Marjan Grootveld and Anneke Neijt

Underspecification in coordination: X-Y-Z-parsing

1. Introduction

Widely accepted methods for dealing with ambiguity are parallelism, backtracking and determinism. Since the former two are costly, we explore the latter approach. Ordinary rewriting rules embody both knowledge about dominance relations between nodes and knowledge about precedence relations. It will be argued, that dividing these information sources enhances the desired determinism. Moreover, this division perfectly suits the three-dimensional representation of coordination, a phenomenon that is a major source of ambiguity. In section 3, the advocated parser is presented. Section 4 connects the 3D-approach to the binding theory and to formal semantics and elaborates on the usefulness and the costs of a third dimension. Section 5 recapitulates the argumentation.

2. Separating dominance and precedence

Many a grammar can expand a given node in more than one way and the parser associated with such a grammar needs to be able to choose the correct expansion. In other words, the parser must know how to deal with ambiguity. Widely accepted are the parallel approach, the serial approach (i.e. backtracking) and the younger approach of minimal commitment, connected to determinism.

It is well-known that parallelism and backtracking are costly. Their (simultaneous and subsequent, respectively) efforts to find a correct analysis form an instantiation of what we call 'overspecification'. In an overspecific strategy conclusions are drawn too fast, too many possibilities are allowed, too much is taken for granted. Too much, that is, to become part of the final output. In this respect we side with the deterministic approach, which demands that all structure created during the parse is made use of in the result; all structure is indelibly constructed (cf. [Marcus, Hindle, Fleck 1983]), no partial analysis may be thrown away.

This strict position calls for a principled approach to ambiguity, which is for instance introduced by coordination. Sometimes a string is just temporarily ambiguous, like 'the saucers' in example (1), which may be the prefix of example (2).

(1) John gave Jill the cups and the saucers
(2) John gave Jill the cups and the saucers were on the table

However, permanent, insoluble ambiguity occurs as well, as in (3):

(3) beautiful men and women

String (3) can be a coordination of beautiful men and non-descript women, but 'beautiful' can equally well predicate over the conjoined men and women. In this paper we will restrict our attention to syntactic ambiguity, ignoring polysemy etc.

To avoid the inefficiency of parallel and backtracking strategies, we have chosen a deterministic approach to ambiguity and therefore to coordination. The reader should notice that this does not entail the use of a Marcus-parser, since it has been shown in [Grootveld 1990] that the Marcus-combination of the limited look-ahead facility and the almost inaccessible stack is fundamentally unfit for coordination.

In our parser, the X-Y-Z-parser, determinism as defined above is achieved by different means, namely the safe strategy of underspecification. The parser is prevented from overenthusiasm in that it must restrict itself to facts, that is, to findings that are certain and by definition unambiguous. In the pair (1) - (2) this approach implies that the parser, upon reading 'the saucers', does not commit itself to either an object or a subject interpretation, but continues to 'eat input' until 'the saucers' is disambiguated (if sentence (1) is the actual input, this will happen almost immediately). Example (3) will not be unambiguously interpreted at all, as 'permanent' already implies. It follows that the parse of (3) maintains the original multiple meaning, which we consider a desirable effect. As a consequence, an output representation is called for that allows ambiguity. It is evident that no ordinary tree diagram as features in the Government and Binding theory (GB) can express structural ambiguity, since one tree represents per se one analysis. Of course, this holds not only for coordination-induced ambiguity: to diagrammatically represent the famous 'Flying planes can be dangerous', GB also needs a parse forest.

In our view, the notions of 'parsing safely with underspecification' and 'un-treelike output' do not accidentally co-occur, but are narrowly connected (cf. [Marcus 1987]). The connection consists in the node relations of 'dominance' and 'precedence'. Information embodied in a tree is information about these relations: a dominance relation exists between a branching node and its daughters and one of the daughters precedes the other one. (In accordance with the Principle of Binary Branching, a node can have at most two daughters.) Usually, both types of grammatical knowledge collapse in a rewriting rule: the left-hand side dominates the right-hand side, whereas the elements in the right-hand side are ordered by precedence. Now it is easy to imagine a situation where the parser safely concludes some dominance fact, without there being a precedence fact, or vice versa. Were the parser in that situation to commit itself to one particular rewriting rule, it would add predictions to facts and in doing so it would be overspecific, with all the inefficiency effects. For this reason, the development of a parser with separate treatment of dominance and precedence is a very interesting goal. This separation has also been advocated in [Marcus 1987]; like Marcus, we assume a third dimension for the representation of coordinate structures. Marcus' analysis of coordination is very sketchy, however, and for a 3D-analysis of coordination to be satisfying, a third node relation is necessary, viz. the relation of behindance. (A three-dimensional analysis of e.g. coordination is presented in [Goodall 1987], but it lacks formalization.) Each of the three relations is starring in one of the X-Y-Z- modules, which are to be discussed now.

3. The parser

Common to all three modules is a database of descriptions (cf. [Marcus, Hindle, Fleck 1983]). In the spirit of our safe strategy, the parser does not create nodes, partial structures, etc., but descriptions of nodes and of structures. The reason for this is the greater flexibility of a description set: in a structure like (4), no node B can be inserted between A and C without destroying the structure. As the reader will remember, though, all structure is permanent and indelible. On the other hand, a description such as (5), which has structure (4) as one of its representations, can easily be extended monotonously to the effect that B intervenes (cf. the expansion from (5) to (6), where 'dom' means 'dominates indirectly').

```
(4) A          (5) dom(A,C).          (6) dom(A,C).
    |                                     dom(A,B).
    C                                     dom(B,C).
```

This attractive flexibility explains why we have the parser build and exploit a database of descriptions. In the following, both excerpts from the database and structures will be given as examples. It must be stressed that the latter, presented instead of the database itself for expository reasons, are merely possible representations of the database. In fact, given the database, constituent structures seem superfluous. Even notions such as c-command and subjacency could be redefined such that they apply to the database, which suggests that the status and necessity of constituent structure is a point of further study.

3.1 X-parse

In the GB-framework the lexicon is very important, because it stores much knowledge about possible structures. A category X is, according to the X-bar-model, the core of a constituent XP, what translates in our model into the creation of a XP whenever a category X is found in the input list. The first module, X-parse, mirrors the major role of the lexicon. The input words are looked up and the first descriptions come into being, based on the categories present. GB's X-bar-module, and therefore our X-module, projects higher levels above the categories, such as an Adjective Phrase above an adjective (cf. (7a, conceivably represented as (7b))). A determiner gives rise to the creation of a Determiner Phrase. Moreover, a determiner selects a Noun Phrase as its sister node and so an NP is created, cf. (7c, 7d). Some categories trigger the creation of a longer dominance chain, e.g. inflected verbs (inflectional affixes and stems are presented to the parser as derived words, not as separate items). (7f) represents the statements in (7e) associated with such a 'I+V'-cluster. It says both that I requires a Verb Phrase as its complement and that I precedes VP (in English).

```
(7a) dom(a1,huge).   (7b) AP1      (7c) dom(det1,the).     (7d)  DP1
     dom(ap1,a1).          |            dom(dp1,det1).            /  \
                          A1           dom(dp1,np1).           DET1  NP1
                          huge         prec(det1,np1).         the

(7e) dom(vp1,v1).      (7f)   IP1
     dom(ip1,i1).             /  \
     dom(ip1,vp1).          I1   VP1
     prec(i1,vp1).                |
                                  V1
```

A note on indices: upon creation, every token of a category receives an index; assuming that the input string is only 'the', the determiner is indexed '1', the DP also, the NP also, etc. Input that causes more tokens of a category to originate, receives a fresh index for each of these tokens. That means, that the input 'The woman developed a parser' adds among many other things four DP-descriptions to the database: 'the' gives rise to DP1, 'woman' to DP2, 'a' to DP3 and 'parser' to DP4. Of course, in a later stage of parsing, this number of DP's will be reduced.

To resume: the X-module hosts the lexical look-up, it stores all dominance facts that X-bar-theory predicts for the given categories and sometimes it stores a precedence fact. The latter only happens in the case of complementation, because ordinary precedence facts, for instance relating to the ordering of the input words, are of no concern here. To put things differently, dominance is this module's central notion, even allowing for the ungrammatical ordering [man, the] to be pipelined to the next module: Y-parse. As Dirksen [1990] notices, gathering as much information as possible from the lexicon is worth doing, but not enough: a secondary proces should transform the various data to a more coherent structure.

3.2 Y-parse

The second module is intended to check (mainly) the precedence of the participating categories and to combine them, thereby making more and more statements about higher nodes. It is important to see that the set of participating categories consists of two subsets: on the one hand all actual, attested, categories (e.g. the determiner in (7c) and (7d)) and on the other hand the postulated, 'predicted with certainty' categories (e.g. the NP in (7c) and (7d)). The algorithm of the Y-parser belongs to the 'shift reduce' family, since it repeatedly tries to reduce two neighbour elements to a higher node. Of course, an important issue is when reduction obtains in our parser. Two answers exist, that is, the parser can follow two strategies. The main one, a top-down strategy, exploits the complementation information as stored by the X-parser. We have demonstrated how the X-parser postulates a category ZP if an actual input category Y selects ZP as its complement. Next, the Y-parser must verify the presence of the ZP. In concreto: (7c), repeated as (8a), and (8b) show the relevant facts that are added to the database when the X-parser encounters the words 'the' and 'men' respectively. The two NP's occurring in the database - one postulated as a sister of the determiner, the other attested as higher projection of 'men' - differ in their index.

```
(8a) dom(det1,the).            (8b) dom(n1,men).
     dom(dp1,det1).                 dom(np2,n1).
     dom(dp1,np1).                  dom(dp2,np2).
     prec(det1,np1).
```

(The reader may need to be reminded that each token is assigned a new index.) When [det1, n1] is input to the Y-parser, the top-down question is 'The first element under consideration is a determiner that selects an NP to its right (cf. (8a)). Is there an NP?'. The answer would be in the negative, were it not for section (8b) of the database, which succesfully links the actual noun in the input list to its ancestor node NP. Since what we know about the two DP's involved is unifiable, we can identify them as different descriptions for the same node (cf. [Marcus 1987]). Even if the noun is part of a larger structure, for instance a coordination ('the men and women'), taking it together with the determiner will not rule out later noun-combining operations, since we assume dominance to be an indirect relation. As argued with respect to examples (4) through (6), indirectness allows for inserting nodes between a pair of nodes. This implies for the case at hand that the top node of the coordinate structure can be wedged between the determiner and the noun, if need be.

Of course, a deterministic parser cannot deal with modifiers in a top-down fashion: how could it safely predict unpredictable material? A bottom-up strategy is therefore needed when the top-down selection checking strategy fails. The Y-parser tries to match the neighbour elements to the right-hand side of a rewriting rule. Now the astute reader will remember our objection to rewriting rules (viz. overspecification). However, our use is far less massive (no projection or complement selecting rule is burdening the Y-grammar, since that knowledge is incorporated in the X-module) and the rewriting grammar is only questioned as a last resort, so scruples about separating dominance and precedence are superfluous here.

Thanks to the huge amount of vertical information in the X-parse, the Y-parser can nearly restrict itself to horizontal relations. The combination of the X-parser and the Y-parser will analyse many different inputs satisfactorily. However, no coordinate structure will be among them, since coordination demands a special treatment for the following reason. The Principle of Binary Branching (cf. [Kayne 1984]) was mentioned before as one of our guidelines. Now, an enumeration like 'Tom, Dick and Harry' outnumbers the available two sister branches. The usually proposed alternative representation, sub-ordination, places each conjunct deeper in the tree than the preceding conjunct, thereby making incorrect predictions about binding relations (see section 4). It is clear that a third dimension is needed to account for coordinate structures and therefore the X-Y-Z-parser exploits the third node relation of behindance, rendering a post-ordinated representation. Consequently, it does so in the third module: Z-parse.

3.3 Z-parse

The ins and outs of the Z-parser will be illustrated with an overview of a complete parsing trace. The running example will be sentence (9), containing two conjunctions for two separate coordinations. The X-parse yields (10) - the input to Y-parse - and (11), which is the relevant part of the database. Precedence facts have been left out.

(9) John and Mary saw Bob and Kim

(10) [np1, co1, np2, i+v1, np3, co2, np4]

```
(11) dom(np1,John).        dom(i+v1,saw).        dom(co2,and).
     dom(co1,and).         dom(vp1,i+v1).        dom(cop2,co2).
     dom(cop1,co1).        dom(ip1,vp1).         dom(np4,Kim).
     dom(np2,Mary).        dom(np3,Bob).
```

The Y-parser has hardly any knowledge about coordination, so it has to pass NP1 unmodified onto the next module. Since the word immediately following a conjunction must be part of the right conjunct, Y-parse will state that COP1 dominates NP2. Furthermore, it fits NP2 and NP3 under the IP as its subject and object, respectively. As an immediate successor of CO2, NP4 becomes a daughter of COP2. The result of the Y-parser can be seen in (12) and (13).

(12) [np1, cop1, ip1, cop2]

```
(13)                    IP1

                COP1                VP1                COP2

     NP1     CO1     NP2      I+V1      NP3     CO2      NP4
```

The Z-parser is presented with (12), an input consisting of the top nodes of four fragments that ideally root in one top node. Going from left to right, the Z-parser cannot but conclude that NP1 is the left conjunct of the first conjunction, that is, NP1 is NP2's counterpart. In a three-dimensional analysis, this calls for the introduction of planes in which the conjuncts are to be placed. Not only precedes NP1 NP2 in the left to right order - 'John' precedes 'Mary' in sentence (9) -, but NP1 is also in front of NP2 when we look at the front-to-back order. In other words, NP2 is positioned in the plane behind NP1, or, stated as a database fact, 'post(np2,np1)'. A third formulation says that, if we impose the same x/y-coordinate system upon every plane, NP1 and NP2 only differ in their z-coordinate, where the z-coordinate of NP2 is greater than the z-coordinate of NP1.

By ranging NP1 within COP1, NP1 becomes part of IP1, which leaves IP1 and COP2 to be combined. NP4 (within COP2) searches and finds its counterpart in the set of CO2's left neighbours (i.e. in the set {NP3, VP1, IP1}), namely NP3 (cf. the notion of "rightmost nodes" in [Snarr 1984]). Again the two conjuncts share x- and y-coordinates and differ in their z-coordinate. This concludes the rough sketch of the Z-parser, yielding indeed the root node: IP1.

4. Extra motivation for the third dimension

By splitting up rewriting rules in a dominance part and a precedence part, one avoids the drawbacks of an overspecific parser, for if syntactic ambiguity arises, no superfluous prediction will be made. This is one of our starting-points, the other being that a succesful account of coordination calls for a third dimension. Taken together, these assumptions yield three node relations of equal status: dominance, precedence and behindance. It will be evident that the latter without the former gives a less natural result: the traditional notion of rewriting rules at the one hand (with dominance and precedence narrowly interwoven) and behindance at the other would make an odd couple. This section exemplifies the possibilities of the 3D-approach and serves to emphasize the economy of that approach: although the extra dimension enlarges the expressive power - because that is what coordinated structures need - this power is severely restricted. The first limitation and the one most easily stated: the initiation of a plane can only be triggered by a coordinating conjunction, i.e., non-coordinate input will be assigned the usual two-dimensional structure. (Disambiguating a comma as coordinating, as opposed to subordinating, may be hard, but that parsing problem does not affect the validity of the rule.) We will not attempt to formalize the main limitation here, but the three cases below (of binding, formal semantics and scope ambiguity) illustrate the fact that the shortest and most efficient routes through the graphic representation correspond to the best (or the unique) reading. Weird, improbable, expensive, long-winding graph routes seem to have no counterpart in language. This finding neatly fits our economical intentions and can be taken to support the 3D-approach.

One last remark about the third dimension before we arrive at the examples: we will draw extremely simple representations in this section. We leave out those parts of the structure that bear no relevance to the postordinated elements and since we restrict the examples to postordinated noun phrases, no occurrence of for instance the verb is represented. Not only serves this practice to highlight what is really important, but our strategy of underspecification also implies this restrictedness: "Do not assume a plane without necessity". In the first example below, binding is at stake and we will argue that it obtains in more than one plane. Because the verb occurs only once in the input (i.e. is not postordinated), at first sight the structure needed for c-command must be absent in at least one plane. However, the underspecification dictum can be rephrased for our purposes as "Every element in a plane has so to speak access to every element that is not explicitly in another plane". It follows, that the position of the verb is unproblematic, whether it is in the first plane, in the second, in both (through some copying mechanism) or superposed. Therefore, the structure necessary for c-command is available in every plane in the representation of sentences like (14) and (16a). As a consequence, the binding theory is obeyed in the way we shall outline now. The exact location of "shared material" will be discussed elsewhere.

In section 3.2 we hinted at the failure of a sub-ordinating analysis to assess the correct binding relations and the crucial example is (14). According to the binding conditions in the Government and Binding theory, anaphora must be bound by their antecedents. Munn [1989] argues for subordination through adjunction: [NP [NP John] [BP [B and] [NP Mary]]], where 'B' stands for 'Boolean'. However, if 'Mary' is located deeper in the tree than 'John', it will be too deeply embedded to bind 'herself' and the sentence is wrongly predicted to be ungrammatical.

(14) John and Mary saw himself and herself

This sentence yields no problems for the 3D-approach. Exactly along the lines of section 3.3 (consider (14)'s resemblance to sentence (9)), the X-Y-Z-parser has reduced the input to an IP-node. The database contains among other things the information

(15) post(np2,np1).
post(np4,np3).

(where [np1, np2, np3, np4] are the NP's in (9) and (14)) and if a graphic representation of the database is produced, NP1 and NP3 will occur in the first plane and NP2 and NP4 in the second. That means, that each of the anaphora is bound within its plane, what we take to be the ideal outcome for the following reason. As no version of the binding theory has been formulated in terms of planes, the default assumption is that the conditions respect planes, i.e. apply per plane. The opposite assumption, that binding conditions hold through planes, is less restrictive and therefore renders a weaker hypothesis.

That the 3D-approach is capable of dealing with binding can hardly be called "extra motivation", as this section's heading promises; any theory of coordination should comply with the binding theory. In the beginning of the paper we argued for a proper treatment of ambiguity and expressed our preference for a parser that conserves the original multiple meanings in the output of the analysis. The time has come to show that our 3D-approach to coordination has this pleasant property. As we explained, the final database is possibly ambiguous, thanks to underspecification. What is more, in some cases the database even allows for an ambiguous graphic representation. We will now present two of these cases.

First, consider the following sentences (sentence (9) is repeated as (16a)):

(16a) John and Mary saw Bob and Kim
(16b) John saw Bob and Mary saw Kim
(16c) They saw them

Sentence (16b) is a typical example of a distributive sentence, where one cannot safely conclude that, at a given moment, more than one person saw more than one other person. This contrasts with (16c), where the collective reading is more prominent. Sentence (16a) permits both interpretations and the 3D-representation contains them. Since the structural relations in sentence (16a) are equal to those in (14), the graph-properties are the same: the first NP (dominating 'John') is in front of the second ('Mary') and the third ('Bob') is in front of the fourth ('Kim'), as in (17). Now, the distributive reading corresponds to an interpretation per plane, whereas the collective reading is obtained by linking the first COP to the second COP. (Under the common assumption that the checking of binding relations precedes semantic interpretation, it is easy to see why the collective interpretation is not available for sentence (14).) The logically possible third reading that John saw Kim without having seen Bob (and that Mary saw Bob without having seen Kim) can in principle be accomplished by crossing links:

(17) second plane	Mary	Kim
first plane	John	Bob

However, despite logic, language does not permit this interpretation (although judgement varies, cf. Van Zonneveld [1991]). What we observed above with respect to binding theory is therefore confirmed by semantics: planes are obvious entities.

Example (3) (repeated below as (18)) serves as a second example of ambiguity which can easily be conserved in the output.

(18) beautiful men and women

(19)

second plane		women
first plane	beautiful	men

The AP has scope over the COP as well as over the separate nouns. The former scope relation yields the reading where everyone is beautiful and the latter predicates only over the men. It is clear again, that items in different planes cannot be connected, except through the COP and if so, the connection "percolates" from the COP to each of the conjuncts: example (18) cannot entail that only the women are beautiful, thereby skipping the noun in the first plane (cf. schema (19)). Once again, planes are respected.

5. Conclusion

We started out with two goals, namely to parse natural language in an efficient manner and to show the efficiency with respect to coordinate structures. Both of these goals are traditionally threatened by ambiguity and therefore a third goal arose: to deal with ambiguous input on a thorough and not ad hoc basis. We have reached the first and the third goal through underspecification, an economical strategy that is accomplished by dividing information about dominance relations and information about precedence relations. We have reached the second goal by the notion of behindance.

We have argued that the usual two-dimensional representation of natural language does not suit the analysis of coordination and hence we implemented a third dimension. The associated information, i.e. the knowledge of which nodes are related through behindance, is on a par with the other information types. This is also clear from the structure of the X-Y-Z-parser, where each module corresponds to a single node relation.

The parser and the underlying theory live by Occam's razor, in the sense that nothing is created unless its existence is unambiguously motivated. This statement is true both for (two-dimensional) nodes and for planes and as a consequence, the three-dimensional representations are very restricted; the introduction of a third dimension does not lead to a huge increase of logically possible but empirically unmotivated structures. Aside from this reassuring aspect, the 3D-account of coordination globally ambiguous structures throughout the parse, which results in an ambiguous output. We take this ambiguous output as a bonus and as an indication that we are on the right track, since the use of language seems to base its robustness on this characteristic.

References

Dirksen, A.: Monitoring processes in language production. Diss. Utrecht, 1990.

Goodall, G.: Parallel structure in syntax (...). Cambridge [etc.], 1987.

Grootveld, M.J.: 'Marcus-Parsing and Coordination: A Mismatch', in: R. Bok-Bennema & P. Coopmans (eds.): Linguistics in the Netherlands 1990. Dordrecht, 1990.

Kayne, R.S.: Connectedness and binary branching. Dordrecht, 1984.

Marcus, M.P.: 'Deterministic parsing and description theory', in: P. Whitelock [et al] (eds.): Linguistic theory and computer applications. London [etc.], 1987.

Marcus, M.P., D. Hindle, M.M. Fleck: 'Talking about talking about trees', in: Proceedings of the 21st annual meeting of the ACL, 1983

Munn, A.: 'Coordinate structures, X-bar theory and parasitic gaps', presented at the Views on Phrase Structure Conference. University of Florida, 1989.

Snarr, V.L.: Theory and parsing of the coordinate conjunction "And". Toronto, 1984. (Technical report CSRI-171)

Zonneveld, R. van: 'The Syntax of Gapping', in: F. Drijkoningen & A. van Kemenade (eds.): Linguistics in the Netherlands 1991. Amsterdam/Philadelpia, 1991.