detail in [10] and [11]; [10] also contains an informal proof that the transformation algorithm will work correctly for all definitions in a well-defined syntactic class.

4. EXECUTION OF THE MATCHING PROCESS

Once the transformation of a definition has been performed, it need never again be repeated (unless the definition itself should change), and the compiled KERNEL structure may be used directly whenever a set of facts comprising a description of a legal case is presented for comparison with the definition.

```
((EXCHANGE X) TRANS1 (TRANS T1)
  IFF         (X (TRANSFEROR1 AGENTOF) T1)
              (X (TRANSPROP2 OBJECTOF) T1)
              (X (TRANSFEROR1 OLDOWNEROF) T1)
              (X (TRANSFEROR2 NEWOWNEROF) T1))

((EXCHANGE X) TRANS2 (TRANS T2)
  IFF         (X (TRANSFEROR2 AGENTOF) T2)
              (X (TRANSPROP1 OBJECTOF) T2)
              (X (TRANSFEROR2 OLDOWNEROF) T2)
              (X (TRANSFEROR1 NEWOWNEROF) T2))

((EXCHANGE X) TRANSFEROR1 (ACTOR A)
  IFF         (X (TRANS1 AGENT) A)
              (X (TRANS1 OLDOWNER) A)
              (X (TRANS2 NEWOWNER) A))

((EXCHANGE X) TRANSFEROR2 (ACTOR A)
  IFF         (X (TRANS2 AGENT) A)
              (X (TRANS2 OLDOWNER) A)
              (X (TRANS1 NEWOWNER) A))
```

Figure 2: A portion of COMPONENT2 for a sample definition.

In order to control possible combinatoric difficulties, the KERNEL structure is decomposed into a set of small networks, against each of which all substructures of the same type in the case description are tested for a structural match (STAGE1). DMATCH [15], a function written by D. Touretzky, performed structural matching in the experimental implementation. The hope is that "small networks" can be selected from the KERNEL in such a way that matching to any single small network will involve a minimal degree of combinatoric complexity. For an exact match, the substructures that survive STAGE1 (and no others) are then combined in all possible valid ways into larger networks of some degree of increase in complexity. A structural match of each of these structures with the corresponding substructure of the KERNEL is then attempted, and bindings constraints between formerly separate components of the new network are thereby tested. This process is repeated with surviving substructures until the structural match is conducted against the KERNEL structure itself. When the criterion for matching at each stage is an exact match, as described above, the survivors of the final stage of structural matching represent all and only the subcases in the case description that meet the conditions expressed in the definition.

The execution of the matcher in the manner described above is illustrated in Figure 4. For this example, five instances of the type TRANS (T1, T2, T3, T4, T5), two instances of the type CONTROL (C1, C2), and two instances of PROPERTY (O6, O9) were used. The value of MAKEFULLLIST shows the survivors of STAGE1. The value of BGO shows the single valid instance of a BREORGANIZATION that can be created from these components.

An inexact matching capability, not currently implemented, would determine, when at any stage a match failed,

1) why it had failed, and

2) how close it had come to being an exact match.

At the next stage, a combination of substructures would be submitted for consideration by the matcher only if it had met some criterion of proximity to an exact match -- either on an absolute scale, or relative to the other candidates for matching. When the final stage of the matching process had been completed, that candidate (or those candidates) that permitted the most nearly exact match could then be selected.

In order to perform the inexact matching function outlined in the preceding paragraph, an algorithm for computing distance from a exact match must be formulated. For the reasons given above, we anticipate that

1) the transformation of definitions into the corresponding KERNEL structures will make that task easier, and that

2) once a distance algorithm has been formulated, the use of the KERNEL structure will contribute to performing the inexact matching function with efficiency and conceptual clarity.

5. CONCLUSIONS

The capability for the intelligent handling of inexact matches has been shown to be an important requirement for the representation of certain classification tasks. A procedure has been outlined whereby a set of criteria for membership in a particular class may be transformed into an exemplary instance of a member of that class.

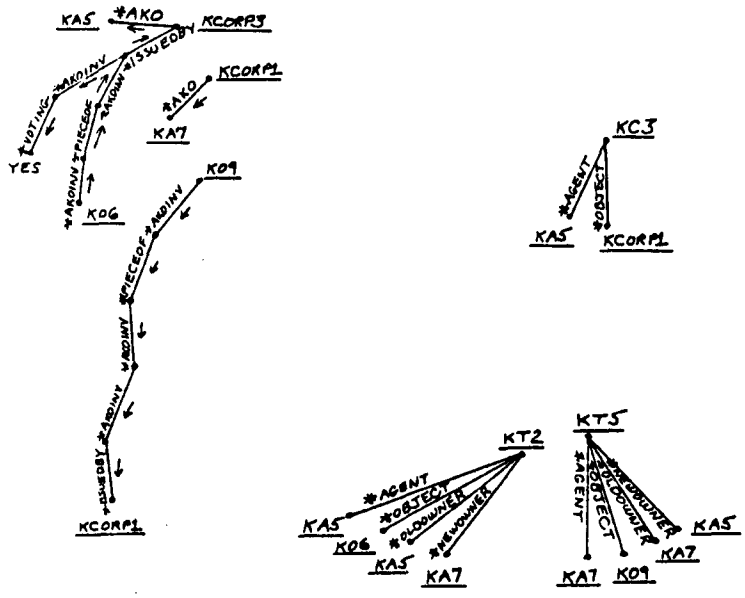

Figure 3: The KERNEL structure for a sample definition.

As we have seen, use of that exemplary instance during the matching process appears to permit important functions associated with inexact matching to be easily performed, and also to have a beneficial effect on the overall efficiency of the matching process.

ACKNOWLEDGEMENTS

The author is grateful to the following for comments and suggestions on the work reported on in this paper: S. Amarel, V. Ciesielski, L. T. McCarty, T. Mitchell, N. S. Sridharan, and D. Touretzky.

BIBLIOGRAPHY

[1] Freuder, E. C. 1978. "Synthesizing Constraint Expressions". CACM, vol. 21, pp. 958-966.

[2] Haralick, R. M. and L. G. Shapiro. 1979. "The Consistent Labelling Problem: Part I". IEEE Transactions on PAMI, vol. 1, pp. 173-184.

[3] Hayes-Roth, F. 1978. "The Role of Partial and Best Matches in Knowledge Systems". Pattern-Directed Inference Systems, ed. by D. Waterman and F. Hayes-Roth. Academic Press.

[4] Hayes-Roth, F. and D. J. Mostow. 1975. "An Automatically Compilable Recognition Network for Structured Patterns". Proceedings of IJCAI-75, vol. 1, pp. 246-251.

[5] Joshi, A. K. 1978a. "Some Extensions of a System for Inference on Partial Information". Pattern-Directed Inference Systems, ed. by D. Waterman and F. Hayes-Roth. Academic Press.

[6] Joshi, A. K. 1978b. "A Note on Partial Match of Descriptions: Can One Simultaneously Question (Retrieve) and Inform (Update)?". TINLAP-2: Theoretical Issues in Natural Language Processing-2.

[7] Kwasny, S. and N. K. Sondheimer. 1979. "Ungrammaticality and Extra-Grammaticality in Natural Language Understanding Systems". This volume.

```
SECOND-CONTEXT >>        (BOO)

Enter MAKEFULLLIST:
!      PROTS = (PROTOTRANS1 PROTOTRANS2 PROTOCONTROL1 PROTO09 PROTO06)
MAKEFULLLIST = ((06) (06 09) (C1 C2) (T2 T4 T5) (T2 T4 T5))


((T2 T5 C2 09 06) NIL)
```

Figure 4: Sample execution of the matching process.

[8] McCarty, L. T. 1977. "Reflections on TAXMAN: An Experiment in Artificial Intelligence and Legal Reasoning". Harvard Law Review, vol. 90, pp. 837-893.

[9] McCarty, L. T., N. S. Sridharan, and B. C. Sangster. 1979. "The Implementation of TAXMAN II: An Experiment in Artificial Intelligence and Legal Reasoning". Rutgers University Report #LCSR-TR-3.

[10] Sangster, B. C. 1979a. "An Automatically Compilable Hierarchical Definition Matcher". Rutgers University Report #LRP-TR-3.

[11] Sangster, B. C. 1979b. "An Overview of an Automatically Compilable Hierarchical Definition Matcher". Proceedings of the IJCAI-79.

[12] Sridharan, N. S. 1978a. (Ed.) "AIMDS User Manual, Version 2." Rutgers University Report #CBM-TR-89.

[13] Sridharan, N. S. 1978b. "Some Relationships between BELIEVER and TAXMAN". Rutgers University Report #LCSR-TR-2.

[14] Srinivasan, C. V. 1976. "The Architecture of Coherent Information System: A General Problem Solving System". IEEE Transactions on Computers, vol. 25, pp. 390-402.

[15] Touretzky, D. 1978. "Learning from Examples in a Frame-Based System". Rutgers University Report #CBM-TR-87.

[16] Woods, W. A. 1975. "What's in a Link: Foundations for Semantic Networks". In Representation and Understanding, ed. by D. G. Bobrow and A. Collins. Academic Press.