ANALYSIS OF CONJUNCTIONS IN A RULE-BASED PARSER

Leonardo Lesmo and Pietro Torasso

Dipartimento di Informatica - Universita' di Torino
Via Valperga Caluso 37 - 10125 Torino (ITALY)

ABSTRACT

The aim of the present paper is to show how a rule-based parser for the Italian language has been extended to analyze sentences involving conjunctions. The most noticeable fact is the ease with which the required modifications fit in the previous parser structure. In particular, the rules written for analyzing simple sentences (without conjunctions) needed only small changes. On the contrary, more substantial changes were made to the exception-handling rules (called "natural changes") that are used to restructure the tree in case of failure of a syntactic hypothesis. The parser described in the present work constitutes the syntactic component of the FIDO system (a Flexible Interface for Database Operations), an interface allowing an end-user to access a relational database in natural language (Italian).

INTRODUCTION

It is not our intention to present here a comprehensive overview of the previous work on coordination, but just to describe a couple of recent studies on this topic and to specify the main differences between them and our approach.

It must be noticed, however, that both systems that will be discussed use a logic grammar as their basic framework, so that we will try to make the comparison picking out the basic principles for the manipulation of conjunctions, and disregarding the more fundamental differences concerning the global system design. It is also worth pointing out that, although the present section is admittedly incomplete, most of the systems for the automatic analysis of natural language do not describe the methods adopted for the interpretation of sentences containing conjunctions in great detail. Therefore, it is reasonable to assume that in many of these systems the conjunctions are handled only by means of specific heuristic mechanisms.

A noticeable exception is the SYSCONJ facility of the LUNAR system (Woods, 1973): in this case, the conjunctions are handled by means of a para-syntactic mechanism that enables the parser to analyze the second conjunct assuming that it has a structure dependent on the hypothesized first conjunct. The main drawback of this approach is that the top-down bias of the ATNs does not allow the system to take advantage of the actual structure of the second conjunct to hypothesize its role. In other words, the analysis of the second conjunct acts as a confirmation mechanism for the hypothesis made on the sole basis of the position where the conjunction has been found. Consequently, all the various possibilities (of increasing levels of complexity) must be analyzed until a match is found, which involves an apparent waste of computational resources.

The solution proposed in the first of the two systems we will be discussing here is quite similar. It is based on Modifier Structure Grammars (MSG), a logic formalism introduced in (Dahl & McCord, 1983), which constitutes an extension of the Extraposition Grammar by F. Pereira (1981). The conjunctions are analyzed by means of a special operator, a "demon", that deals with the two problems that occur in coordination: the first conjunct can be "interrupted" in an incomplete status by the occurrence of the conjunction (this is not foreseeable at the beginning of the analysis) and the second conjunct must be analyzed taking into account the previous interruption point (and in this case, mainly because the second conjunct may assume a greater number of forms, some degree of top-down hypothesization is required).

The first problem is solved by the "backup" procedure, which forces the satisfaction (or "closure" in our terms) of one or more of the (incomplete) nodes appearing in the so-called "parent" stack. The choice of the node to which the second conjunct must be attached makes the system hypothesize (as in SYSCONJ) the syntactic category of the second conjunct and the analysis can proceed (a previous, incomplete constituent would be saved in a parallel structure, called "merge stack" that would be used subsequently to complete the interpretation of the first conjunct).

Apart from the considerable power offered by MSGs for semantic interpretation, it is not quite clear why this approach represents an advance with respect to Woods' approach. Even though the analysis times reported in the appendix of (Dahl & McCord, 1983) are very low, the top-down bias of

----

The research project described in this paper has partially been supported by the Ministero della Pubblica Istruzione of Italy, MPI 40% Intelligenza Artificiale.

MSGs produces the same problems as ATNs do. The "backup" procedure, in fact, chooses blindly among the alternatives present in the parent stack (this problem is mentioned by the authors). A final comment concerns the analysis of the second conjunct: since the basic grammar aims at describing "normal" English clauses, it seems that the system has some trouble with sentences involving "gapping" (see the third section). In fact, while an elliptical subject can be handled by the hypothesization, as second conjunct, of a verb phrase (this is the equivalent of treating the situation as a single sentence involving a single subject and two actions, and not as two coordinated sentences, the second of which has an elliptical subject; it seems a perfectly acceptable choice), the same mechanism cannot be used to handle sentences with an elliptical verb in the second conjunct.

The last system we discuss in this section has been described in (Huang, 1984). Though it is based, as the previous one is, on a logic grammar, it starts from a quite different assumption: the grammar deals explicitly with conjunctions in its rules. It does not need any extra-grammatical mechanisms but the positions where a particular constituent can be erased by the ellipsis have to be indicated in the rules. Even though the effort of reconstructing the complete structure (i.e. of recovering the elliptical fragment) is mainly left to the unification mechanism of PROLOG, the design of the grammar is rendered somewhat more complex.

The fragment of grammar reported in (Huang, 1984) gives the impression of a set of rules "flatter" than the ones that normally appear in standard grammars (this is not a negative aspect; it is a feature of the ATNs too). The "sentence" structure comprises a NP (the subject, which may be elliptical), an adverbial phrase, a verb (which also may be elliptical), a restverb (for handling possible previous auxiliares) and a rest-sentence component. We can justify our previous comment on the increased effort in grammar development by noting that two different predicates had to be defined to account for the normal complements and the structure that Huang calls "reduced conjunction", see example (13) in the third section. Moreover, it seems that a recovery procedure deeply embedded within the language interpreter reduces the flexibility of the design. It is difficult to realize how far this problem could affect the analysis of more complex sentences (space contraints limited the size of the grammar reported in the paper quoted), but, for instance, the explicit assumption that the absence of the subject makes the system retrieve it from a previous conjunct, seems too strong. Disregarding languages where the subject is not always required (as it is the case for Italian), in English a sentence of the form "Go home and stay there till I call you" could give the parser some trouble.

In the following we will describe an approach that overcomes some of the problems mentioned above. The parser that will be introduced constitutes the syntactic component of the FIDO system (a Flexible Interface for Database Operations), which is a prototype allowing an end-user to interact in natural language (Italian) with a relational data base. The query facility has been fully implemented in FRANZ LISP on a VAX-780 computer. The update operations are currently under study. The various components of the system have been described in a series of papers which will be referenced within the following sections. The system includes also an optimization component that converts the query expressed at a conceptual level into an efficient logical-level query (Lesmo, Siklossy & Torasso, 1985).

OVERALL ORGANIZATION OF THE PARSER

In this section we overview the principles that lie at the root of the syntactic analysis in FIDO. We try to focus the discussion on the issues that guided the design of the parser, rather than giving all the details about its current implementation. We hope that this approach will enable the reader to realize why the system is so easily extendible. For a more detailed presentation, see (Lesmo & Torasso, 1983 and Lesmo & Torasso, 1984).

The first issue concerns the interactions between the concept of "structured representation of a sentence" and "status of the analysis". These two concepts have usually been considered as distinct: in ATNs, to consider a well-known example, the parse tree is held in a register, but the global status of the parsing process also includes the contents of the other registers, a set of states identifying the current position in the various transition networks, and a stack containing the data on the previous choice points. In logic grammars (Definite Clause Grammars (Pereira & Warren, 1980), Extraposition Grammars (Pereira, 1981), Modifier Structure Grammars (Dahl & McCord, 1983)) this book-keeping need not be completely explicit, but the interpreter of the language (usually a dialect of PROLOG) has to keep track of the binding of the variables, of the clauses that have not been used (but could be used in case of failure of the current path), and so on. On the contrary, we tried to organize the parser in such a way that the two concepts mentioned above coincide: the portion of the tree that has been built so far "is" the status of the analysis. The implicit assumption is that the parser, in order to go on with the analysis does not need to know how the tree was built (what rules have been applied, what alternatives there were), but just what the result of the previous processing steps is[1].
Of course, this assumption implies that all information present in the input sentence must also be

----

[1] We must confess that this assumption has not been pushed to its extreme consequences. In some cases (see (Lesmo & Torasso, 1983) for a more detailed discussion) the backtracking mechanism is still needed, but, although we are not unable to provide experimental evidence, we believe that it could be substituted by diagnostic procedures of the type discussed, with different purposes and within a different formalism, in (Weischedel & Black, 1980).

present in its structured representation; actually, what happens is that new pieces of information, which were implicit in the "linear" input form, are made explicit in the result of the analysis. These pieces of information are extracted using the syntactic knowledge (how the constituents are structured) and the lexical knowledge (inflectional data).

The main advantage of such an approach is that the whole interpretation process is centered around a single structure: the dependency structure of the constituents composing the sentence. This enhances the modularity of the system: the mutual independence of the various knowledge sources can be stated clearly, at least as regards the pieces of knowledge contained in each of them; on the contrary, the control flow can be designed in such a way that all knowledge sources contribute, by cooperating in a more or less synchronized way, to the overall goal of comprehension (see fig.1).

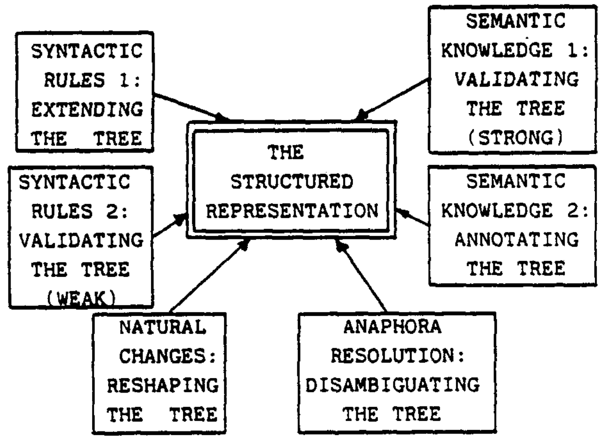


Fig.1: A single structure is the basis of the whole interpretation process.

A side-effect of the independence of knowledge sources mentioned above is that there is no strict coupling between syntactic analysis and semantic interpretation, contrarily to what happens, for instance, in Augmented Phrase Structure Grammars (Robinson, 1982). This means that there is no one-to-one association between syntactic and semantic rules, a further advantage if we succeed in making the structured representation of the sentence reasonably uniform. This result has been achieved by distinguishing between "syntactic categories", which are used in the syntactic rules to build the tree, and "node types", whose instantiations are the elements the tree is built of.[2] Since the number of syntactic categories (and of syntactic rules) is considerably larger than the number of node types (6 node types, 22 syntactic categories, 61 rules), then some general constraints and interpretation rules may be expressed in a more compact form. Without entering into a discussion on semantic interpretation, we can give an example using the rules that validate the tree from a syntactic point of view (SYNTACTIC RULES 2 in fig.1). One of these rules specifies that the subject and the verb of the sentence must agree in number. On the other hand, the subject can be a noun, a pronoun, an interrogative pronoun, a relative pronoun: each of them is associated with a different syntactic category, but all of them will finally be stored in a node of type REF (standing for REFerent); independently of the category, a single rule is used to specify the agreement constraint mentioned above.

Let us now have a look at the box in fig.1 labelled "SYNTACTIC RULES 1: EXTENDING THE TREE". The rules that are logically contained in that box are the primary tool for performing the syntactic analysis of a sentence. Each of them has the form:

PRECONDITION ---> ACTION

where PRECONDITION is a boolean expression whose terms are elementary conditions; their predicates allow the system to inspect the current status of the analysis, i.e. the tree (for instance: "What is the type of the current node?", "Is there an empty node of type X?"); a look-ahead can also be included in the preconditions (maximum 2 words). The right-hand side of a rule (ACTION) consists in a sequence of operations; there are two operators:

CRLINK (X,Y)
which creates a new instance of the type X and links it to the nearest node of type Y existing in the rightmost path of the tree (and moving only upwards)

FILL (X,V)
which fills the nearest node (see above) of type X with the value V (which in most cases coincides with the lexical data about the current input word).

The rules are grouped in packets, each of which is associated with a lexical category. It is worth noting that the choice of the rule to fire is non-deterministic, since different rules can be executed at a given stage. On the other hand, the non-determinism has been reduced by making the preconditions of the rules belonging to the same packet mutually exclusive; consequently, the status is saved on the stack only (but not always) if the input word is syntactically ambiguous. Note that nothing prevents there being exceptions to this rule. For example, in English the past indicative and the past participle usually have the same form: in this case, two different rules of the VERB packet could be activated if the context allows for both interpretations.

----

[2] Six node types have been introduced (each node is actually a complex data structure): REL (RELations, mainly verbs), REF (REFerents, nouns, pronouns, etc.), CONN (CONNectors, e.g. prepositions), DET (DETerminers), ADJ (ADJectives), and MOD (MODifiers, mainly adverbs). Beyond these six types, a special node (TOP) has been included to identify the main verb(s) of the sentence.

Currently, the syntactic categories of an ambiguous word are ordered manually in the lexicon; since the "first" rule is determined by that order, the selection of the rule to execute depends only on the choices made by the designer of the lexicon. Some experiments have been made to include a weighting mechanism, which should depend both on the syntactic context and on the semantic knowledge (Lesmo & Torasso, 1985).

A second "syntactic" box appears in fig.1. It refers to rules that are, in a sense, weaker than the rules of the set discussed above. The rules of the first set are aimed at defining acceptable syntactic structures, where "acceptable" is used to mean that the resulting structure is semantically interpretable (for instance, a determiner cannot be used to modify an adjective). On the contrary, the rules of the second set specify which of the meaningful sentences are well formed; in particular, they are used to check gender and number agreement and the ordering of constituents (e.g. the fact that in English an adjective should occur before the noun it refers to, whereas this is not always the case in Italian). The separation between the rules of the two sets is the feature that makes the system robust from a syntactic point of view (see (Lesmo & Torasso, 1984) for further details).

It may be noticed that, in fig.1, both the second set of syntactic rules we have just discussed and a part of the semantic knowledge have the purpose of "validating the tree". Independently of the fact that the second-level syntactic constraints can be broken (they are "weak" constraints), whilst the semantic constraints can not (they are "strong" constraints), some action must be performed when the structure hypothesized by the first-level rules does not match those constraints. The task of the rules called "natural changes" (see fig.1) is to restructure the tree in order to provide the parser with a new, "correct" structure. We will not go into further details here, since the natural changes (in particular the one concerning the treatment of conjunctions) will be discussed in a following section; however, in order to give a complete picture of the behavior of the parser, we must point out that the natural changes can fail (no correct structure can be built). In this case, the parser returns to the original structure and issues a warning message, if the trigger of the natural changes was a weak constraint; otherwise (semantic failure) it backtracks to a previous choice point.

ANALYSIS OF CONJUNCTIONS

Before starting the description of the mechanisms adopted to analyze conjunctions, it is worth noting that the analysis of conjunctions was already mentioned in a previous paper (Lesmo & Torasso, 1984). The present paper represents an advance with respect to the referenced one in that some new solutions have been adopted, which greatly enhance the homogeneity of the parsing process (not to mention the fact that the behavior of the parser was treated very sketchily in the previous paper). The presentation of the solution we adopted is based on the classification of sentences containing conjunctions reported in (Huang, 1984): we will start from the simpler cases and introduce the more complex examples later. A last remark concerns the language: as stated above, the FIDO system works on Italian; in order to enhance the readability of the paper, we present English examples. Actually, we are doing some experiments using a restricted English grammar, but it must be clear that the facilities that will be described are fully implemented only for the Italian grammar (the cases where Italian behaves differently from English will be pointed out during the presentation).

As for all other syntactic categories, the category "conjunction" also has an associated set of rules: the set contains a single, very simple rule: it saves the conjunction in a global register, which is available during the subsequent stages of processing. The simplest case of conjunction is the one referred to in (Huang, 1984) as "unit interpretation":

(1) Bob met Sue and Mary in London

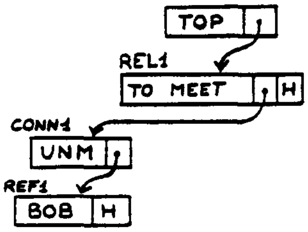
Normally, the rules associated with nouns hypothesize the attachment of a newly created REF node to a connector that (if it does not already exist) is, in turn, created and attached to the nearest node of type REL above the current node (or to the current node itself if it is of type REL). After the analysis of "Bob met", the situation of the parse tree would be as in fig.2.a (and REL1 is the current node). The analysis of "Sue" would produce the tree of fig.2.b. The noun rules have been changed to allow for the attachment of more than one noun to the same connector (should a conjunction be present in the register). In fig.2.c, the tree built after the analysis of sentence (1) is reported.

It must be noted that the most common example of natural change (the one called MOVEUP) is also useful when a conjunction is present. Consider, for instance, the sentence:

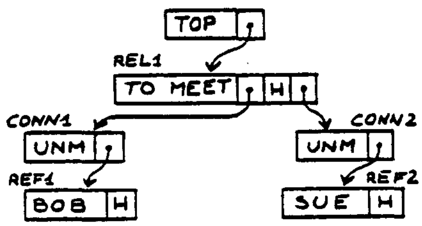
(2) John saw the boy you told the story and the girl you met yesterday

After the analysis of the fragment ending with "story", we get the tree of fig.3.a (and REF4 is the current node). According to the previous discussion, the noun "girl" would be stored in a REF node attached to CONN4. On the other hand, the semantics would reject this hypothesis, since the case frame (TO TELL: SUBJ/PERSON; DIROBJ/PERSON; INDOBJ/PERSON) is not acceptable. The portion of the tree representing "and the girl" would be "moved up" and attached to CONN2, thus yielding the tree of fig.3.b (that would be expanded subsequently, by attaching the relative clause "you met yesterday" to REF5).

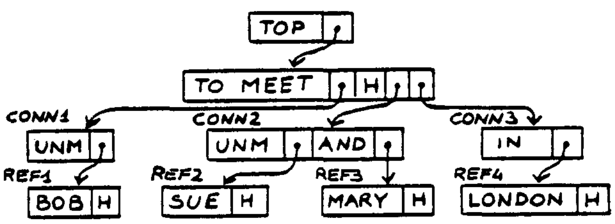
Unlike what happens in the previous cases, a new rule had to be added to account for the other types of conjunctions. This rule is a new natural change, that the system executes when the conjunction implies the existence of a new clause in the sentence. The need for such a rule is clear if we



(a)

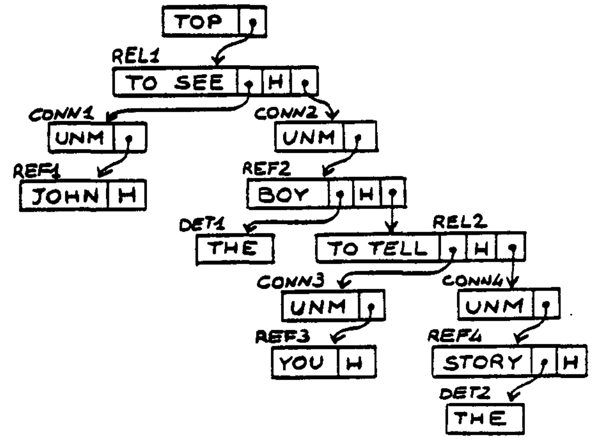


(b)

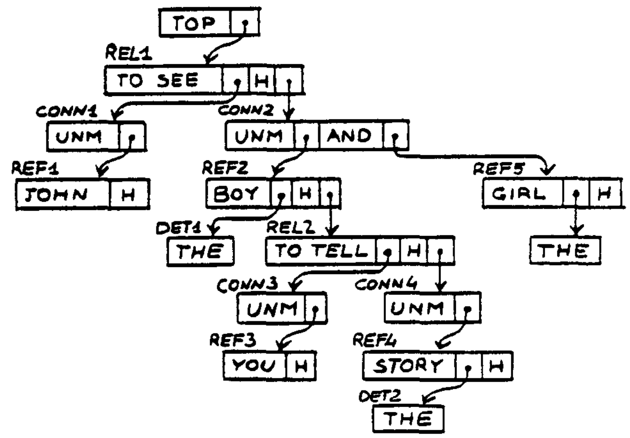


(c)

Fig.2 - Different phases of the interpretation of the sentence "Bob met Sue and Mary in London".
H means "head" and indicates the position of the node filler within the sequence of dependent structures.
UNM means "Unmarked" and indicates that the corresponding verb case is not marked by a preposition



(a)



(b)

Fig.3 - Two phases in the analysis of the sentence "John saw the boy you told the story and the girl you met yesterday" (the subtree relative to "you met yesterday" is not shown).

consider one of the basic assumptions of the parser. In a sense, the parser knows that it has to parse a sentence because, before starting the analysis, the tree is initialized by the creation of an empty REL node. Analogously, when a relative pronoun is found, the relative clause is "initialized" via the creation of a new empty REL node and its attachment to the REF node which the relative clause is supposed to refer to. The only exception to this rule is represented by gerunds and participles, which are handled by means of explicit preconditions in the VERB rule set. Of course, this can give rise to ambiguities when the past indicative and the past participle have the same form, as in the well known garden path:

(3) The horse raced past the barn fell

In the case of sentence (3), the choice of the indicative tense would be made, and the past participle rule would be saved to allow for a possible backtracking in a subsequent phase, as would actually occur in example (3) (we must note here that such an ambiguity does not occur in Italian). A further comment concerns the relative clauses with the deleted relative pronouns (as in (2) above): this phenomenon does not occur in Italian either; we believe that it could be handled by means of a natural change very similar to the one described below.

We can now turn back to the problem of conjunctions. Let's consider first a sentence where the right conjunct is a complete phrase.

(4) Bob met Sue and Mary kissed her

After the analysis of the sentence as far as "Mary", the structure of the tree would be as in fig.2.c (apart from the subtree referring to "in London"). When "kissed" is found, no empty REL node exists to accomodate it, thus the natural changes are triggered and, because of the preconditions, the new one (called INSERTREL) is executed. It operates according to the following steps:

1) A conjunction is looked for in the right subtree
2) It is detached together with the structure following it
3) The conjunction is inserted in the node above the first REL that is found going up in the hierarchy (in fig.2.c, starting from CONN2 and going upwards, we find REL1 and the node above it is TOP)
4) A new empty REL is created and attached to the node found in step 3
5) The structure detached in step 2 is attached to the new REL, inserting, when needed, a connector.

The execution of INSERTREL in the case of example (4) produces the structure depicted in fig.4, that is completed subsequently, by inserting "TO KISS" in REL2 and by creating the branch for "her" in the usual way.

Two more complex examples show that the ability of the parser to analyze conjunctions is not limited to main clauses:

(5) Henry heard the story that John told Mary and Bob told Ann

With regard to sentence (5), we can see the result of the analysis of the portion ending with "Bob" in fig.5.a. It is apparent that the execution of the steps described above causes the insertion of a new REL node at the same level of REL2 and attached to REF2; this seems intuitively acceptable and provides FIDO with a structure consistent with the compositive semantics adopted to obtain the formal query (Lesmo, Siklossy & Torasso, 1983).

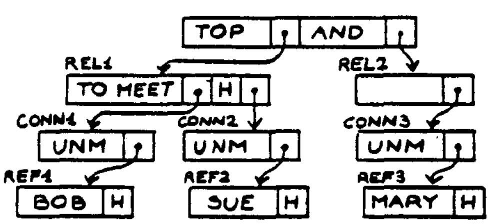


Fig.4 - Partial structure built during the analysis of the sentence "Bob met Sue and Mary kissed her".

An even more interesting example is provided by the following sentence:

(6) Henry heard the story John told Mary and Bob told Ann his opinion

where the INSERTREL and MOVEUP cooperate in building the right tree. What happens is as follows: after the execution of INSERTREL (in the way described above) "his opinion" is attached to REL3. The selection restrictions are not respected because four unmarked cases are present for the verb "to tell" (including the elliptical relative pronoun extracted from the first conjunct), so the smallest right subtree ("his opinion") is moved up and attached to REL1; again, the hypothesis is rejected (three unmarked cases for "to hear"). The tree returns to the original status and MOVEUP is tried again on a larger subtree (the one headed by REL3). Since a conjunction is found in the node above REL3, it is moved too and the analysis finally succeeds.

The last type of sentences that we will consider involves gapping. An example of clause-internal ellipsis is:

(7) I played football and John tennis.

When the name "John" is encountered, a unit interpretation is attempted ("football and John ") and it is rejected for obvious reasons. The only alternative left to the parser is the execution of INSERTREL, which, working in the usual way, allows the parser to build up the right interpretation.

Note that an empty node is left after the analysis of the sentence is completed, which is not done in the examples described above. This is handled by non-syntactic routines that build up the semantic interpretation of the sentence (formal query construction in FIDO). However the actual verb is made available as soon as possible, because the interpretation routines do not wait until the analysis of the command is finished before beginning their work.

As the reader will see from the following examples, no trouble is caused for the parser by the other kinds of gapping:

- left-peripheral ellipsis with two NP-remnants. For example:

  (8) Max gave a nickel to Sally and a dime to Harvey

  (unit interpretation "to Sally and a dime" attempted and rejected; INSERTREL executed; the semantic routines also have to recover the elliptical subject).
- left-peripheral ellipsis with one NP remnant and some non-NP remnant(s). For example:

  (9) Bob met Sue in Paris and Mary in London

  (exactly the same case as (8); the parser makes no distiction between NPs and non-NPs)
- Right peripheral ellipsis concomitant with clause internal ellipsis. For example:

(10) Jack asked Elsie to dance and Wilfred Phoebe

(same processing as before; more complex semantic recovery of lacking constituents is necessary).

Not very different is the case where "the right conjunct is a verb phrase to be treated as a clause with the subject deleted". As an example consider the following sentence:

(11) The man kicked the child and threw the ball.

In this case, the search for an empty REL node fails in the usual way and INSERTREL is executed as discussed above, except that the conjunction is still in the register and no structure follows it, so that the steps 1,2, and 5 are skipped.

Finally, the "Right Node Raising", exemplified by:

(12) The man kicked and threw the ball.

The problem here is that the left conjunct is not a complete sentence. However, the syntactic rules have no troubles in analyzing it; it is a task of semantics to decide whether "the man kicked" can be accepted or not. In other words, "the ball" could be considered as an elliptical object in the first clause; although the procedures for ellipsis resolution are unable, at the present stage of development, to handle such a case, it is not difficult to imagine how they could be extended.

To close this section, two cases must be mentioned that the parser is unable to analyse correctly. In sentence (13)

(13) John drove his car through and completely demolished a plate glass window

a preposition (through) has no NP attached to it. The problem here is very similar to that of "dangling prepositions" (and, like the latter, it does not occur in Italian). A simple change in the syntax would allow a CONN node to be left without any dependent REF. Less simple would be the changes necessary in the anaphora procedures to allow them to reconstruct the meaning of the sentence (the difficulty here is similar to the "Right Node Rais-

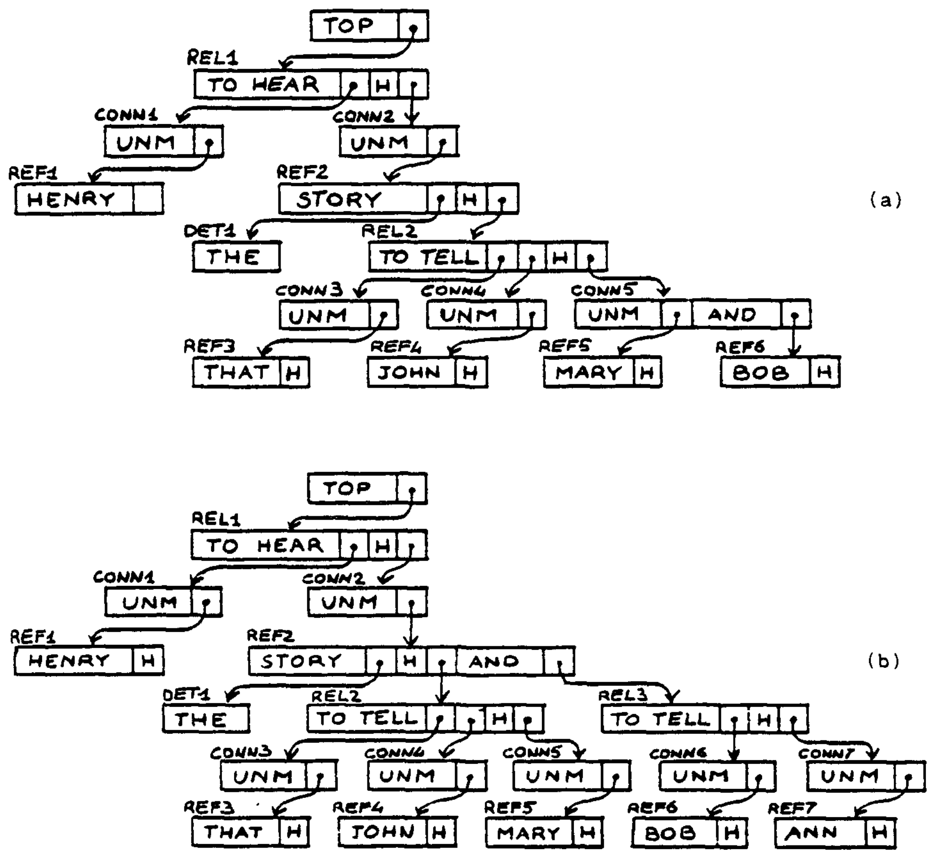


Fig. 5 - Two phases in the analysis of the sentence: "Henry herd the story that John told Mary and Bob told Ann".

ing" discussed above).

The last problematic case is concerned with multi-level gappings, as in the following example:

(14) Max wants to try to begin to write a novel and Alex a play.

In this case, the insertion of an empty REL node to account for the second conjunct ("Alex a play") does not allow the parser to build a structure that corresponds to the one erased by the ellipsis. We have not gone deeply into this problem, which, unlike the preceding ones, also occurs in Italian. However, it seems that, also in this case, the increased power of the procedures handling elliptical fragments could provide some reasonable solutions without requiring substantial changes to the presented approach to parsing.

CONCLUSIONS

AS stated in the introduction, a proper treatment of coordination involves the ability to interrupt the analysis of the first conjunct when the conjunction is found and the ability to analyze the second conjunct taking into account what happened before.

The system described in the paper deals with the two problems by adopting a robust and modular bottom-up approach. The first conjunct is extended as far as possible using the incoming words and the structure building syntactic rules. Its completeness and/or acceptability is verified by means of another set of rules that fit easily in the proposed framework and do not affect the validity of the other rules.

The second conjunct is analyzed using the same standard set of structure building rules, plus an exception-handling rule that accounts for the presence of a whole clause as second conjunct. The need to take into account what happened before is satisfied by the availability of the portion of the tree that has already been built and that can be inspected by all the rules existing in the system.

The paper shows that the approach that has been adopted enables the system to analyze correctly most sentences involving conjunctions. Although some cases are pointed out, where the present implementation fails to analyze a correct sentence, we believe that the solutions presented in the paper enlight some of the advantages that a rule-based approach to parsing has with respect to the classical grammar-based ones.

REFERENCES

V.Dahl, M.McCord (1983): Treating Coordination in Logic Grammars. AJCL 9, 69-91.

X.Huang (1984): Dealing with Conjunctions in a Machine Translation Environment. Proc. COLING 84, Stanford, 243-246.

L.Lesmo, L.Siklossy, P.Torasso (1983): A Two Level Net for Integrating Selectional Restrictions and Semantic Knowledge. Proc. IEEE Int. Conf. on Systems, Man and Cybernetics, India, 14-18.

L.Lesmo, L.Siklossy, P.Torasso (1985): Semantic and Pragmatic Processing in FIDO: a Flexible Interface for Database Operations. Information Systems 10, n.2.

L.Lesmo, P.Torasso (1983): A Flexible Natural Language Parser Based on a Two-Level Representation of Syntax. Proc. 1st Conf. ACL Europe, Pisa, 114-121.

L.Lesmo, P.Torasso (1984): Interpreting Syntactically Ill-Formed Sentences. Proc. COLING 84, Stanford, 534-539.

L.Lesmo, P.Torasso (1985): Weighted Interaction of Syntax and Semantics in Natural Language Analysis. 9th IJCAI, Los Angeles.

F.Pereira (1981): Extraposition Grammars. AJCL 7, 243-256.

F.Pereira, D.Warren (1980): Definite Clause Grammars for Language Analysis: A Survey of the Formalism and a Comparison with Transition Networks. Artificial Intelligence 13, 231-278.

J.J.Robinson (1982): DIAGRAM: A Grammar for Dialogues. Comm. ACM 25, 27-47.

R.M.Weischedel, J.E.Black (1980): Responding Intelligently to Unparsable Inputs. AJCL 6, 97-109.

W.A.Woods (1973): An Experimental Parsing System for Transition Network Grammars. In R.Rustin (ed.): Natural Language Processing, Algorithmics Press, New York, 111-154.