The Costs of Inheritance in Semantic Networks

Rob't F. Simmons

The University of Texas, Austin

Abstract

Questioning texts represented in semantic relations[1] requires the recognition that synonyms, instances, and hyponyms may all satisfy a questioned term. A basic procedure for accomplishing such loose matching using inheritance from a taxonomic organization of the dictionary is defined in analogy with the unification algorithm used for theorem proving, and the costs of its application are analyzed. It is concluded that inheritance logic can profitably be included in the basic questioning procedure.

AI Handbook Study

In studying the process of answering questions from fifty pages of the AI Handbook, it is striking that such subsections as those describing problem representations are organized so as to define conceptual dictionary entries for the terms. First, class definitions are offered and their terms defined; then examples are given and the computational terms of the definitions are instantiated. Finally the technique described is applied to examples and redefined mathematically. Organizing these texts (by hand) into coherent hierarchic structures of discourse results in very usable conceptual dictionary definitions that are related by taxonomic and partitive relations, leaving gaps only for non-technical terms. For example, in "give snapshots of the state of the problem at various stages in its solution," terms such as "state", "problem", and "solution" are defined by the text, while "give", "snapshots", and "stages" are not.

Our first studies in representing and questioning this text have used semantic networks with a minimal number of case arcs to represent the sentences and Superset/Instance and *Of/Has arcs to represent, respectively, taxonomic and partitive relations between concepts. Equivalence arcs are also used to represent certain relations signified by uses of "is" and apposition and *AND and *OR arcs represent conjunction. Since June 1982, eight question-answering systems have been written, some in procedural logic and some in compilable ELISP. Although we have so far studied questioning and data manipulation operations on about 40 pages of the text, the detailed study of inheritance costs discussed in this paper was based on 170 semantic relations (SRs), represented by 733 binary relations each composed of a node-arc-node triple. In this study the only inference rules used were those needed to obtain transitive closure for inheritance, but in other studies of this text a great deal of power is gained by using general inference rules for paraphrasing the question into the terms given by an answering text. The use of paraphrastic inference rules is computationally expensive and is discussed elsewhere [Simmons 1983].

The text-knowledge base is constructed either as a set of triples using subscripted words, or by establishing node-numbers whose values are the complete SR and indexing these by the first element of every SR. The latter form, shown in Figure 1, occupies only about a third of the space that the triples require and neither form is clearly computationally better than the other.

The first experiments with this text-knowledge base showed that the cost of following inheritance arcs, i.e. obtaining taxonomic closures for concepts, was very high; some questions required as much as a minute of central processor time. As a result it was necessary to analyze the process and to develop an understanding that would minimize any redundant computation. Our current system for questioning this fragment knowledge base has reduced the computation time to the range of 1/2 to less than 15 seconds per question in uncompiled ELISP on a DEC 2060.

I believe the approach taken in this study is of particular interest to researchers who plan to use the taxonomic structure of ordinary dictionaries in support of natural language processing operations. Beginning with studies made in 1975 [Simmons and Chester, 1977] it was apparent to us that question-answering could be viewed profitably as a specialized form of theorem proving that

[1] supported by NSF Grant IST 8200976

Example SR representation for a sentence:

```
 (C100 A STATE-SPACE REPRESENTATION OF A PROBLEM EMPLOYS TWO
    KINDS OF ENTITIES: STATES, WHICH ARE DATA STRUCTURES GIVING
    "SNAPSHOTS" OF THE CONDITION OF THE PROBLEM AT EACH STAGE OF ITS
    SOLUTION, AND OPERATORS, WHICH ARE MEANS FOR TRANSFORMING THE
    PROBLEM FROM ONE STATE TO ANOTHER)

(N137 (REPRESENTATION SUP  N101 HAS N138 EG N139 SNT C100))
(N138 (ENTITY NBR PL QTY 2. INST N140 INST N141 SNT C100))
(N140 (STATE NBR PL EQUIV N142 SNT C100))
(N142 (STRUCTURE *OF DATA INSTR* N143 SNT C100))
(N143 (GIVE TNS PRES INSTR N142 AE N144 *AT N145 SNT C100))
(N144 (SNAPSHOT NBR PL *OF N146 SNT C100))
(N146 (PROBLEM NBR SING HAS N145 SUP N79 SNT C100))
(N145 (STAGE NBR PL IDENT VARIOUS *OF N147 SNT C100))
(N147 (SOLUTION NBR SING SNT C100))
(N141 (OPERATOR NBR PL EQUIV* N148 SNT C100))
(N148 (PROCEDURE NBR PL INSTR* N149 SNT C100))
(N149 (TRANSFORM TNS PRES AE N146 *FROM N164 *TO N165 SNT C100))
(N164 (STATE NBR SING IDENT ONE SUP N140 SNT C100))
(N165 (STATE NBR SING IDENT ANOTHER SUP N140 SNT C100))
```

Example of SR representation of the question, "How many entities are used in the state-space representation of a problem?"

```
(REPRESENTATION *OF (STATE-SPACE *OF PROBLEM) HAS (ENTITY QTY X))
```

Figure 1. Representation of Semantic Relations

```
Query Triple:  A R  B
Match Candid.  + +  +  + means a match by unification.
               + +  C  (CLOSAB C B)
               + +  C  (CLOSCP R C B)
               + R1 +  (SYNONYM R R1)
               B R1 A  (CONVERSE R R1)
               C +  +  (CLOSAB C A)
```

where CLOSAB stands for Abstractive Closure and is defined in procedural logic (where the symbol < is shorthand for the reversed implication sign <--, i.e. P < Q S is equivalent to Q ^ S --> P):

```
(CLOSAB N1 N2) < (OR (INST N1 N2)(SUP N1 N2))
(INST N1 N2) < (OR (N1 INST N2)(N1 EQUIV* N2))
(INST N1 N2) < (INST N1 X)(INST X N2)
(SUP N1 N2) < (OR (N1 EQUIV N2)(N1 SUP N2))
(SUP N1 N2) < (SUP N1 X)(SUP X N2)
```

CLOSCP stands for Complex Product Closure and is defined as

```
(CLOSCP R N1 N2) < (TRANSITIVE R)(N1 R N2)
                     "N1 R N2 is the new A R B"
(CLOSCP R N1 N2) < (N1 *OF N2)**
(CLOSCP R N1 N2) < (N1 LOC N2)**
(CLOSCP R N1 N2) < (N1 *AND N2)
(CLOSCP R N1 N2) < (N1 *OR N2)
```

** These two relations turn out not to be universally true complex products; they only give answers that are possibly true, so they have been dropped for most question answering applications.

Figure 2. Conditions for Matching Question and Candidate Triples

used taxonomic connections to recognize synonymic terms in a question and a candidate answer. A procedural logic question-answerer was later developed and specialized to understanding a story about the flight of a rocket [Simmons 1984, Simmons and Chester, 1982, Levine 1980]. Although it was effective in answering a wide range of ordinary questions, we were disturbed at the magnitude of computation that was sometimes required. This led us to the challenge of developing a system that would work effectively with large bodies of text, particularly the AI Handbook. The choice of this text proved fortunate in that it provided experience with many taxonomic and partitive relations that were essential to answering a test sample of questions.

This brief paper offers an initial description of a basic process for questioning such a text and an analysis of the cost of using such a procedure. It is clear that the technique and analysis apply to any use of the English dictionary where definitions are encoded in semantic networks.

Relaxed Unification for Matching Semantic Relations

In the unification algorithm, two n-tuples, n1 and n2, unify if Arity(n1) = Arity(n2) and if every element in n1 matches an element in n2. Two elements e1 and e2 match if e1 or e2 is a variable, or if e1 = e2, or in the case that e1 and e2 are lists of the same length, each of the elements of e1 matches a corresponding element of e2.

Since semantic relations (SRs) are unordered lists of binary relations that vary in length and since a question representation (SRq) can be answered by a sentence candidate (SRc) that includes more information than the question specified, the Arity constraint is revised to Arity(SRq) Less/Equal Arity(SRc).

The primitive elements of SRs include words, arcnames, variables and constants. Arcnames and words are organized taxonomically, and words are further organized by the discourse structures in which they occur. One or more element of taxonomic or discourse structure may imply others. Words in general can be viewed as restricted variables whose values can be any other word on an acceptable inference path (usually taxonomic) that joins them. The matching constraints of unification can thus be relaxed by allowing two terms to match if one implies the other in a taxonomic closure.

The matching procedure is further adapted to read SRs effectively as unordered lists of triples and to seek for each triple in SRq a corresponding one in SRc. The two SRs below match because Head matches Head, Arc1 matches Arc1, Val1 matches Val1, etc. even though they are not given in the same order.

SRq (Head Arc1 Val1, Arc2 Val2, ..., Arcn Valn)
SRc (Head Arc2 Val2, Arc1 Val1, ..., Arcn Valn)

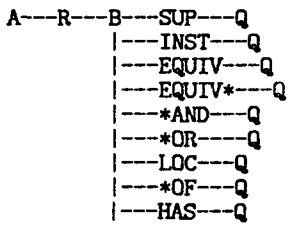
The SR may be represented (actually or virtually) as a list of triples as follows:

SRq ((Head Arc1 Val1)
(Head Arc2 Val2) ..., (Head Arcn Valn))

Two triples match in Relaxed Unification according (at least) to the conditions shown in Figure 2. The query triple, A R B may match the candidate giving + + + to signify that all three elements unified. If the first two elements match, the third may be matched using the procedures CLOSAB or CLOSCP to relate the non-matching C with the question term B by discovering that B is either in the abstractive closure or the complex product closure of C. The *abstractive closure* of an element is the set of all triples that can be reached by following separately the SUP and EQUIV arcs and the INST and EQUIV* arcs. The *complex product closure* is the set of triples that can be reached by following a set of generally transitive arcs (not including the abstractive ones). The arc of the question may have a synonym or a converse and so develop alternative questions, and additional questions may be derived by asking such terms as C R B that include the question term A in their abstractive closure. Both closure procedures should be limited to n-step paths where n is a value between 3 and 6.

Computational Cost

In the above recursive definition the cost is not immediately obvious. If it is mapped onto a graphic representation in semantic network form, it is possible to see some of its implications. Essentially the procedure first seeks a direct match between a question term and a candidate answer; if the match fails, the abstractive closure arcs, SUP, INST, EQUIV, and EQUIV* may lead to a new candidate that does match. If these fail, then complex product arcs, *OF, HAS, LOC, AND, and OR may lead to a matching value. The graph below outlines the essence of the procedure.



This graph shows nine possible complex product paths to follow in seeking a match between B and Q. If we allow each path to extend N steps such that each step has the same number of possible paths, then the worst case computation, assuming each candidate SR has all the arcs, is of the order, 9 raised to the Nth. If the A term of the question also has these possibilities, and the R term has a synonym, then there appear to be 2*2*9**Nth possible candidates for answers. The first factor of 2 reflects the converse by assigning the A term 9**N paths. Assuming only one synonym, each of two R terms might lead to a B via any of 9 paths, giving the second factor of 2. If the query arc is also transitive, then the power factor 9 is increased by one.

In fact, SRs representing ordinary text appear to have less than an average of 3 possible CP paths, so something like 2*3**Nth seems to be the average cost. So if N is limited to 3 there are about 2*81=162 candidates to be examined for each subquestion. These are merely rough estimates, but if the question is composed of 5 subquestions, we might expect to examine something on the order of a thousand candidates in a complete search for the answer. Fortunately, this is accomplished in a few seconds of compùtation time.

The length of transitive path is also of importance for two other reasons. First, most of the CP arcs lead only to probable inference. Even superset and instance are really only highly probable indicators of equivalence, while LOC, HAS, and *OF are even less certain. Thus if the probability of truth of match is less than one for each step, the number of steps that can reasonably be taken must be sharply limited. Second, it is the case empirically that the great majority of answers to questions are found with short paths of inference. In one all-answers version of the QA-system, we found a puzzling phenomenon in that all of the answers were typically found in the first fifteen seconds of computation although the exploration continued for up to 50 seconds. Our current hypothesis is that *the likelihood of discovering an answer falls off rapidly as the length of the inference path increases.*

Disussion

It is important to note that this experiment was solely concerned with the simple levels of inference concerned in inheritance from a taxonomic structure. It shows that this class of inference can be embedded profitably in a procedure for relaxed unification. In addition it allows us to state rules of inference in the form of semantic relations.

For example we know that the commander of troops is responsible for the outcome of their battles. So if we know that Cornwallis commanded an army and the army lost a battle, then we can conclude correctly that Cornwallis lost the battle. An SR inference rule to this effect is shown below:

Rule Axiom:

```
((LOSE AGT X AE Y) <-- (SUP X COMMANDER)
      (SUP Y BATTLE)
      (COMMAND AGT X AE W)
      (SUP W MILITARY-GROUP)
      (LOSE AGT W AE Y))
```

Text Axioms:

```
((COMMAND AGT CORNWALLIS
       AE (ARMY MOD BRITISH)))
   ((LOSE AGT (ARMY MOD BRITISH)
        AE (BATTLE *OF YORKTOWN)))
   ((CORNWALLIS SUP COMMANDER))
   ((ARMY SUP (MILITARY-GROUP)))
   ((YORKTOWN SUP BATTLE))
```

Theorem:

```
((LOSE AGT CORNWALLIS
    AE (BATTLE *OF YORKTOWN)))
```

The relaxed unification procedure described earlier allows us to match the theorem with the consequent of the rule which is then proved if its antecedents are proved. It can be noticed that what is being accomplished is the definition of a theorem prover for the loosely ordered logic of semantic relations. We have used such rules for answering questions of the AI handbook text, but have not yet determined whether the cost of using such rules with relaxed unification can be justified (or whether some theoretically less appealing compilation is needed).

References

Levine, Sharon, Questioning English Text with Clausal Logic, Univ. of Texas, Dept. Comp. Sci., Thesis, 1980.

Simmons, R.F., *Computations from the English*, Prentice-Hall, New Jersey, 1984.

Simmons, R.F., A Text Knowledge Base for the AI Handbook, Univ. of Texas, Dept. of Comp. Sci., TR-83-24, 1983.

Simmons, R.F., and Chester, D.L. Inferences in quantified semantic networks. PROC 5TH INT. JT. CONF. ART. INTELL. Stanford, 1977.