Inference In Text Understanding

Peter Norvig
Computer Science Dept., Evans Hall
University of California, Berkeley
Berkeley CA 94720
This work was supported in part by National Science Foundation grant IST-8208602 and by Defense Advanced Research Projects Agency contract N00039-84-C-0089.

Abstract

The problem of deciding what was implied by a written text, of ‘‘reading between the lines’’ is the problem of *inference*. To extract proper inferences from a text requires a great deal of general knowledge on the part of the reader. Past approaches have often postulated an algorithm tuned to process a particular kind of knowledge structure (such as a script, or a plan). An alternative, unified approach is proposed. The algorithm recognizes six very general classes of inference, classes that are not dependent on individual knowledge structures, but instead rely on patterns of connectivity between concepts. The complexity has been effectively shifted from the algorithm to the knowledge base; new kinds of knowledge structures can be added without modifying the algorithm.

1. The Problem of Inferencing

The reader of a text is faced with a formidable task: recognizing the individual words of the text, deciding how they are structured into sentences, determining the explicit meaning of each sentence, and also making *inferences* about the likely implicit meaning of each sentence, and the implicit connections between sentences. An *inference* is defined to be any assertion which the reader comes to believe to be true as a result of reading the text, but which was not previously believed by the reader, and was not stated explicitly in the text. Note that inferences need not follow logically or necessarily from the text; the reader can jump to conclusions that seem likely but are not 100% certain.

In the past, there have been a variety of programs that handled inferences at the sentential and inter-sentential level. However, there has been a tendency to create new algorithms every time a new knowledge structure is proposed. For example, from the Yale school we see one program, MARGIE, [Sch73d] that handled single-sentence inferences. Another program, SAM [Cul78] was introduced to process stories referring to *scripts*, and yet another, PAM, [Wil78] dealt with *plan/goal* interactions. But in going from one program to the next a new algorithm always replaced the old one; it was not possible to incorporate previous results except by re-implementing them in the new formalism. Even individual researchers have been prone to introduce a series of distinct systems. Thus, we see Charniak going from demon-based [Cha72] to frame-based [Cha78b] to marker-passer based [Cha86] systems. Granger has gone from a plan/goal based system [Gra80b] to a spreading activation model [Gra84b]. One could say that the researchers gained experience, but the programs did not. Both these researchers ended up with systems that are similar to the one outlined here.

2. The Inferencing Algorithm

I have implemented an inferencing algorithm in a program called FAUSTUS (Fact Activated Unified STory Understanding System). A preliminary version of this system was described in [Nor83], and a complete account is given in [Nor86]. The program is designed to handle a variety of texts, and to handle new subject matter by adding new knowledge rather than by changing the algorithm or adding new inference rules. Thus, the algorithm must work at a very general level. The algorithm makes use of six inference classes which are described in terms of the primitives of this language. The algorithm itself can be broken into steps as follows:

Step 0: Construct a knowledge base defining general concepts like actions, locations, and physical objects, as well as specific concepts like bicycles and tax deductions. The same knowledge base is applied to all texts, whereas steps 1-5 apply to an individual text.

Step 1: Construct a semantic representation of the next piece of the input text. Various conceptual analyzers (parsers) have been used for this, but the process will not be addressed in this paper. Occasionally the resulting representation is vague, and FAUSTUS resolves some ambiguities in the input using two kinds of non-marker-passing inferences.

Step 2: Pass markers from each concept in the semantic representation of the input text to adjacent nodes, following along links in the semantic net. Markers start out with a given amount of *marker energy*, and are spread recursively through the network, spawning new markers with less energy, and stopping when the energy value hits zero. (Each of the primitive link types in KODIAK has an energy cost associated with it.) Each marker points back to the marker that spawned it, so we can always trace the marker *path* from a given marker back to the original concept that initiated marker passing.

Step 3: Suggest Inferences based on marker collisions. When two or more markers are passed to the same concept, a *marker collision* is said to have occurred. For each collision, look at the sequence of *primitive link types* along which markers were passed. This is called the *path shape*. If it matches one of six pre-defined path shapes then an inference is suggested. Suggestions are kept in a list called the *agenda*, rather than being evaluated immediately. Note that inferences are suggested solely on the basis of primitive link types, and are independent of the actual concepts mentioned in the text. The power of the algorithm comes from defining the right set of pre-defined path shapes (and associated suggestions).

Step 4: Evaluate potential inferences on the agenda. The result can be either making the suggested inference, rejecting it, or deferring the decision by keeping the suggestion on the agenda. If there is explicit contradictory evidence, an inference can be rejected immediately. If there are multiple potential inferences competing with one another, as when there are several possible referents for a pronoun, then if none of them is more plausible than the others, the decision is deferred. If there is no reason to reject or defer, then the suggested inference is accepted.

Step 5: Repeat steps 1-4 for each piece of the text.

Step 6: At the end of the text there may be some suggested inferences remaining on the agenda. Evaluate them to see if they lead to any more inferences.

The knowledge base is modeled in the KODIAK **representation language, a semantic net-based formalism with a fixed set of well-defined primitive links. We present a simplified version for expository reasons; see [Wil86] for more details.** KODIAK **resembles** KL-ONE **[Bra85], and continues the renaissance of spreading activation approaches spurred by [Fah79].**

3. Describing Inference Classes by Path Shapes

FAUSTUS has six marker-passing inference classes. These classes will be more meaningful after examples are provided below, but I want to emphasize their formal definition in terms of path shapes. Each inference class is characterized in terms of the shapes of the two path-halves which lead to the marker collision. There are three path-half shapes, which are defined in terms of primitive link types: H for hierarchical links between a concept and one of its super-categories, S for links between a concept and one of its ''slots'' or relations, and R for a link indicating a range restriction on a relation. A * marks indefinite repetition, and a $^{-1}$ marks traversal of an inverse link.

Inference Classes	**Path 1**	**Path 2**
Elaboration	Ref	Elaboration
Double Elaboration	Elaboration	Elaboration
Reference Resolution	Ref	Ref
Concretion	Elaboration	Filler
Relation Concretion	Elaboration	Filler
View Application	Constraint	View

Path Name	**Path Shape**
Elaboration	origin H* S R H* collision
Ref	origin H* collision
Filler	origin S^{-1} S H* collision
Constraint	origin H* R^{-1} collision
View	origin H* V H* R^{-1} collision

Non-Marker-Passing Inference Classes
Relation Classification
Relation Constraint

For example, an elaboration collision can occur at concept X when one marker path starts at Y

and goes up any number of H links to X, and another marker path starts at Z and goes up some H links, out across an S link to a slot, then along an R link to the category the filler of that slot must be, and then possibly along H links to X.

Given a collision, FAUSTUS first looks at the shape of the two path-halves. If either shape is not one of the three named shapes, then no inference can come of the collision. Even if both halves are named shapes, a suggestion is made only if the halves combine to one of the six inference classes, and the suggestion is accepted only if certain inference-class-specific criteria are met.

The rest of the paper will be devoted to showing a range of examples, and demonstrating that the general inference classes used by FAUSTUS can duplicate inferences made by various special-purpose algorithms.

4. Early Marker-Passing Based Inferences

One of the first inferencing programs was the Teachable Language Comprehender, or TLC, [Qui69] which took as input single noun phrases or simple sentences, and related them to what was already stored in semantic memory. For example, given the input ‘‘lawyer for the client,’’ the program could output ‘‘at this point we are discussing a lawyer who is employed by a client who is represented or advised by this lawyer in a legal matter.’’ The examples given in [Qui69] show an ability to find the main relation between two concepts, but not to go beyond that. One problem with TLC was that it ignores the grammatical relations between concepts until the last moment, when it applies ‘‘form tests’’ to rule out certain inferences. For the purposes of generating inferences, TLC treats the input as if it had been just ‘‘Lawyer. Client’’. Quillian suggests this could lead to a potential problem. He presents the following examples:

lawyer for the enemy	enemy of the lawyer
lawyer for the wife	wife of the lawyer
lawyer for the client	client of the lawyer

In all the examples on the left hand side, the lawyer is employed by someone. However, among the examples on the right hand side, only the last should include the employment relation as part of the interpretation. While he suggests a solution in general terms, Quillian admits that TLC as it stood could not handle these examples.

FAUSTUS has a better way of combining information from syntax and semantics. Both TLC and FAUSTUS suggest inferences by spreading markers from components of the input, and looking for collisions. The difference is that TLC used syntactic relations only as a filter to eliminate certain suggestions, while FAUSTUS incorporates the meaning of these relations into the representation before spreading markers. Even vague relations denoted by *for* and *of* are represented as full-fledged concepts, and are the source of marker-passing.

Trace output #1 below shows that FAUSTUS can find a connection between lawyer and client without the **for** relation, just like TLC. Markers originating at the representations of ‘‘lawyer’’ and ‘‘client’’ collide at the concept **employing-event**. The shape of the marker path indicates that this is a double elaboration path, and the suggested inference, that the lawyer employs the client, is eventually accepted.

In output #2 a non-marker-passing inference first classifies the **for** as an **employed-by** relation, because a lawyer is defined as a **professional-service-provider**, which includes an **employed-by** slot as a specialization of the **for** slot. This classification means the enemy must be classified as an **employer**. Once this is done, FAUSTUS can suggest the **employing-event** that mediates between an employee and an employer, just as it did in #1. Finally, in #3, the **of** is left with the vague interpretation **related-to**, so the enemy does not get classified as an employer, and no employment event is suggested.

Quillian #1

[1] Lawyer.

Rep: (LAWYER)

[2] Client.

Rep: (CLIENT)

Inferring: there is a EMPLOYING-EVENT such that
the CLIENT is the EMPLOY-ER of it and
the LAWYER is the EMPLOY-EE of it.
This is a DOUBLE-ELABORATION inference.
Quillian #2

[1] lawyer for the enemy

Rep: (LAWYER (FOR = THE ENEMY))

Inferring: a FOR of the LAWYER is the EMPLOYED-BY
This is a RELATION-CLASSIFICATION inference.

Inferring: there is a EMPLOYING-EVENT such that
the ENEMY is the EMPLOY-ER of it and
the LAWYER is the EMPLOY-EE of it.
This is a DOUBLE-ELABORATION inference.
Quillian #3

[1] enemy of the lawyer

Rep: (ENEMY (OF = THE LAWYER))

Inferring: a OF of the ENEMY is
probably a RELATED-TO
This is a RELATION-CONCRETION inference.

It should be noted that [Cha86] has a marker-passing mechanism that also improves on Quillian, and is in many ways similar to FAUSTUS. Charniak integrates parsing, while FAUSTUS does not, but FAUSTUS has a larger knowledge base (about 1000 concepts compared to about 75). Another key difference is that Charniak uses marker strength to make decisions, while FAUSTUS only uses markers to find suggestions, and evaluates them with other means.

5. Script Based Inferences

The SAM (Script Applier Mechanism) program [Cul78] was built to account for stories that refer to stereotypical situations, such as eating at a restaurant. A new algorithm was needed because Conceptual Dependency couldn't represent scripts directly. In KODIAK, there are no arbitrary distinctions between ''primitive acts'' and complex events, so **eating-at-a-restaurant** is just another event, much like **eating** or **walking**, except that it involves multiple agents and multiple sub-steps, with relations between the steps. Consider the following example:

The Waiter

[1] John was eating at a restaurant with Mary.

Rep: (EATING (ACTOR = JOHN)(SETTING = A RESTAURANT)
(WITH = MARY))

Inferring: a WITH of the EATING is probably
the ACCOMPANIER
because Mary fits it best.
This is a RELATION-CONCRETION inference.

Inferring: the EATING is a EAT-AT-RESTAURANT.
This is a CONCRETION inference.

[2] The waiter spilled soup all over her.

Rep: (SPILLING (ACTOR = THE WAITER)(PATIENT = SOUP)
(RECIPIENT = HER))

Inferring: there is a EAT-AT-RESTAURANT such that
the SOUP is the FOOD-ROLE of it and
the RESTAURANT is the SETTING of it.

This is a DOUBLE-ELABORATION inference.

Inferring: there is a EATING such that
the SOUP is the EATEN of it and
it is the PURPOSE of the RESTAURANT.
This is a DOUBLE-ELABORATION inference.

Inferring: there is a EAT-AT-RESTAURANT such that
the WAITER is the WAITER-ROLE of it and
the SOUP is the FOOD-ROLE of it.
This is a DOUBLE-ELABORATION inference.

The set of inferences seems reasonable, but it is instructive to contrast them with the inferences SAM would have made. SAM would first notice the word *restaurant* and fetch the restaurant script. From there it would match the script against the input, filling in all possible information about restaurants with either an input or a default value, and ignoring input that didn't match the script. FAUSTUS does not mark words like *restaurant* or *waiter* as keywords. Instead it is able to use information associated with these words only when appropriate, to find connections to events in the text. Thus, FAUSTUS could handle *John walked past a restaurant* without inferring that he ordered, ate, and paid for a meal.

6. Plan Based Inferences

In the previous section we saw that FAUSTUS was able to make what have been called ''script-based inferences'' without any explicit script-processing control structure. This was enabled partially by adding causal information to the representation of script-like events. The theory of plans and goals as they relate to story understanding, specifically the work of Wilensky [Wil78], was also an attempt to use causal information to understand stories that could not be comprehended using scripts alone. Consider story (4):

(4a) John was lost.
(4b) He pulled over to a farmer by the side of the road.
(4c) He asked him where he was.

Wilensky's PAM program processed this story as follows: from (4a) it infers that John will have the goal of knowing where he is. From that it infers he is trying to go somewhere, and that going somewhere is often instrumental to doing something there. From (4b) PAM infers that John wanted to be near the farmer, because he wanted to use the farmer for some purpose. Finally (4c) is processed. It is recognized that asking is a plan for knowing, and since it is known that John has the goal of knowing where he is, there is a match, and (4c) is explained. As a side effect of matching, the three pronouns in (4c) are disambiguated. Besides resolving the pronouns, the two key inferences are that John has the goal of finding out where he is, and that asking the farmer is a plan to achieve that goal.

In FAUSTUS, we can arrive at the same interpretation of the story by a very different method. (4a) does not generate any expectations, as it would in PAM, and FAUSTUS cannot find a connection between (4a) and (4b), although it does resolve the pronominal reference, because John is the only possible candidate. Finally, in (4c), FAUSTUS makes the two main inferences. The program recognizes that being near the farmer is related to asking him a question by a **precondition** relation (and resolves the pronominal references while making this connection). FAUSTUS could find this connection because both the asking and the being-near are explicit inputs. The other connection is a little trickier. The goal of knowing where one is was not an explicit input, but ''where he was'' is part of (4c), and there is a collision between paths starting from the representation of that phrase and another path starting from the asking that lead to the creation of the **plan-for** between John's asking where he is and his hypothetical knowing where he is.

The important conclusion, as far as FAUSTUS is concerned, is that both script- and goal-based processing can be reproduced by a system that has no explicit processing mechanism aimed at one type of story or another, but just looks for connections in the input as they relate to what is known in memory. For both scripts and goals, this involves defining situations largely in terms of their causal structure.

7. Coherence Relation Based Inferences

In this section we turn to inferences based on coherence relations, as exemplified by this example proposed by Kay and Fillmore [Kay81]:

(5) A hiker bought a pair of boots from a cobbler.

From the definition of **buying** one could infer that the hiker now owns the boots that previously belonged to the cobbler and the cobbler now has some money that previously belonged to the hiker. However, a more complete understanding of (5) should include the inference that the transaction probably took place in the cobbler's store, and that the hiker will probably use the boots in his avocation, rather than, say, give them as a gift to his sister. The first of these can be derived from concretion inferences once we have described what goes on at a shoe store. The problem is that we want to describe this in a neutral manner − to describe not ''buying at a shoe store'' which would be useless for ''selling at a shoe store'' or ''paying for goods at a shoe store'' but rather the general ''shoe store transaction.'' This is done by using the **commercial-event** absolute, which dominates **store-transaction** on the one hand, and **buying, selling** and **paying** on the other. Each of these last three is also dominated by **action**. Assertions are made to indicate that the **buyer** of **buying** is both the **actor** of the **action** and the **merchant** of the **commercial-event**. The next step is to define **shoe-store-transaction** as a kind of **store-transaction** where the **merchandise** is constrained to be **shoes**. With that done, we get the following:

The Cobbler and the Hiker

[1] A cobbler sold a pair of boots to a hiker.

Rep: (SELLING (ACTOR = A COBBLER)(PATIENT = A BOOT)
(RECIPIENT = A HIKER))

Inferring: the SELLING is a SHOE-STORE-TRANSACTION.
This is a CONCRETION inference.

Inferring: there is a WALKING such that
it is the PURPOSE of the BOOT and
the HIKER is the OBJECT-MOVED of it.
This is a DOUBLE-ELABORATION inference.

The program concludes that a selling involving shoes is a shoe store transaction, and although it was not printed, this means that it takes place in a shoe store, and the seller is an employee of the store. The second inference is based on a collision at the concept **walking**. The purpose of boots is walking, and the walking is to be done by the hiker, because that's what they do. Note that the representation is not sophisticated enough to distinguish between actual events and potential future events like this one.

8. Unexpected Inferences

One hallmark of an AI program is to generate output that was not expected by the program's developer. The following text shows an example of this:

The President

[1] The president discussed Nicaragua.

Rep: (DISCUSSING (ACTOR = THE PRESIDENT)
(CONTENT = NICARAGUA))

[2] He spoke for an hour.

Rep: (TALKING (ACTOR = HE) (DURATION = AN HOUR))

Inferring: 'HE' must be a PERSON,
because it is the TALKER
This is a RELATION-CONSTRAINT inference.

Inferring: 'HE' refers to the PRESIDENT.
This is a REFERENCE inference.

Inferring: the NICARAGUA is a COUNTRY such that
it is the HABITAT of 'HE' and
it is the COUNTRY of the PRESIDENT.

This is a DOUBLE-ELABORATION inference.

Inferring: the TALKING refers to the DISCUSSING.
This is a REFERENCE inference.

This example was meant to illustrate action/action co-reference. The **talking** in the second sentence refers to the same event as the **discussing** in the first sentence, but neither event is explicitly marked as definite or indefinite. FAUSTUS is able to make the inference that the two actions are co-referential, using the same mechanism that works for pronouns. The idea of treating actions under a theory of reference is discussed in [Loc80]. FAUSTUS correctly finds the coreference between the two actions, and infers that 'he' refers to the president.

But FAUSTUS infers that Nicaragua is the president's home or habitat and is the country of his constituency. This makes a certain amount of sense, since presidents must have such things, and Nicaragua is the only country mentioned. Of course, this was unexpected; we interpret the text as referring to the president of the United States because we are living in the U.S., and our president is a salient figure. Given FAUSTUS's lack of context, the inference is quite reasonable.

9. Conclusion

We have shown that a general marker-passing algorithm with a small number of inference classes can process a wide variety of texts. FAUSTUS shifts the complexity from the algorithm to the knowledge base to handle examples that other systems could do only by introducing specialized algorithms.

References