SOME EXAMPLES OF DISAMBIGUATION THROUGH DEDUCTION

by

Erik Sandewall

UPPSALA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCES

DATALOGILABORATORIET
Inst f informationsbehandling
Uppsala universitet

1973-11-12 DLU 73/27

SOME EXAMPLES OF DISAMBIGUATION THROUGH DEDUCTION

by

Erik Sandewall

Abstract: The paper follows a general approach where an extended predicate calculus is used as a high-level notation for describing the data base structure and major parts of the programs in a "language understanding" system. In the present paper, a calculus which enables some simple question-answering, is assumed to be known. This formalism is extended so that plausibility information can be stored in the data base, and so that disambiguation of phrases using plausibility information can be performed by deduction.

This is a slightly modified version of the paper "Deductive search in a semantic net" presented at the symposium "Organismic Information Processing", organized by the Gesellschaft für Psychologie der DDR in Berlin on 11-14 September, 1973.

The research reported here was supported in part by the Swedish Natural Science Research Council (Contract Dnr 2654-008).

This paper describes through an example how one essential function in a "computer understanding" system, namely disambiguation using plausibility information, may be described in a formal manner. It is argued that the formal description is useful as a precise and compact model of the computer program. We use a subset of the PCF-2 formalism, described by the author in a previous report,[1] with some minor changes which have been made since that paper was written. Most of the material in the present paper has been implemented and experimented with in a program called SEPAC, which is an implementation of PCF-2.

Disambiguation experiments of similar phrases have preiouvsly been performed by Quillian[2]. The purpose of the work reported here has not primarily been to improve the quality of the judgements in choosing the right meaning of a phrase, but instead to obtain a more precise formulation of how the judgement is done.

Overall organization of system. Our general framework is that of a system which consists of both program(s) and a data base (= semantic net), and whose knowledge is accumulated in the latter. The structure of the data base has been specified on a logical level in reference (1).

The program part of the system processes input sentences. Each phrase which is input to the system passes in principle through the following steps:

1. Parsing. Converts the sentence to a tree structure.
2. Assimilation. Relates the sentence to the "data base" or "semantic net" of the receiving system. This involves e.g. finding the referents of pronouns and of noun phrases starting with a definite article. It also involves attempts to disambiguate words and phrases and to reconstruct them so that their intended meaning becomes more explicite.
3. Response to the sentence. In a question-answering context, this means storing assertions and answering questions. In a more general context, it requires a procedure which maintains a model of the conversation or input text.

These steps are assumed to be "locally sequential", i.e. each sub-tree in the parse tree runs through them in this order. However, some sub-trees on lower levels may proceed through assimilation before other sub-trees have yet been generated. Moreover, backtracking must be possible when the program has gone astray, and it then upsets the sequence of the above steps.

Generation of the output sentence might require an extra step. On the other hand, for the purpose of some experiments, some of the given steps may be shortcut or simplified. In particular, for the test runs with the material in this paper, the "parsing" program was very simplified, and no backtracking across step boundaries was required.

To illustrate what is done in these successive steps, let us consider one example, the half-sentence

The hand is badly damaged (, and ...)

This sentence might be preceded by e.g.

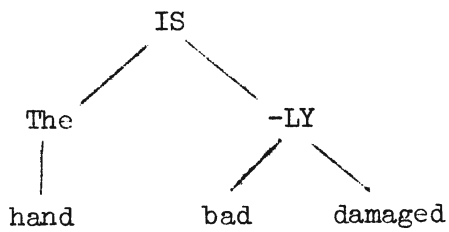
John has been severely hurt in the accident.

or

This clock needs repairing.

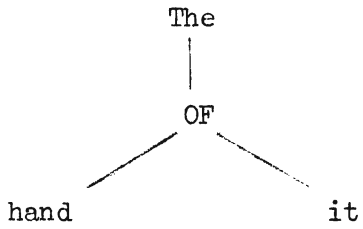
Clearly, "hand" in the considered sentence has a different meaning in the two cases. The processing of the sentence goes roughly as follows:

1. The sentence is parsed into



In fact, "damaged" goes into a sub-tree, but let us bypass that matter.

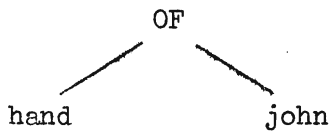
2. The sub-tree under the node "The" is assimilated. The operator associated with "The" decides (since no hand has previously been mentioned) that "The hand" means "The hand of it" i.e. it calls for the assimilation of



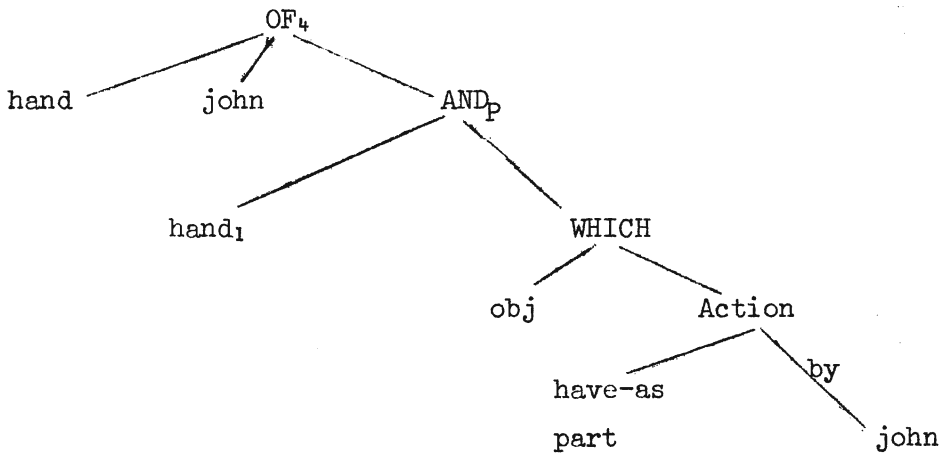
(More precisely, rather than "it", we here use an artificial pronoun which is the union of "me", "you", "he", "she", "it", etc.)

3. Starting from below again, "it" is assimilated and found to refer to John or to clock, in the two examples of predecessor sentences. (One can of course construct examples where several hypotheses have to be tried at this point).

4. The sub-tree



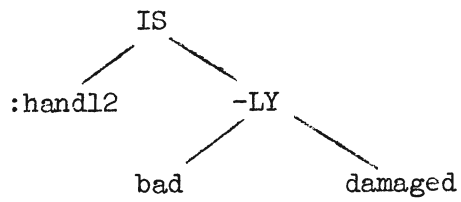
(we assume from now on the first predecessor sentence) is assimilated, and it is decided that it must be taken as "hand$_1$ which John has as part", where "hand$_1$" is the primary meaning of "hand". Assimilation returns a more complex structure, roughly



which contains both the original information and the preferred interpretation. OF_4 in the tree is one of the internal forms corresponding to the English "of"; AND_P forms the intersection of two properties, namely that of being a $hand_1$, and that of being something which John has as part.

5. The sub-tree headed by "The" now contains the OF_4 tree shown on last page. It proceeds to the assimilation step. A new node :handl2 (say) is introduced in the semantic network, and intended to stand for the damaged hand. It is provided with the adequate connections, i.e. being a $hand_1$ and being a part of John.

6. The tree for the whole sentence now has the form



Assimilation leaves the -LY sub-tree and the whole tree in principle unaffected. In the "response" step, the assertion is presumably accepted, i.e. the node :handl2 is provided with additional arcs which show that the hand is badly damaged.

These examples have illustrated the points we shall consider in this paper, namely:

(a) the procedure associated with the definite article "the",

(b) the disambiguation of "of" into "which is a part of".

(c) the disambiguation of "hand" into "$hand_1$".

The definite article is however covered only as a prerequisite for the two latter points, rather than for its own interest.

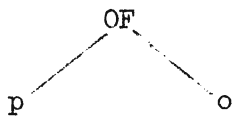
The definite article. We associate with the definite article (like with a number of other "function" words) a procedure which is used in the assimilation step. Such a procedure takes as argument(s) the sub-tree(s) which appear under the node of the function word. It is also able to find out what is above and beside it in the tree, although that was not needed for the example above.

The present, incomplete definition of the "The" procedure takes essentially one argument p, which is to denote a "property" (as defined in reference (1)), e.g. a typical noun, and does the following:*

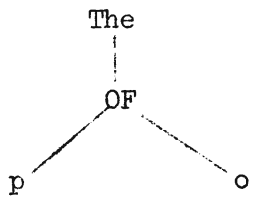
1. If the argument p is an OF-expression (i.e. a tree with an OF_i node at its root), then the procedure looks for some object in the data base which has the property p. (The question may arise when the object should have the property. The procedure for "the" in fact has a second optional argument, which should be the situation in which the desired object has the property). If it is found, that object is returned, otherwise a new object o * is created, assigned the property p, and returned.

2. A list of lately mentioned objects is scanned to find some which has (have) the property p. If one is found, it is returned. If several are found, a choice is made.

3. If some object o is in "focus" from the preceding context, then it is checked whether



has a plausible interpretation (see below). If so,



is assimilated and returned

4. If neither (2) nor (3) succeedes, then a new object node o* is created as in case (1).

This definition has been sufficient for our experiments, but clearly it misses some cases, notably

---

* Our definition is similar to the one used in reference (3), although it was arrived at independently.

5. Sometimes when p is an OF-expression, the object which has the property p shall not be retrieved, but instead "The" shall return its argument, e.g. in expressions involving modals like "He wondered who is the author of Waverly".

6. Sometimes "the" is used to denote the second or later occurence of what should be represented internally as a variable. Example: "If a prince kisses a frog, then the frog turns into a beautiful princess".
   To recognize such cases, the procedure for "The" must be able to look around in the tree above it.

The disambiguation of "OF". The assimilation routine associated with OF consists of a number of cases, just like the routine for "The". Most of these cases are also used by the routine for the genitive 's. Some specialized cases (exemplified by "color of", "size of", "name of", etc.) as well as phrases that denote an event ("the rise of", "the sight of") are sorted out using essentially word-class information. Sometimes backtracking may be needed. - There remains those OF-expressions which denote a new object, including a person (e.g. "John's father", "John's car"), opposed to a property or other abstract entity ("color of", "size of"). Among these cases, some are considered to be "functional", such as "John's father", but not "John's car". One can treat "father of" as a function of (essentially) one argument, whereas for "car of" that would not be natural. The functional case is (presently) recognized by a flag on the functional noun (such as "father").

The non-functional case, which is what we abvoe referred to as OF_4, occurs in phrases like

John's hands
John's wallet
John's lawyer
a row of bricks
a table of wood

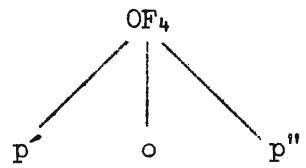
These phrases have obvious interpretations, i.e.

the hands that are physically part of John
the wallet that John owns
the lawyer who serves John
a row consisting of bricks
a table made from wood

We believe that the number of possible relations (part of, owns, etc.) is small and for practical purposes can be kept fixed. Our task is to choose between them in each particular case. The procedure for OF in the OF_4 cases adds one more branch under the OF node, besides the two old branches that correspond to the phrases before and after OF. The new branch contains a list of candidate interpretations.

This candidate list may be used as follows: if the node above the "OF" node is a "The" node (as in the example in our initial example), then the procedure for "The" obtains an argument p which has the structure

p =



where p" is a list of candidates. If this list contains only one alternative, then the routine which looks for an object which has the property p will be satisfied if it finds one which has the property p". Similarly, the routine which assigns the property p to an object will also assign to it the property p". These are the routines that were mentioned in the specification of the "The" procedure. - If several candidates are on the list, then the system postpones the choice between them. (In the presently existing program, the choice is then in fact never made).

Thus in principle, the extra branch in the tree serves to communicate the candidate list from one part of the program, namely the "OF" routine, through e.g. the "The" routine, to the routines for storage and retrieval of properties of objects. This way of doing it is considered more sound than by using extra arguments to procedures, and similar makeshift devices.

The "The" procedure does not need to know what happens, and other procedures which also need to store and retrieve properties of objects (such as the "An" procedure) may appear in the place of "The".

The candidate list is computed in the following way: For given arguments p´ and o, the "OF" procedure goes through the candidate verbs and forms trees for the actions

o owns a p´
a p´ is physically a part of o
a p´ serves o
a p´consists of o

e.g.

John owns a knee
a knee is physically part of John

etc., and then checks which of these are plausible. Plausibility can be achieved through specialization of a more general action. Thus the action "a knee is physically a part of John" is a specialization of "a bodypart is physically a part of a person", assuming of course that the data base also contains information such as "a knee is a bodypart". If the more general action has been marked as plausible, then the specialization is also considered to be plausible.

We notice that if the system has been told that one rather general action is to be considered as plausible, then this may be used for a large number of cases, but also that several plausibility statements may be needed for a single verb. Thus the data base should know that "a root is physically a part of a plant" and "a battery is physically a part of a car" are plausible, as well as the generalization about bodyparts that is used in the example.

The procedure that has just been described can be more compactly and precisely specified using the notation of predicate calculus and the conventions and definitions of reference (1). We shall not be able to repeat that material here, but must assume from here on that it is

known.* However, we believe that it should be possible to read most of the remainder of the paper without reference to (1).

We shall need the following relations:

MAYMEAN: property X property

This relation is exemplified by

hand MAYMEAN $hand_1$

and in general connects an expression and a possible reading of it. The reasons for having such a relation shall be further discussed below.

Plausible: action

This relation of one argument marks an action as plausible.

SUBACT: action X action

This relation connects a specialized and a more general action, such as "John has a $hand_1$ as part" and "a person has a bodypart as part".

Ofmeaning: action X casefunction X casefunction

This relation is used to mark the possible meanings of the OF_4 case of OF, as discussed above. Thus as (p OF o) may be interpreted as "a p owned by o", we shall have

Ofmeaning(own,obj,by)

where by is the "case indicator" which marks him who owns, and obj marks the thing being owned. (The proper choice of case indicators e.g. for the verb "to own" is not an issue here).

We have the axioms

$$\text{Plausible}(\text{Action}(a,\langle c,p,c',o\rangle)) \wedge \text{Ofmeaning}(a,c,c') \rightarrow (p \text{ OF } o) \text{ MAYMEAN } (p \text{ AND}_p (c \text{ WHICH } \text{Action}(a,\langle c', o\rangle))) \quad (1)$$

---

* We have modified the notation as follows, compared to the PCF-2 report (1):

(a) The situation argument is consistently omitted. We define
Holds(a) $\equiv (\forall s)$ a INN s
and assume that all objects "exist" in the situation we work in.

(b) Spelling changes: ANDPROP → AND_p, Expected → Plausible.

(c) Instead of the infix case functions, we use a function Action so that
a c o c' o' ... = Action(a, <c,o,c',o',...>)
where c,c',... are case functions. This is closer to the implementation in the SEPAC program.

(d) For simplification, we treat OF-expressions as properties, and have
(p OF o IN s) VAL o' $\equiv$ o' IS (p OF o) IN s

a SUBACT a' Λ Plausible(a') → Plausible(a) (2)

plus obvious axioms for SUBACT, WHICH, etc. that were given in (1). We shall only remind the reader that the axioms there imply*

o IS2 (c WHICH Action(a,<...1...>)) →
Holds(Action(a,<c,o,... 1...>)) (3)

where IS2 is a binary relation which says that the object has the property, and Holds marks an action as occurring in the world. (In this paper we do not care which world).*

With this machinery, we can specify the procedure for assimilating OF(p,o) in the OF_4 case as follows: Find all p' for which it can be proved

(p OF o) MAYMEAN p'

and add the list of those p' as a third branch under the OF node. Rename the node as OF_4.

A deduction which shows how this works for the example of "the hand of John" will be given below.

Likewise, the deductions that are done in the routines for storing and retrieving the property of an object can be characterized (not completely formally) by the rule

p MAYMEAN p' Λ (only one such p' for that p) →
o IS2 p ≡ o IS2 p' (4)

Disambiguation of "hand" in the example. We want our system to implement a "continuous", rather than an "atomistic" view of word meaning. That is, we do not want to hypothesize that each word has a small, fixed number of simple, well-defined, conceptually indivisible "meanings". Instead, we view the "meaning" of a word as a "blob", which may be sub-divided by various criteria into more specialized meanings, which in their turn may be sub-divided. As one continues to sub-divide, more and more properties become characteristic of the remaining sub-meaning.

---

* We assume tacitly that the object o "exists in" the situation or "world".

As a consequence, we want in principle that both the original word, and its various sub-divisions, shall be represented as nodes in the network. Thus we would want to have nodes for "hand", "hand of person" (= "primary meaning of 'hand' "), "hand of clock", "hand of ape", "hand of robot", etc.

The relation which connects a word and one of its sub-meanings is MAYMEAN, which we have used for phrases. Thus if $\underline{\text{hand}_1}$ is the node for the primary meaning of "hand", we would have in the network

hand MAYMEAN hand_1

We also have additional axioms

p MAYMEAN p' $\wedge$ p' MAYMEAN p" $\rightarrow$ p MAYMEAN p" (5)

and

p MAYMEAN p' $\rightarrow$ (p OF o) MAYMEAN (p' OF o) (6)

The latter axiom is an instance of an obvious axiom schema.

Let us exemplify these axioms by working out the formal deduction whereby "hand of John" is disambiguated. We assume that the data base contains

hand_1 SUB_p bodypart (11)

hand MAYMEAN hand_1 (12)

Plausible(Action(haveaspart,<by person obj bodypart>)) (13)

john IS2 person (14)

Ofmeaning(haveaspart,obj,by) (15)

The deduction then proceeds as follows (successive conclusions are numbered to the right, and their support is indicated to the left):

6,11 (hand OF john) MAYMEAN (hand_1 OF john) (16)

2,11,13, 14 Plausible(Action(haveaspart,<by john obj hand_1>)) (17)

1,15,17 (hand$_1$ OF john) MAYMEAN (hand$_1$ AND$_P$
(obj WHICH Action(haveaspart,<by john>))) (18)

5,16,18 (hand OF john) MAYMEAN
(hand$_1$ AND$_P$
(obj WHICH Action(haveaspart,<by john>)))) (19)

This deduction mirrors rather closely what is done in the procedure for "OF", as specified above. The deduction leading to (17) contains in fact several steps, but is trivial (although for some other examples the corresponding deduction might be less trivial).

In the assimilation of the phrase "the hand (of John)", the procedure for "The" as specified above may introduce a new constant :hand for which it asserts

:hand IS2 (hand OF john) (20)

This statement enables the following deduction:

4,19,20 :hand IS2 (hand$_1$ AND$_P$
(obj WHICH Action(haveaspart,<by john>))) (21)

21 :hand IS2 hand$_1$ (22)

whereby it has been determined that :hand is a hand in the primary sense of the word. Also:

22 :hand IS2 (obj WHICH Action(haveaspart,<by john>)) (23)

3,23 Holds(Action(haveaspart,<by john obj :hand>)) (24)

whereby it has been determined that :hand is physically part of John.

In these deductions, supporting axioms which have not been listed in this paper, but which occur in reference (1), have not been indicated in the left-hand column. The constant haveaspart should in fact be a new expression, or set equal to such an expression, in a fully developed system.

In these deductions, supporting axioms which have not been listed in this paper, but which occur in reference (1), have not been indicated in the left-hand column. The constant haveaspart should in fact be a new expression, or set equal to such an expression, in a fully developed system.

Discussion. We have tried to show, through the selected example, that the operation of a relatively complex program can be characterized with high precision using the notation of predicate calculus (with some carefully chosen amendments) and a relatively small number of axioms. The machinery shown in this paper makes heavy use of material in reference (1) which is useful not only for disambiguation, but also for other purposes such as question-answering. The following items are needed specifically for the disambiguation task:

- The relations MAYMEAN, Ofmeaning, and Plausible
- Axioms 1,2,4,5,6 (where 6 should be generalized into an axiom schema)
- Plausibility information in the data base, such as in axiom 13
- Information about the possible meanings of OF-expressions, such as in axiom 15.

Of these, the items that have to do with MAYMEAN and Plausible can be used for other disambiguation as well (such as composite nouns: "glass bottle", "whisky bottle", "horse shoe", "alligator shoe", etc., or for finding referents of pronouns). The relation Ofmeaning and the items associated with it can of course only be used for the purpose discussed in this paper.

We have chosen this particular example, disambiguation using common-sense information, in order to emphasize that the method of modelling a program through a deductive system is not limited to the manipulation of logical truth, but can also be used for operating on heuristic information. The deduction itself is of course precise, just like the operation of the program that we wish to model, but the assumptions and the conclusions in the deduction have a heuristic content.

The reason for doing this formal exercise is to structure the problem. It is useful before one writes the program: the formalization is a blueprint for the program. It specifies what deductions the program must make, but does not specify exactly how they shall be done - that information must be added when the program is written. The formalization is also useful afterwards, as a documentation of the program, an aid to debugging, and a means to compare its capacity with other programs.

It might be argued that the example here is all very well, but what happens when several plausibility statements have been used in a deduction? Presumably at some point one would like to cut off the deduction and say "now the plausibility has been eroded so much that we do not wish to continue". How could this be done formally? One answer is obvious: in the same way as a program would do it, i.e. by keeping track of the accumulated uncertainty as the deduction goes on. For a simple solution, the relation Plausible could have an additional, numerical argument, which is an estimate of the plausibility.

The formal description of the program should enable the reader to determine what the program can do, by performing the deductions manually. The present paper is not a formal description in this sense, partly because we have chosen to omit some formal detail, and partly because we have had to refer to reference (1) to keep the paper small, while at the same time our notation has been modified somewhat since reference (1) was written. However, it is hoped that the degree of precision in this paper was sufficient to convince the reader that 100 percent precision in a useful program description can be achieved.

In the specification of the OF procedure and at some other points, we have used some informal comments together with the formal machinery. We foresee the development and use of a set of formal operators, which can be used in the specification of procedures associated with "The", "OF", etc., but we want to postpone that work until we have more experience with what operations are needed in such procedures.

References

1. Erik Sandewall
PCF-2, a first-order calculus for representing conceptual information
Mimeographed report, Uppsala University, Computer Science Department, February 1972

2. Ross Quillian
The teachable language comprehender: a simulation program and theory of language
Comm. ACM, Vol. 12, No. 8 (August, 1969)

3. David Rumelhart and Donald Norman
Active Semantic Networks as a Model of Human Memory
Proc. Third International Joint Conference on Artificial Intelligence
Stanford Research Institute, 1973