POPEL-HOW: A Distributed Parallel Model for Incremental Natural Language Production with Feedback

Wolfgang Finkler
Deutsches Forschungszentrum fur
Kiinstliche Intelligenz GmbH
Stuhlsatzenhausweg 3
6600 Saarbriicken 11, FRG

Giinter Neumann
Lehrstuhl fur Computerlinguistik
Universitat des Saarlandes
Im Stadtwald 15, Bau 17
6600 Saarbriicken 11, FRG

fine@fbiOvax.mformatik.uni saarland.dbp.de

Abstract

This paper[1] presents a new model for the production of natural language. The novel idea is to combine incremental and bidirectional generation with parallelism. The operational basis of our model is a distributed parallel system at every level of representation. Starting point of the production are segments of the conceptual level. These segments are related to active objects which have *to* map themselves across the linguistic levels (i.e. functional-semantic, syntactic and morphologic level) as fast and as independently as possible. Therefore the incremental behavior caused by a successive input is propagated on all levels. Linguistic requirements detected by objects which are related to already produced segments at any level influence and restrict the decision of what to say next.

1 Introduction

In an *incremental* generation system the component determining the contents of the desired utterances *(what-to-say* component) can submit prematurely computed partial results to the *how-to-say* component in order to activate the production process before the planning is complete. Such systems simulate an important property of language production processes in human speakers: humans often start speaking before they know exactly what the whole contents of the utterance will be [Schriefers and Pechmann 88]. The how-to-say component must try to map its input structures across all representation levels as fast as possible to maintain the incremental behavior involved by the successive input. This results in speeding up the generator's output. A quick verbalization of parts of the contents is necessary if a user in man-machine communication must immediately react on this information (e.g. in systems for air traffic control). In the case that the mapping of a partial structure is not possible because of unspecified but required elements, it is necessary to request missing information from the preceding level. This allows the continuation of the production, set in motion by the how-to-say component. As a consequence, linguistic restrictions detected during the production provide *feedback* for the planning of what to say next [Hovy 87], [Reithinger 88]. Because the single partial structures shall be verbalized independently, we decided to consider them as active and autonomous objects which have to verbalize themselves. Then the whole utterance is produced in a distributed system of such objects. Therefore our model can be characterized as a combination of methods of *Distributed Artificial Intelligence* and natural language generation.

Up till now systems with interaction between the what-to-say and how-to-say component don't treat incremental formulation explicitly (e.g. [Hovy 87], [Appelt 85]) and incremental generation systems haven't considered a bidirectional flow of control (e.g. [DeSmedt and Kempen 87]). We present POPEL-HOW, a how-to-say component which for the first time realizes an incremental and bidirectional production which is based on a distributed parallel model at several linguistic levels, i.e. from conceptual structures to the utterances.

The component is part of POPEL [Reithinger 88 which is the generator of the XTRA dialog system [Allgayer et al. 89]. The integration in a dialog system suggests the common usage of the knowledge sources of the entire system. With the exception of the grammar, the same knowledge sources could be used for both analysis and generation. Because of the incremental and parallel processing in POPEL-HOW, a *declarative grammar* treating the specific requirements had to be developed.

2 Overview of the Architecture of POPEL-HOW

In POPEL-HOW there are four processing levels (see figure 1) according to the different knowledge sources which are used during the verbalization. The knowledge sources for the first two levels - the Conceptual Knowledge Base (CKB) and the Functional Semantic Structure (FSS) are defined using SB-ONE, a representation language similar to KL-ONE [Kobsa 89]. The next two levels are related to syntactic structures. The central knowledge source for

[1]The work presented here is being supported by the Gernan Science Foundation (DFG) in its Special Collaborative Program on AI and Knowledge-Based Systems (SFB 314), project NI (XTRA).

Thanks to Barbara Fairlight and Norbert Reithinger for their helpful comments on previous versions of this paper.

these levels is POPEL-GRAM, a unification grammar based on PATR-II [Shieber et al. 83]. The rules are divided into an ID- and LP-part in order to separate the construction of syntactic structures at the DBS (Dependency Based Structures) level and the linearization of these structures at the ILS (Inflected Linearized Structures) level[2]. The inflection of the stems is performed at this level by means of the package MORPHIX [Finkler and Neumann 88].

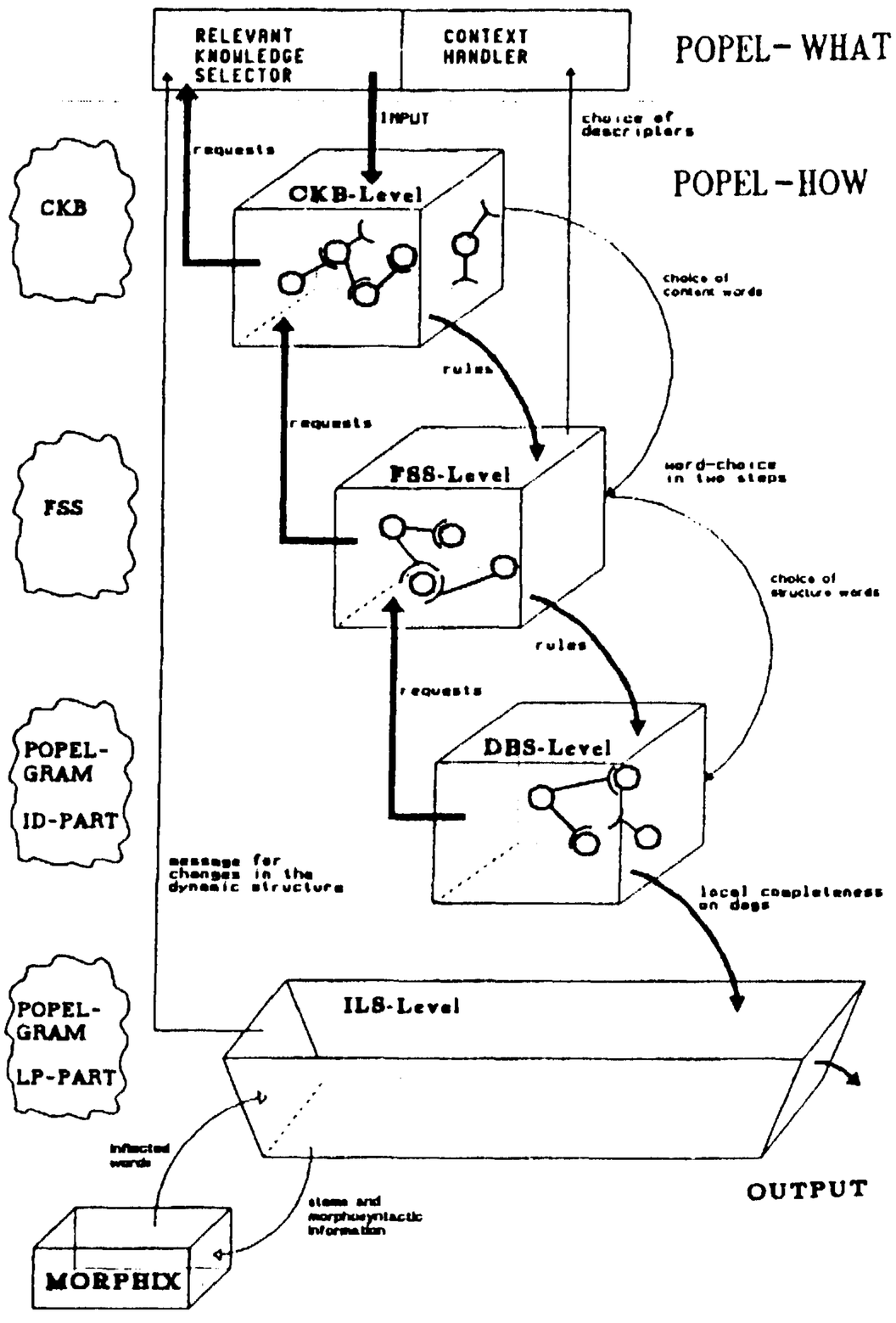


Figure 1: The Architecture of POPEL-HOW

The first three processing levels have been conceived with the same underlying *distributed parallel model* The production of structures at these levels is performed by a set of cooperating processes. Therefore the knowledge source of each level is decomposed into segments and related to the processes. All processes together represent the structure of the utterance constructed thus far. The main task of each process is the mapping of its segment into the next level by creating a new process which represents the goal structure of the mapping. If the mapping fails, the process can communicate with processes of the level above in order to request the missing information. It is possible that a process which receives such a request cannot handle it at its level. In order to fullfil the request such processes themselves send requests. In this way POPEL-WHAT, the what-to-say component of POPEL, can get *feedback* from POPEL-HOW which influences the planning of what to say next.

The distributed processing permits the incremental specification of the input for POPEL-HOW. Parts of the input can be mapped into the next level before new input structures arrive. Because the result of the mapping is also a process which tries to map its structure into the next level, the stepwise input of segments is propagated over all levels. Input which arrives later has to be integrated into the existing structures in order to allow an *incremental generation* at each level of description.

3 The Underlying Parallel Model

Our model follows the design methodology of *object-oriented concurrent programming* [Yonezawa and Tokoro 87]. Characteristic for such models is that there exist a collection of concurrently working self-contained *objects* which must interact in order to solve a problem cooperatively. An essential design issue is the question of how to decompose the overall problem into components which can be represented and handled by the objects.

When an object in POPEL-HOW has solved its part of the problem at a given moment, it does not yet terminate. The reasons are: first, it can send information to other objects in order to help them and second, because of the incremental processing in POPEL-HOW, new information can alter the object's task[3]. The relevant steps of the life cycle of an object are:

1. become acquainted with newer objects,
2. eventually establish communication links to these objects,
3. try to map the represented component into the next level,
4. send or answer requests,
5. start with the first step.

Every new object enters its address into an attendance list which is managed by a particular object. Objects already present in the system can then become acquainted with newer ones. An object tests whether a neighborhood relation to a found new object can be set up by a directed communication link. This decision must be performed synchronously by the objects and is made with regard to their

[2]The interpretation of the PATR-II formalism doesn't allow the definition of grammars which express both relations explicitly. Therefore the interpretation of rules has been modified; the context free part of rules at the DBS level is interpreted as the dominance part, the LP rules constrain the linear order of phrases.

[3]The termination of all objects happens only when no more input for POPEL HOW is specified. Of course, this criterion is determined only by POPEL-WHAT, which sends a termination message in this case.

underlying represented components. The set of the communication links forms the context of an object. During the mapping which is performed by local rules, an object compares the context with the condition part of a rule. According to the complexity of the rule, this step requires communication with linked objects. The result of a successful mapping is the creation of a new object at the next level. The new object is provided with patterns for the communication links. These patterns describe the relation between the objects of the level above and the represented components of their mapping results (see figure 2). The

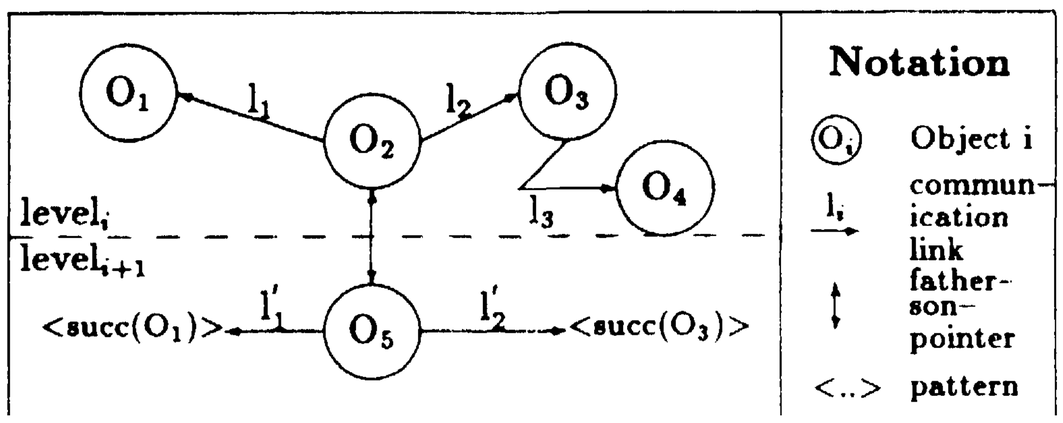


Figure 2: Objects on two Successive Levels

bidirectional link connecting father and son objects and the patterns of the son objects support the handling of requests between objects of successive levels. The answering of requests takes place asynchronously so that the object which receives requests is not interrupted needlessly often.

Because the presented program is the same for all objects, they have equal weights. Therefore the desired process model should be fair.

The Decomposition and Distribution of the Knowledge Sources

In order to use the declaratively formulated knowledge sources of each level efficiently in the distributed parallel model described above, it is necessary to decompose them into atomic components. The definition of the atomic units depends on the underlying formalism of the knowledge source. Basic units at the epistemological level of SB-ONE are concepts and their attribute descriptions for the general and individual level of representation [Kobsa 89]. Therefore we defined a concept plus its attribute descriptions as atomic component for the SB-ONE-based levels CKB and FSS. When a component is related to objects, the roles of the attribute descriptions correspond to the communication links of the object and the value restrictions are used as patterns to reduce the number of possible candidates for communication.

The ID-rules of POPEL GRAM constitute the knowledge source of the DBS-level in POPEL HOW. The linguistic basis of the grammar is *dependency theory,* a traditional approach of continental linguists (c.f. [Tesniere 59],[Engel 82]), which is expressed and applied in a modified PATR-II-formalism. ID-rules are denned as follows: There exists a particular constituent - the so-called head element - which determines the phrase by indicating the essential grammatical features and which restricts the number and quality i.e. syntactic function of the remaining constituents according to its valence.

Each ID-rule is regarded as an atomic syntactic unit and can be assigned to objects in the parallel model. This is done by associating a phrase or its head-element with an object and the dependent constituents with the communication links. Because of the explicit description of grammatical relations between constituents, the object representing the phrase has to maintain the feature specification as well. So the whole dag[4] of a phrase is assigned to the object.

The omission of the linear order at this level and the dependency relation defined between the constituents of a phrase allow an object-oriented view of syntactic structures and hence support the parallel and incremental production.

5 The Distributed Mapping between the Levels

The different representation levels must be related to each other in order to allow a mapping between conceptual units and the utterance. In our model the mapping has to be performed in a distributed way so that the components of the knowledge sources can be verbalized independently. For the knowledge sources based on SB-ONE, we define a set of local transition rules for every general concept. The precondition part of simple rules specifies attribute descriptions for the concept. The action parts of the mapping rules formulate relations between the detected elements of the left hand side and the structures to be created at the next level. In the case where a compositional and independent mapping of several structures is not possible for example if some conceptual knowledge units are represented in greater detail than the corresponding linguistics! knowledge - complex rules can be defined allowing the simultaneous mapping of structures consisting of several concepts which are connected by roles into structures at the next level.

Starting point for the distributed mapping of structures at the DBS-level are the dags of the objects. An object has to test whether its dag fulfills a condition of local completeness. This is true if the subdag of the head element contains no unspecified features. In this case the dag is sent to the ILS-level, where the last step of the mapping process takes place: the linearization, inflection and incremental output of the surface strings.

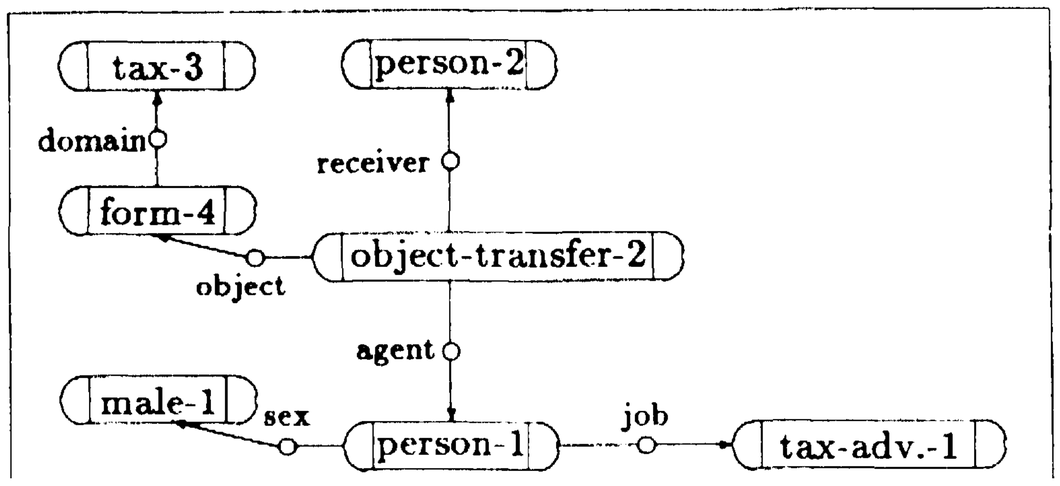
During the mapping the choice of words is performed in two steps. In the first step the *content words* (e.g. nouns) are selected in the course *of* the transition from CKB- to FSS-structures. The precondition of an applica-

[4] *Dags* (directed acyclic graphs) are the basic means of representation in PATR-II.

ble rule serves as a discrimination net to select the appropriate word. The choice of descriptors - the determination whether a nominal concept should be realized as a definite or indefinite nominal phrase, as an anaphora or should be suppressed in the case of elliptical verbalization - is done by a communication with POPEL-WHAT, which knows about previously mentioned individualized concepts. In the second step there is an additional choice *of function words,* for instance, prepositions. It is performed during the mapping into the DBS-level.

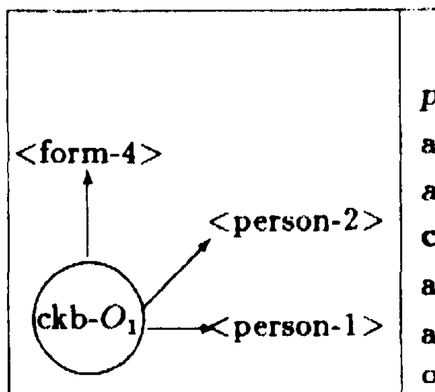
6 Processing in POPEL–HOW from the Local Viewpoint of one Object

In this section we describe a scenario of the production in POPEL–HOW. Starting point for the generator, especially for its first component POPEL–WHAT is the following individual CKB net:



POPEL-WHAT decides which structures of the individual CKB have to be included or excluded in a dialog contribution and the order in which they shall be uttered. The order is computed according to focus values which are extracted from the linguistic dialog memory [Reithinger 88]. The sequence of selected structures is passed to POPEL-HOW as input step by step.

In our example the first concept to be verbalized is object-transfer-2. At this moment this is the only known concept in the how-to-say component. POPEL HOW creates an object ckb-O_1 which represents this concept. The patterns of the communication links of this object (see next-picture) refer to objects which will represent the indicated individual concepts if these structures are passed as input later on. Because ckb-Oi is as yet the sole object, it tries to map itself into the next level immediately[5]. Therefore it checks the applicability of its rules.

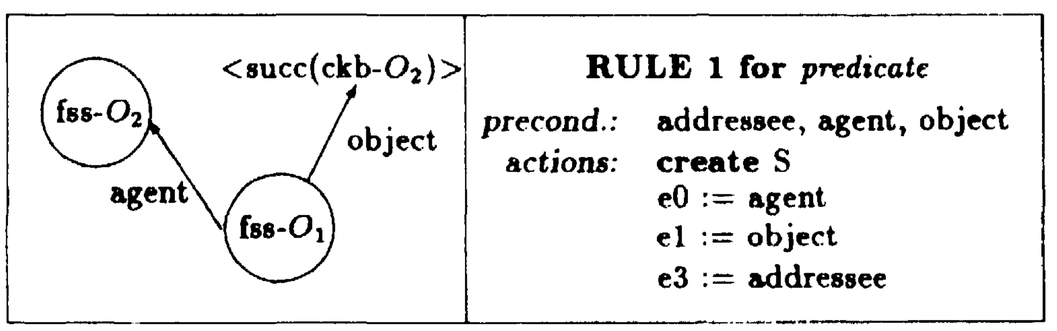


RULE 1	RULE 2
preconditions: agent, object, receiver	*preconditions:* agent, object
actions: **create** predicate agent := agent addressee := receiver object := object	*actions:* **create** predicate agent := agent object := object

[5] according to an object's life cycle; see section 3.

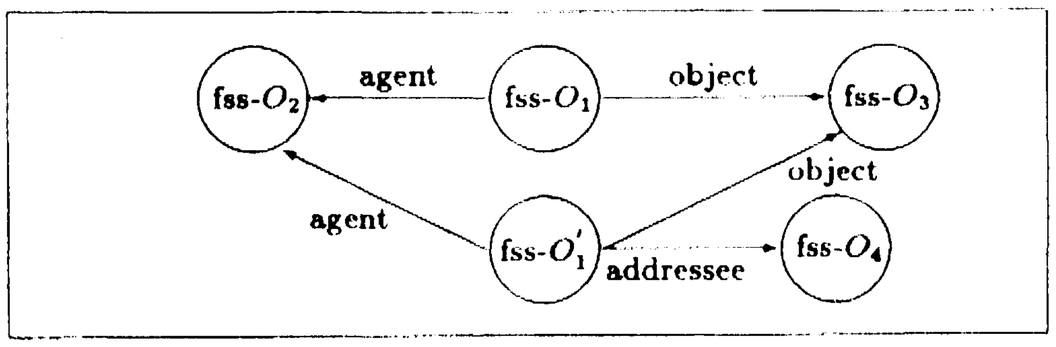
Because none of the rules is applicable, but on the other hand a mapping must be performed as quickly as possible to maintain the incremental behavior, the rule which specifies the fewest preconditions (rule 2) is chosen to request the missing information from POPEL-WHAT. In our scenario the request is satisfied. The objects ckb-0_2 representing form-4 and ckb-03 for person-1 are created and communication links to them are established by ckb-O_1.

Now ckb-Oi successfully applies rule 2. The result is the new object fss-O_1 which represents only a potential individual concept[6] at the FSS-level.



In the meantime another object fss-0_2 was created and linked with fss-Oi. The only rule for the mapping of a *predicate* in the scheme above requires one more role than is described in the context of fss-Oi. As a consequence, the father of fss-Oi is asked for the missing communication link. ckb-Oi searches for a rule which enables the creation of an object for *predicate* with a sufficient context. The application of the appropriate rule causes a request for an object representing person-2. If POPEL-WHAT didn't want to select this concept up till now, then the satisfaction of this request means that linguistic restrictions detected by POPEL-HOW affect the planning[7].

After the creation of the requested object ckb-0_4 and after its connection with ckb-Oi, fss-O_1 is the result of the application of rule 1 by ckb-O_1. This new object is in competition with fss-O_1 because of the same father. The next figure illustrates the net of the connected objects at the FSS-level after the objects ckb-0_2 and ckb-0_4 successfully transmitted themselves to this level.



Now the competitor fss-O_1' can apply its rule and create an object at the DBS–level representing a sentence phrase S. In the initial phase this object dbs-O_1 has to constuct its dag. Thereby the ID–rules of the phrase identifier S and the lexemes of the head and modifiers are used. The selection of the appropriate ID–rule(s) is constrained by the

[6] At this moment it isn't foreseeable whether fss-O_1 can verbalize itself. Hence no individualization of the general FSS concept *predicate* can be done.

[7] If the request isn't satisfied, fss-O_1 can only proceed by default-rules.

lexical information of the head element "geben" (to give). This means that the selection mechanism is *lexically driven.* When objects at the DBS-level install communication links they form a dependency-based connection of ID-structures. The explicit feature description of this connection has to be constructed by means of unification of the dags associated with the objects. This causes feature exchange and determination of the syntactic function of the dependent object.

Assume that in the meantime fss-0_3 has created object dbs-0_2 representing an *np* with an unspecified feature 'case'. During the set-up of the communication link *el* between dbs-O_1 and dbs-0_2 the syntactic function of the nominal phrase represented by dbs-0_2 is determined as "np in the accusative".

Because all relevant features of the dag of dbs-0_2 have values, the dag is now local complete and mapped to the ILS-level. The inflection and linearization of the lexemes in this dag produces the first part of the utterance: das formular (the form).

The object dbs-O_1 is not yet complete because an object representing the subject (the constituent named *eO)* is missing. A request has been made but is not propagated further because at this moment fss-0_2 creates its successor object dbs-0_3, which represents an *np* that becomes the subject of the sentence. After the feature exchange, the dags of both objects are complete. If dbs-0_3's dag arrives at the next level first, it cannot be uttered because no common linear order with the existing constituent can be given. So the new string: "der steuerberater" (the tax advisor) is retained. When the dag of dbs-Oi arrives, the lexeme for its head-element, the verb "geben" (to give) is inflected. Because the head -element dominates the existing phrases subject and direct object (eO and el), LP-rules are applicable that describe the linear order of the constituents so far. Therefore the next output is: gibt der Steuerberater (gives the tax advisor). Finally the *np* dem Kollegen (to the colleague) is uttered if the object ckb-0_3 succeeds in mapping itself across all representation levels.

7 The General Behavior of the Natural Language Production in our Model

The example in the previous section was presented in detail in order to demonstrate the production in POPEL-HOW. The produced utterance is influenced by the order of the stepwise input. If the input of the example is given in a different sequence, then the word order in the utterance can differ, too; e.g. the following sentence can be produced:

"Dem Kollegen gibt der Steuerberater das Formular."
("*To the colleague gives the tax advisor the form.")

Changes in the word order of the utterance are not the only consequence of another input sequence. The choice of words can be influenced because the application of alternative mapping rules results in the construction of different structures on the next levels. If the following input is passed to POPEL-HOW in the indicated sequence: concepts denoting the destination "Detroit", a "travelling" action, a traveller "Peter" and finally the vehicle "airplane", then the utterance:

"Nach Detroit fahrt Peter"
("*To Detroit goes Peter")
and its continuation: *"mit dem Flugzeug."*
("by airplane.") is produced step by step.

If the input sequence is: the vehicle, the person, the action and the destination, then the utterance will be:

"Peter flight nach Detroit."
("Peter flies to Detroit.")

Because the instrument of the action is known in due time, a verb which is more special than "fahren" (to go) and which incorporates the vehicle is chosen. Therefore the production of the prepositional phrase "mit dem Flugzeug" (by airplane) is suppressed.

In addition, it is possible that a different utterance is produced from the same input sequence. This happens if information gained by communication between objects in POPEL-HOW and the context-handler of POPEL-WHAT direct the application of different mapping rules and the choice of descriptions. For instance it is determined whether a noun phrase or a pro-word will be generated or whether a predicate shall be expressed as a verb phrase or a noun phrase.

The course of the generation process in the example of the previous section can be characterized as a synthesis free of conflicts. New structures are integrated into the already existing structure of the utterance. This means that the representation of the utterance at each level grows monotonicly. We call this special behavior monotonic incremental generation. Upward and downward expansion[8] as characterized in [DeSmedt and Kempen 88] for the syntactic incrementation take place at each level in POPEL HOW.

The incremental augmentation of the structure related to an object must be processed by this object itself in order to transmit its new quality into the next level. If the augmented object already possesses a successor, then a mapping creates an object which will become a competitor to the existing successor. If the competitor represents a more detailed segment than the previously existing object, in the sense that the competitor is an augmentation in comparision with the other object, the incremental behavior is still monotonic.

It is possible that objects in competition with each other don't represent more detailed segments but describe reformulations. The integration of these structures into the already existing one changes the structure in a non

[8]Insertion cannot occur in our model because two connected objects represent immediate constituents. Of course, objects representing non-immediate constituents can be put together by a combined upward and downward expansion.

monotonic manner. As a consequence, in addition to the already described operations, e.g., establishing communication links or unification, non monotonic operations are defined at each level in order to be able to handle reformulations. For instance an operation for the replacement of subdags in a dag in question is defined for DBS-objects. We denote this kind of generation as non—monotonic incrementation.

Managing the output in such an incremental system is not a trivial task. The order in the utterance is determined by indicating a desired order of specified concepts at the input level and is constrained by linearization rules at the ILS-level. However, in the parallel model the input sequence is generally lost. An immediate output of the phrases which first arrive at the ILS -level causes some problems because it can happen that new parts would have to be positioned in front of already uttered parts, or that the new parts have to augment or replace previously uttered fragments. If a filter is used in order to control the output of parts of the utterance then there is only a partially incremental production on the last step. However, if a phrase shall be uttered immediately then devices for 'self-corrections' are necessary as stated in DeSmedt and Kempen 87]. At present in POPEL HOW the parts of the utterance in question are repaired by repeating the utterance of the corresponding constituents.

8 Conclusions and Future Research

This paper has demonstrated how natural language production is performed incrementally based on a bidirectional and distributed parallel model. We showed that the incremental generation process across several representation levels can be influenced by linguistic restrictions. Such a model has not yet been investigated in research about natural language generation systems. Therefore we focused on describing the parallel generation process which requires a decomposition of all used knowledge sources and their distribution to active objects. For this reason we could not present the details regarding the linguistic background and the knowledge sources which would have demonstrated the linguistic ability.

The model has been fully implemented on a Symbolics 3640 lisp-machine under Genera. The simulation of the parallel machine was done by using the process facilities and the scheduler of the lisp-machine. Because of the formulation of a declarative generation grammar in PATR-II its augmentation can be performed easily. However the grammar writer has to pay attention to the application of the grammar in a distributed system.

Our next research concerns the elaboration of the distributed word choice, which we feel cannot be performed in such an independent way.

References

[Allgayer et al. 89] J. Allgayer, K. Harbusch, A. Kobsa, C. Reddig, N. Reithinger and D. Schmauks. XTRA: A natural-language access system to expert systems. *International Journal of Man-Machine Studies,* 1989. To appear.

[Appelt 85] D. Appelt. *Planning English Sentences.* Cambridge: Cambridge University Press, 1985.

[DeSmedt and Kempen 87] K. DeSmedt and G. Kempen. Incremental Sentence Production, Self-Correction and Coordination. In: G. Kempen (ed.), *Natural Language Generation,* pp. 365-376, Dordrecht: Martinus Nijhoff, 1987.

[DeSmedt and Kempen 88] K. DeSmedt and G. Kempen. *The Representation of Grammatical Knowledge in a Model for Incremental Sentence Generation.* Paper presented at the 4th IWG, Santa Catalina Island, 7 1988.

[Engel 82] U. Engel. *Syntax der deutschen Gegenwartssprache.* Berlin: Erich Schmidt Verlag, 1982.

[Finkler and xNeumann 88] W. Finkler and G. Neumann. MORPHIX: A Fast Realization of a Classification-Based Approach to Morphology. In: *Proceedings of the "Wiener Workshop Wissensbasierte Sprachverarbeitung" ,* Berlin, 1988.

[Hovy 87] E. Hovy. *Generating Natural Language Under Pragmatic Constraints.* PhD thesis, Yale University, 1987.

[Kobsa 89] A. Kobsa. The SB-ONE Knowledge Representation Workbench. In: J. Sowa (ed.), *Formal Aspects of Semantic Networks,* Los Altos, CA: Morgan Kaufmann, 1989. To appear.

[Reithinger 88] N. Reithinger. *POPEL: A Parallel and Incremental Natural Language Generation System.* Paper presented at the 4th IWG, Santa Catalina Island, 7 1988.

[Schriefers and Pechmann 88] H. Schriefers and T. Pechmann. Incremental production of referential noun-phrases by human speakers. In: M. Zock and G. Sabah (eds.), *Advances in Natural Language Generation,* Vol. 1, pp. 172 179, London: Pinter Publishers, 1988.

[Shieber et al. 83] S.M. Shieber, H. Uszkoreit, F.C.N Pereira, J.J. Robinson and M. Tyson. The Formalism and Implementation of PATR-II. In: *Research on Interactive Acquisition and Use of Knowledge,* Chapter 4, pp. 39 79, Menlo Park, California: Artificial Intelligence Center, SRI International, 1983.

[Tesniere 59] L. Tesniere. *Elements de Syntaxe Structural.* Paris: Klincksieck, 1959.

[Yonezawa and Tokoro 87] A. Yonezawa und M. Tokoro (eds.). *Object-Oriented Concurrent Programming.* Massachusetts Institute of Technology: The MIT Press, 1987.