

# **REVISIONS DURING GENERATION USING NON-DESTRUCTIVE UNIFICATION**

Koenraad De Smedt  
NICI, University of Nijmegen  
Postbus 9104, 6500 HE Nijmegen  
The Netherlands  
desmedt@psych.kun.nl

## **ABSTRACT**

A realistic model for natural language generation must account for overt revisions of the syntactic structure (self-corrections) as well as covert revisions (backtracking on syntactic options). This paper presents the preliminaries of a hybrid architecture for grammatical encoding (the 'tactical' phase in sentence generation) which allows such revisions. This architecture combines the concept of activation with a non-destructive variant of the unification algorithm.

## REVISIONS DURING SENTENCE GENERATION

While much recent work in natural language processing is devoted to the formalization of grammars, little attention is generally paid to how these grammars must function in realistic tasks. Realistic models of natural language processing should not only incorporate linguistic competence, but should also operate under the same constraints. A human speaker produces utterances under constraints of time pressure, incomplete knowledge, a changing world, and limited memory. Time pressure requires the construction of an utterance in an incremental fashion (Kempen & Hoenkamp, 1987). Incomplete knowledge during incremental generation forces the generator to make early choices which may later turn out to be non-optimal (De Smedt & Kempen, 1987). A changing world may invalidate the message which is being uttered (ib.). Finally, a limited memory puts an upper bound on the complexity and active time span of syntactic structures (Kolk, 1987).

All of these conditions contribute to the rather frequent occurrence of revisions to the syntactic structure during generation. At least five mechanisms of change may be distinguished. First, it must not be assumed that the presumably language-independent Conceptualizer (the ‘strategic’ component in generation) plans the message so that the Grammatical Encoder (the ‘tactical’ component) can build a syntactic structure in a strictly top-down and left-to-right fashion. Hence, an incremental mode of generation must allow upward and downward expansion of a partial syntactic structure as well as insertion into a structure (De Smedt & Kempen, 1987). An example of insertion is given in (1a).

Second, the Grammatical Encoder does not know when a message from the Conceptualizer is complete, although this knowledge is necessary for some lexical and structural choices. For example, a missing *agent* may eventually invalidate the choice of an active lexical entry and necessitate the substitution of a passive lexical entry for the active one, as in (1b).

- (1) a. John and Mary were at home... —> John and Mary seemed to be at home.  
b. Show... —> It is shown...

Third, even if the successful construction of syntactic structures is in principle independent of the ordering of conceptual inputs in real time, syntactic constraints flow in a specific direction. In particular, if a partial syntactic structure is expanded from the bottom upward, it may be too late for subcategorization restrictions at higher levels to constrain lexical selection at lower levels (De Smedt 1990:177). For example, the non-finite subclause in (2a) is incompatible with a verb which is subcategorized to take a finite subclause (2b).

- (2) a. \*I know... I ...him to come tomorrow.  
b. I know ... that he comes tomorrow.

Fourth, the limits of human memory may cause elements of the intended utterance to be forgotten, confused or substituted by unintended elements during speech, leading to hesitations, syntactic incoherence, speech errors and restarts, e.g. (3a).

Finally, the Conceptualizer may revise the message which is being formulated by the Grammatical Encoder, or which is perhaps already partially uttered. This may result in an overt self-correction, as in (3b).

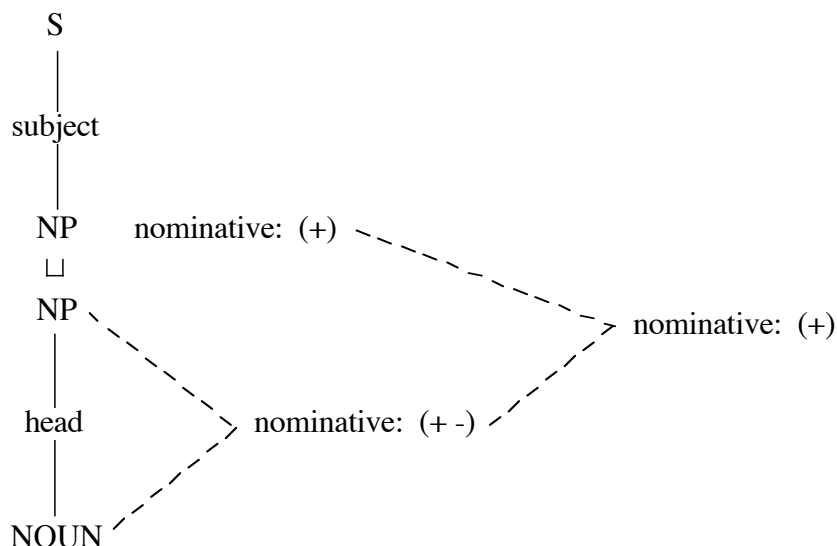
- (3) a. The next speaker will be given by Jonathan Slocum. (heard at an ACL conference)
- b. The man with the moustache ...uh... with the spectacles pushes the clown.  
(adapted from experimental data collected by Van Wijk (Van Wijk & Kempen, 1987)).

### UNIFICATION IN SEGMENT GRAMMAR

Unification-based grammars allow some of the flexibility needed for a generation task. For instance, such grammars are independent of the order in which various functional descriptions are unified (Kay, 1985), thereby allowing both top-down and bottom-up construction of the syntactic structure as well as the construction of several partial structures in parallel.

The IPF system (De Smedt 1990; forthcoming) is a model of grammatical encoding which employs a unification-based grammar formalism called Segment Grammar (SG). SG is especially suited to incremental generation (Kempen, 1987) since it defines the incremental addition of single branches called syntactic *segments* to partial syntactic trees. Each syntactic segment consists of two nodes, a *root* and a *foot* node, which are linked by a grammatical function label. For example, an S-subject-NP segment represents a subject relation between a sentence and a noun phrase. The nodes are functional descriptions (FDs or feature structures) which can be *unified* with other nodes, thus causing segments to join and form a larger structure (see Figure 1). As usual, the successful operation of unification is dependent on the agreement of syntactic features and subcategorization restrictions.

IPF models incremental sentence generation by allowing the piecemeal construction of a syntactic structure. In addition, IPF is a model of distributed grammatical encoding, allowing different parts of the structure to be formulated in parallel. Clearly, such a model must allow structural revisions if it is to be realistic, as indicated above.



**Figure 1**

Unification of nodes changes the disjunctive values of the shared feature ‘nominative’.

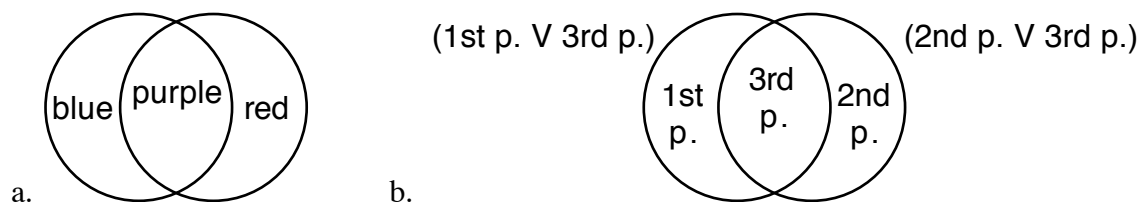
### NON-DESTRUCTIVE UNIFICATION

As originally formulated in theorem-proving, unification is non-destructive: it yields a substitution which can be used to update values. In practice, however, unification is often viewed as a destructive operation which causes two distinct FDs to become literally identical (Karttunen & Zwicky, 1985). As such, it is not suited to a model such as IPF where structures may need revision. Restoring the original structures joined by destructive unification is a possible solution, but may be a computationally expensive operation. If features with disjunctive values are involved, the feature values in the resulting FD are reduced to the intersections of the original values (see Figure 1). Clearly, a destructive unification operation must keep past states in memory if it wants to restore the original values when a unification is undone. This is complicated by the fact that FDs may be *reentrant* in the sense that several nodes share the same features (cf. also Shieber, 1986), which causes chaining of revisions in the syntactic structure.

Another solution, adopted in the implementation of Kempen and Vosse’s (1989) model, consists of using a grammar without disjunctive values. However, this will of course cause a massive proliferation of segments, which not only increases the size of the grammar but also increases the number of attempted unifications during generation.

I propose a better solution by devising a non-destructive or ‘virtual’ unification mechanism which does not destroy the initial configuration of nodes, nor their disjunctive feature values. Rather than computing a new FD which represents the unification of two other structures, virtual unification accesses the original structures *as if* they are unified. The metaphor of the subtractive effect of coloured light filters is useful here. If two structures A and B are virtually unified, then for the duration of their unification the features of A may

only be looked at with those of B as a filter on them. This is schematically represented in Figure 2.



**Figure 2**

Subtractive effect of filters: (a) colours; (b) syntactic features: the intersection of first or second person and second or third person yields second person.

The filter metaphor illustrates the non-destructive property of virtual unification: as soon as the filter is taken away, the original colours, become visible. Similarly, as soon as the non-destructive unification is undone, the original features are immediately accessible.

The technique of non-destructive unification consists of manipulating the *access* to features of FDs (rather than changing the FDs themselves). This is implemented by assigning a set of *companions* (initially empty) to each FD. When an FD unifies with another one, they mutually become a member of each other's set of companions. Consequently, each access of a feature is filtered through the corresponding feature of each companion of the FD. Undoing a unification corresponds simply to the removal of a companion. Non-destructive unification has been implemented in CommonORBIT (De Smedt, 1987) and has been incorporated into SG.

Since the access of a feature has thus become more complicated in favor of an easy updating, an efficient implementation is critically dependent on fast access. The cost of 'filtered' access is not so high in a system with 'flat' (non-recursive) feature matrices where each value is a set of atomic features, as is the case in SG. When such structures are represented as bit vectors, they can be unified in a very fast way using the 'logical AND' machine instruction (Proudian & Pollard, 1985).

### THE 'UNIFICATION SPACE'

As an further exploration of what easy revisions to the syntactic structure might allow, let us consider a new control structure which would allow a maximal amount of changes to the syntactic structure over time. The proposed architecture should allow the syntactic structure to be reconfigured due to incoming information or contextual influences and should take into account constraints of time and memory.

A novel approach to sum various factors in one network is the use of *activation* (e.g. Stemberger, 1985; Waltz & Pollack, 1985; Dell, 1986). But whereas activation has been associated mainly with elements of the lexicon in earlier work in generation (e.g. Dell, 1986), it is possible to associate activation with larger units as well. Syntactic categories and

syntactic segments could very well have an activation level which plays a role in the construction of syntactic structures. More precisely, the activation level of segments can determine their chance of being composed by means of (non-destructive) unification and the duration that unification will last.

For parsing, such a new architecture based on syntactic segments has been proposed by Kempen and Vosse (1989). Its control structure is governed by activation decay and simulated annealing. Its mode of processing is roughly described as follows:

‘Imagine syntactic tree formation taking place in a ‘mental testtube’ containing the segments and mobiles retrieved from the lexicon. Assume furthermore that their nodes continuously attempt to combine with other nodes they hit upon, and that the likelihood of a successful combination—or, rather, unification—depends not only on their feature composition but also on their level of activation. Whenever two nodes actually unify, they merge with one another and join together the segments (or trees/mobiles) they belong to, thus effectively creating a larger tree. Unifications are not granted the life everlasting, though. Depending on the strength of the original bond and due to activation decay, merged nodes may separate again, thereby clearing the way for other—maybe stronger—unifications. This dynamic process of coalescence, disintegration and reintegration will gradually come to rest accordingly as the segments in the testtube succeed in connecting up with one another in bonds of sufficient strength to lend stability to the resulting tree structure.’ (Kempen & Vosse, 1989:280)

It seems possible to apply this architecture to generation as well as to parsing. With each segment, an activation level is associated. Rather than simply choosing the first suitable segment in the grammatical encoding process (cf. De Smedt, 1990), all possible segments in the current context are activated and simultaneously try to combine with the current syntactic structure. The activation level of two nodes defines their probability of getting unified.

Although a particular unification might initially succeed, this structure may disintegrate: as more conceptual input enters the Formulator, another segment which perhaps fits better in the new context may take its place. Thus, the need for backtracking is eliminated in favor of a more general mechanism which is called *annealing*: a steadily dropping temperature decreases the chance of structures to disintegrate. Finally, a steady equilibrium state is reached with a ‘frozen’ configuration of segments, which is called a *conformation*. Due to space limits, I refer to Kempen and Vosse’s paper for more details.

The proposed architecture is compatible with assumptions concerning incremental formulation, parallelism, and the representation of a grammar and lexicon in terms of syntactic segments, as in SG. It combines the general structure-building power of a non-destructive unification mechanism with the context-sensitive choice capabilities of activation. For instance, due to contextual influences or simply the simultaneous presence of equally strong alternatives in the Unification Space, the ‘wrong’ segments may concatenate or furcate, thus accounting in particular for speech errors such as example (3a).

The development of this model is work in progress and has so far revealed some interesting problems. First, it is not clear how the Grammatical Encoder can be prodded to form just one coherent sentence rather than several isolated parts. This problem has been

solved by the requirement to arrive at just one conformation at the end of the annealing process. Second, it is not straightforward how the output of the Unification Space might become incrementally available to the next module in the generation process—the Phonological Encoder. If the speaker is to start uttering part of a sentence without immediately being forced to correct himself, then at least part of the sentence should be in a steady state. A solution may consist of ‘freezing’ *partial* structures, which signals to the Phonological Encoder that something is ready to be further processed.

## CONCLUSION

The research work presented in this paper adds flexibility and robustness to natural language generation systems by combining the best of two worlds: (1) the unification algorithm of a symbol processing approach, and (2) the concept of activation of connectionist systems. This research is aimed at an investigation of how such a hybrid model can deal with syntactic revisions during the generation of a sentence.

## REFERENCES

- De Smedt, K. & Kempen, G. 1987. Incremental sentence production, self-correction and coordination. In: Kempen, G. (ed.) *Natural language generation: new results in Artificial Intelligence, psychology and linguistics* (pp. 365-376). Dordrecht / Boston / Lancaster: Kluwer Academic Publishers.
- De Smedt, K. (forthcoming) Parallelism in incremental sentence generation. In: Adriaens, G. & Hahn, U. (eds.) *Parallelism in natural language generation*. Ablex.
- De Smedt, K. 1987. Object-oriented programming in Flavors and CommonORBIT. In: Hawley, R. (ed.) *Artificial Intelligence programming environments* (pp. 157-176). Chichester: Ellis Horwood.
- De Smedt, K. 1990. IPF: an incremental parallel formulator. In: Dale, R., Mellish, Ch. & Zock, M. (eds.) *Current research in natural language generation* (pp. 167-192). London: Academic Press.
- Dell, G.S. 1986. A spreading-activation theory of retrieval in sentence production. *Psychological Review* 93, 238-321.
- Karttunen, L. and Zwicky, A.M. 1985. Introduction. In: Dowty, D.R., Karttunen, L. & Zwicky, A.M. (eds.) *Natural language parsing: psychological, computational and theoretical perspectives* (pp. 1-25). Cambridge: Cambridge University Press.
- Kay, M. 1985. Parsing in functional unification grammar. In: Dowty, D.R., Karttunen, L. & Zwicky, A.M. (eds.) *Natural language parsing: psychological, computational and theoretical perspectives* (Chapter 7, pp. 251-278). Cambridge: Cambridge University Press.

- Kempen, G. & Hoenkamp, E. 1987. An incremental procedural grammar for sentence formulation. *Cognitive Science* 11, 201-258.
- Kempen, G. & Vosse, Th. 1989. Incremental syntactic tree formation in human sentence processing: a cognitive architecture based on activation decay and simulation annealing. *Connection Science* 1, 275-292.
- Kempen, G. 1987. A framework for incremental syntactic tree formation. In: *Proceedings of the 10th IJCAI*, Milan (pp. 655-660). Los Altos: Morgan Kaufmann.
- Kolk, H. 1987. A theory of grammatical impairment in aphasia. In: Kempen, G. (ed.) *Natural language generation: new results in Artificial Intelligence, psychology and linguistics* (pp. 377-391). Dordrecht/Boston/Lancaster: Kluwer Academic Publishers.
- Shieber, S.M. 1986. *An introduction to unification-based approaches to grammar*. CSLI Lecture Notes 4. Stanford: CSLI.
- Stemberger, J.P. 1985. An interactive activation model of language production. In Ellis, A. (ed.) *Progress in the psychology of language* (Vol. 1). London: Lawrence Erlbaum.
- Van Wijk, C. & Kempen, G. 1987. A dual system for producing self-repairs in spontaneous speech: evidence from experimentally elicited corrections. *Cognitive Psychology* 19, 403-440.
- Waltz, D.L. & Pollack, J.B. 1985. Massively parallel parsing: a strongly interactive model of natural language interpretation. *Cognitive Science* 9, 51-74.