

# Perspectives for Incremental MT with Charts\*

Jan W. Amtrup  
University of Hamburg. Comp. Sci. Dept.  
Vogt-Kölln-Str. 30, D-22527 Hamburg  
e-mail: amtrup@informatik.uni-hamburg.de  
Phone: (+49 40) 54 715-519, Fax: (+49 40) 54 715-515

February 1995

## Abstract

Obviously, human speech comprehension works in an incremental way: Unbiased introspection as well as evidence from psycholinguistics strongly suggest that we start to recognize a word before it has been completely uttered; even more, we may interrupt our conversation partner at that point if we disagree with what we think (s)he will be saying. Thus, incrementality is not restricted to any particular level of human speech comprehension, but is a general principle that is valid for all levels of processing — thus making simultaneous interpretation possible.

From a more technical point of view, incrementality would offer several advantages when constructing a machine interpretation system. The enormous processing amount that is necessary to cope with e.g. input consisting of continuous and spontaneous speech can partially be reduced by means of incremental processing.

There have been several investigations on the use of incrementality for natural language understanding at various levels. We will argue that the same properties can be used by the transfer step in a machine interpretation system.

We will show how techniques that are well known from the processing of other linguistic domains (especially parsing) can be adapted to transfer. A chart can be used to store partial and intermediate results during structural transfer. Empirical evidence is provided by the architectural subproject of Verbmobil.

---

\*This research has been funded by the Federal Ministry for Science and Technology within the framework of the Verbmobil joint research project under grant no. BMFT 01 IV 101 A/O

# 1 Introduction

This paper is concerned with the use of incremental techniques for machine translation, with a bias on transfer and the adaptation of mechanisms for incremental processing that are already being used for other aspects of the language understanding process. Obviously, human speech comprehension works in an incremental way. Informally this means that a device starts working on subparts of the problem input and that it even produces partial results before the input is complete. This piecemeal fashion of work takes place on a variety of levels and has strong influences on the abilities humans have.

The *cohort model* of Marslen-Wilson et al. is an example of a theory of word recognition that assigns an incremental property to this early stage of human speech processing. It assumes a competitive model of lexical selection in which, based on the sensory input at hand, several candidates for lexems are considered, one of which is finally recognized. The initial cohort is created during the first 200 ms of speech signal and is maintained until a lexical decision can be made at the *recognition point*. The only word candidate remaining in the cohort is assumed to be the one actually spoken. This is done regardless whether or not the signal is completed for that word: Cross-modal priming experiments show that a fragment such as /capt/ is as good a prime for a semantically related target word SHIP as the complete form /captain/. This processing schema helps us to recognize words and utterances nearly as fast as they are produced.

On a higher linguistic level there are also cues for the incremental nature of human speech performance. For example, Niv [7] argues that certain syntactic-functional decisions are being made during sentence processing with regard to some non-syntactical facts. A major principle claimed by Niv is to *avoid new subjects*, i.e. object readings of NPs are preferred over subject readings if the NP in question is not already known in the current discourse. He uses the fragment „The maid disclosed the safe’s location ...“ and shows that the continuation „...to the officer“ is preferred over „...had been changed“ which would require „the safe“ to be introduced into the current discourse by a subject NP. It is indicated by this example that syntax and discourse processing interact to a certain amount. This can only be the case if the syntactic processor works incrementally, especially if the first occurrence of a discourse entity as subject is positioned shortly after the introduction (which was probably done in an object position).

Interaction, which presupposes incremental operation, is not restricted to syntax and possibly higher levels, but also accounts for some effects in word recognition. E.g., if a sentence context strongly prefers one of the lexems currently competing in a cohort, this is reflected in the activation of that word within the cohort. This can again be shown with priming experiments. If subjects are provided with the context „The men stood around the grave. They mourned at the loss of their ...“ the fragment /cap/ reveals a priming for /captain/ compared to the target word /capital/, which is not supported by con-

textual information [10]. Thus, Incrementality is not only an accidental feature of human speech perception but also a feature that makes the comprehension process as effective and fast as it is<sup>1</sup>. Incrementality is a prerequisite for any kind of interaction between different levels of analysis, and it is used to reduce ambiguity as early as possible. It seems appropriate to investigate the principles of incrementality and interaction in the framework of language understanding, too. In the next section of this paper we will give a more formal definition of incrementality and provide some arguments for why we think that incremental processing should be a major goal for NLP. Even if one does not postulate that human cognitive principles should be mapped onto properties of an artificial device for the same purposes, there are applications where incrementality is a *conditio sine qua non*. Section 3 concentrates on one particular part of a machine interpreting system, the transfer module. We will give a sketch of a system designed with the chart paradigm applied for syntactic analysis and the following transfer based on syntactic features. The use of a chart as central representation device guarantees an incremental operation throughout the whole process of analysis and translation.

## 2 Incrementality: A closer look

In order to estimate the consequences of incremental processing we first need to define formally what incrementality is meant to be. We take the architecture depicted in Fig. 1 as a model for an incremental device. Let  $F$  be a functional unit that produces some result by applying operations to a given input. The input data to  $F$  is given as a set  $I$  of input elements  $A_i, i \in \{1, \dots, n\}$ . The  $A_i$  each arrive in  $I$  according to some order of arrival, i.e. there is a stream of input elements into  $I$ . All  $A_i$  are independent of each other in that their order of arrival is not predetermined by their content. This of course does not exclude the order from being associated with some property of the input data, e.g. time stamps. Indeed, a good example for  $I$  would be the set of word hypotheses that arrive as output of a word recognizer in the temporal order of the end points of the hypotheses.

The set  $O$  is made up of final results of  $F$  and is constructed analogously to  $I$ .  $O$  consists of output elements  $B_j, j \in \{1, \dots, m\}$ . Each result appearing in  $O$  is immediately transmitted to any other system attached to the output stream of  $F$ .

Intermediate results  $C_k, k \in \{1, \dots, p\}$  are stored in a set  $Z$ . We assume that no processing is done in  $I$  or  $O$ , so that each operation  $F$  either changes the content of  $Z$  or produces new output in  $O$ . A result in  $O$  is calculated by applying a function  $f$  to a subset of input elements and intermediate results:

---

<sup>1</sup>There seems to be evidence that interactive behavior of the human speech processor is restricted to the cases in which there is a benefit for the computational process at hand (cf. [8])

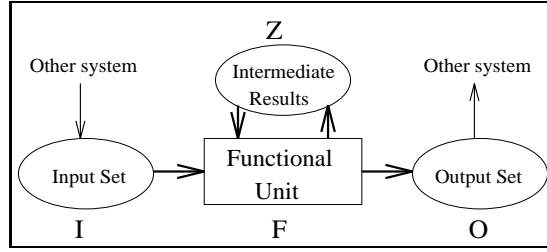


Figure 1: The architecture of an incremental system

$$B_j := f(D_j), D_j \subset I^* \times Z^*$$

Now we can define the state of the system  $F$  by examining the contents of the three sets  $I$ ,  $O$  and  $Z$ :

At every time  $t$  with  $0 \leq t < \infty$  the state of processing in  $F$  is given as the actual contents of  $I(t)$ ,  $O(t)$  and  $Z(t)$ .

There are two extreme situations  $F$  may be in. In the initial situation, all three sets are empty:

$$I(0) = \emptyset \wedge O(0) = \emptyset \wedge Z(0) = \emptyset$$

On the other hand, processing is completed if no changes occur on these sets. All input elements have been taken into consideration, all output elements have been created:

$$t_{\text{End}} := t : I(t) = \{A_1, \dots, A_n\} \wedge O(t) = \{B_1, \dots, B_m\} \wedge Z(t) = \{C_1, \dots, C_p\}.$$

Now, there is a threefold characterization of incrementality<sup>2</sup>:

- Incremental input. The system  $F$  starts to work on input items without waiting for all input elements:

$$\exists t : \#(I(t)) \neq n \wedge Z(t) \neq \emptyset$$

An example application might be a NLP system capable of answering Yes-No-questions which starts to process input as soon as it arrives but which cannot issue the answer until all input has been assessed.

---

<sup>2</sup>These definitions are formalizations of hypotheses shown in [5]

- Incremental output. Some elements of the output are presented before all processing is complete, i.e. before  $I$  and  $Z$  both got their final content:

$$\exists t : \#(I(t)) \neq n \wedge \#(Z(t)) \neq p \wedge O(t) \neq \emptyset$$

You will get an impression of incremental output by thinking of a system that reads out sentences: It gets a whole sentence and reads it out while computing intonation step by step.

- Incremental system. An incremental system works on parts of the input and starts to issue output elements before the input is complete:

$$\exists t : \#(I(t)) \neq n \wedge O(t) \neq \emptyset.$$

An incremental parser that reads its input word by word, it then processes the input and delivers partial hypotheses about growing portions of it, is an example for an incremental system.<sup>3</sup>

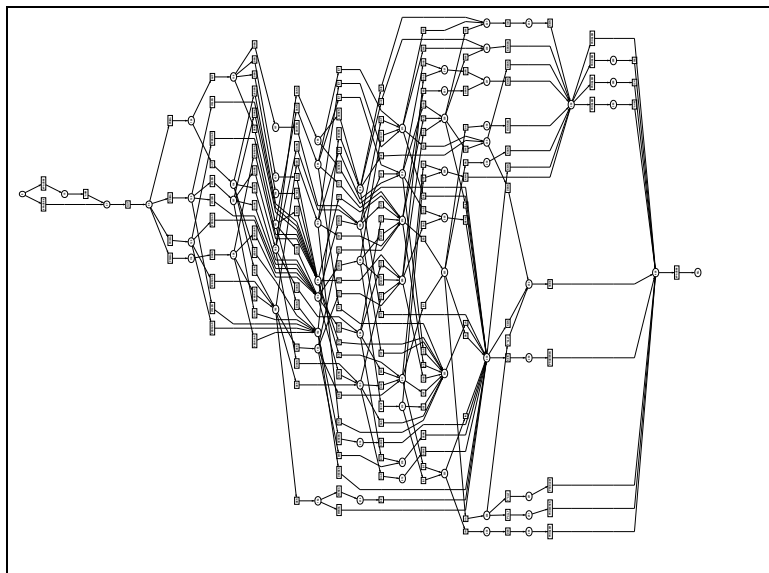


Figure 2: A word graph representing the utterance „Gut prima vielen Dank dann is das ja kein Problem“ (Approx. „Fine thanks then this is no problem“)

At least at first glance, the introduction of incrementality leads to a computational overhead. If we consider a parser in a speech interpreting task, the

<sup>3</sup>Incremental input and incremental output can be combined without yielding an incremental system. In such systems there is a time  $t$  such that  $\#(I(t)) = n \wedge \#(Z(t)) \neq p \wedge O(t) = \emptyset$ , i.e. input and output do not overlap.

input to the parser could consist of a word graph as shown in Fig. 2. The graph is organized from left to right, the starting point of the utterance being the leftmost circle. Word hypotheses are depicted by rectangular boxes, alternative hypotheses being indicated by branching. Such word graphs can represent a great variety of different utterance hypotheses in a compact manner. The overhead produced by an incremental system produces is due to the fact that a word graph of this shape cannot be generated incrementally: Only a non-incremental system is able to delete all „dead ends“ within the graph prior to parsing. An incremental system cannot decide whether or not a specific path within the graph will possibly be extended in the future, finally leading to a complete utterance hypothesis. Thus, the input to the parser in an incremental system is much broader and more tree-like, with only a subset of the paths leading to the endpoint of the graph.

But then again, even in this case, there are clear advantages of incremental behavior. It is possible to interleave word recognition and parsing and to cut off branches of the graph that would never lead to a syntactically valid parse<sup>4</sup>. A component is able to build up interpretations at a very early stage and to prune misleading hypotheses in other components at the moment of detecting the inappropriateness. Another advantage of incremental operation is the possibility to generate predictions. Assumptions about the properties of future input may be made by a component and transmitted top-down to other components which can use these expectations to improve their performance.

Besides from being of some value concerning efficiency or elegance, some properties and applications are only feasible if the components constituting a system exploit incrementality:

- The introduction of *Parallelism* is impossible if only one component processes input at any given time.
- The *Keeping of Hard Time Constraints* which arises in real-world applications like dialog interpretation can only be attempted by using incremental and interactive architectures that can create time pressure (cf. [6]).
- *Overlapping Input and Output* is a demand for some applications, e.g. simultaneous interpretation.
- *Turn Taking*, the ability to interrupt a conversation partner and to take over initiative, is dependent on understanding the dialogue contribution up to the present point.

---

<sup>4</sup>These branches need not be considered further by the word recognition component. See [3] for an example of this schema.

### 3 Transfer using Charts

In order to construct a device that aims at simultaneous interpretation, one has to implement incremental components throughout the system. We will now describe the way a data structure originally introduced for parsing can support the concept of an incremental transfer component. Afterwards, we will present a program using this model and some of its properties.

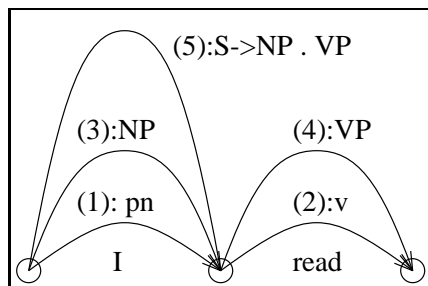


Figure 3: An analysis chart

The notion of a chart was introduced by Kay [4] as a data structure for parsing. A chart is a graph whose nodes represent points in time<sup>5</sup> and whose edges correspond to words or phrases. All results of the analysis are thus stored in the chart. It extends the concept of a *wellformed substring table* by allowing storage of incomplete derivations, too. This property divides the set of edges in the chart into two classes: *Inactive edges* correspond to words or completely analyzed constituents, whereas *active edges* represent incomplete constituents. They are normally related to phrase structure rules with right sides which have not yet been completely processed. In Fig. 3, edge (4) is inactive, while edge (5) is active, waiting for a VP to be complete. The *fundamental rule* underlying chart parsing could be used to combine both edges in order to establish an inactive edge spanning both words and describing the complete sentence.

Charts offer a great amount of flexibility to the implementor. The central data structure that holds all complete and incomplete partial results, together with the introduction of an agenda, which lists the tasks yet to be carried out, enables the designer to specify an analysis algorithm that abstracts from actual strategies for processing and search. This separation of *what* from *how* allows for a pragmatic insight into the computational processes affecting the analysis.

There are several analogies between syntactic analysis and transfer. A partial translation could in fact be used many times during operation. With incremental transfer there are several competing analyses. In most cases, one can not tell in advance which of them finally will turn out to be the best one. Thus, a

<sup>5</sup>For written input, word junctures are used as nodes; if we assume speech input using word graphs, nodes directly represent points in time.

mechanism for storing partial results like a wellformed substring table is very useful. The extension into the direction of a transfer chart is straightforward: Incomplete analyses correspond to partially transferred items that have at least one open transfer equation.

Open transfer equations may be encountered in compositional transfer. Many rules for structural transfer require constituents to be transferred recursively. A transfer rule as represented in Fig. 4 is responsible for the head-switching cases of German-English adverb translation. It describes translations of some German adverbs into English verbs as contrasted in the following examples:

<p><i>Ich lese gerne</i> I read likelyly “I like to read.”</p>	<p><i>Ich lese täglich</i> I read daily “I read daily.”</p>
--	---

The rule states that a source language verbal phrase consisting of a verb and an adverb has to be translated into an infinitive construction in the target language if the source adverb is translated as verb. A partially transferred verbal phrase could have been processed including the main verb. The remaining unsolved transfer equation calls for the translation of the remaining adverb.

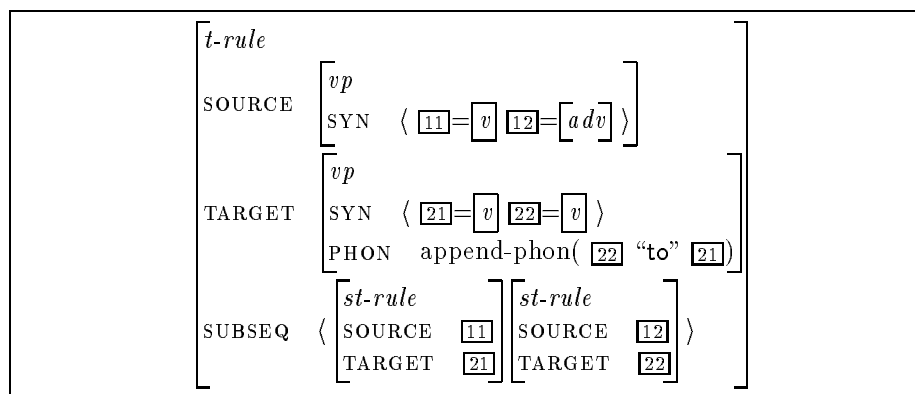


Figure 4: Transfer rule for head switching triggered by German adverbs

What are the elements of a transfer chart? Starting from a simple approach we assume a transfer chart to be an extension of an analysis chart that could have been used for parsing. Annotations constitute the transfer chart: Every inactive analysis chart edge is the root of a tree of transfer edges. Thus, translations are directly derived from analyzed syntactic constituents. Each node at the first level of the rooted tree is the result of the application of one transfer rule to the given constituent. The number of applicable transfer rules determines the number of branches of the tree. If one of the transfer rules contains recursive transfer equations, subsequent levels of the tree are built. A daughter node within the tree is created by solving exactly one transfer equation which has

been open in the description of the mother node.

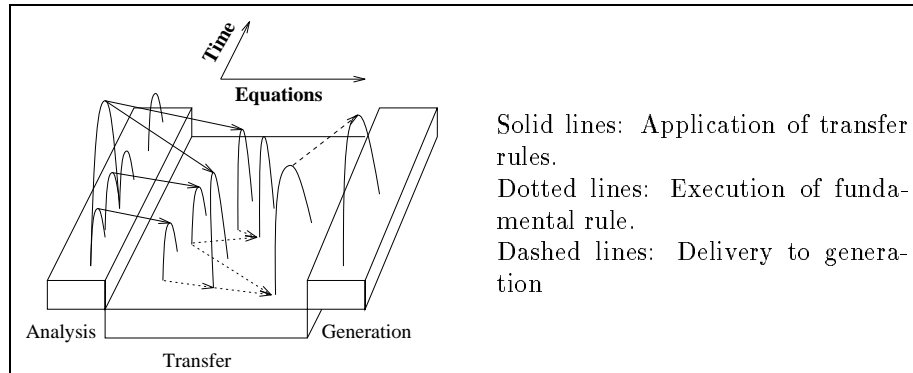


Figure 5: The transfer chart mediates between analysis and generation

Overall, the transfer chart can be represented as two-dimensional structure (Fig. 5). It contains one temporal dimension that describes the progress of time within the input signal, and one dimension for the decreasing number of unsolved transfer equations that describe the progress while constructing target language equivalents. Each transfer edge belongs to an analysis edge which provides important information such as start and end time, confidence of the described constituent as well as the original feature graph of the derivation. The transfer equations still to be solved control the activity of an edge: If there are open equations the edge is called active, otherwise it is inactive. One can say that active transfer edges describe incomplete translations of a constituent while active analysis edges refer to incompletely recognized constituents.

The principal course of processing runs in the following procedure: Inactive analysis edges that represent completely analyzed constituents trigger the introduction of the first level of the associated transfer edge tree: Every transfer rule that is applicable to the constituent generates an initial transfer edge that is inserted into the chart.

Following this insertion, the fundamental rule of the algorithm consists of searching two transfer edges that can be glued together. The combination of two transfer edges makes sense, if one of them (call it the big one) contains an unsolved transfer equation, including a source language part that can be unified with the source part of the other edge (the small one). This being done, a second unification with the target part has to occur; the newly constructed edge can be inserted into the chart.

To give an example of the results of the application of chart transfer to syntactically analyzed constituents, Fig. 6 shows how the verbal phrase „lese gerne“ is represented just before it is being transmitted to generation.

Charts as a basis for transfer offer several advantages concerning architec-

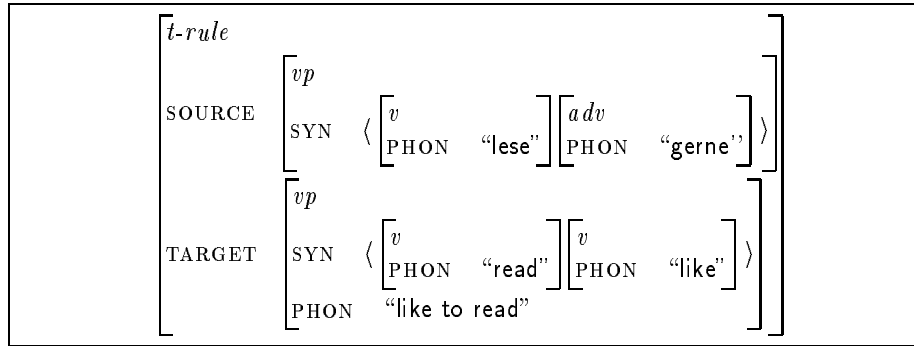


Figure 6: Example of the translation of „lese gerne“

tural issues. We list some of them below:

- *Re-usability*: The translation of “lese” can be used for the two possible alternative translations which can not be decided upon before the adverb is encountered. Once the translation into “read” has been performed, the transfer lexicon does not have to be consulted for that lexical item again. Other transfer rules simply reuse translation values recorded within the transfer chart. Obviously, this effect also extends to translation of more complex constituents, which was not shown here.
- *Incrementality*: The translation of the verbal phrase can be carried out incrementally. Incremental input is processed by transferring partial analyses as soon as they arrive. These results enter the transfer component in a left-to-right fashion beginning with single words and continuing with larger partial analyses of constituents. Consequently, transfer starts with lexical items, and it processes larger chunks of the input whenever they appear. Incremental output can be delivered to further components. The transfer component may come up with results at the moment a complete translation of a source language constituent has been created. Consequently, a transfer module based on chart algorithms can in any respect be characterized as incremental system.
- *Interaction*: There is no need for interaction within a system for a limited domain as the one described. But in a larger context, the introduction of communication tasks into the transfer agenda would allow this component to trigger communication with other subsystems and to suspend execution of the original task until an answer from the consulted subsystem has arrived.

- *Parallelization*: Parallelization generally possible for transfer since the chart is a directed graph. Workload can be distributed between some processors e.g. by attaching a subset of the edges to each processor. Approaches that exploit such strategies have been examined for parsing (e.g. [9]) and can be applied to transfer.
- *Multiple hypotheses*, which often are problematic for other transfer models, are the basis of any chart-based processing schema. Whenever two edges have identical start and end vertices, they represent alternative (and thus competing) descriptions of the input signal between the two vertices. A transfer module may translate all alternative analyses, and another component will eventually decide which translation is most adequate according to the standards represented in the system. Despite of this being an advantage of chart based mechanisms for transfer, this raises a new problem: How should the size of translation units be controlled? How can a generation component cope with a stream of partial translations instead of getting delivered a single translation of the input? There are approaches to generation that are able to cope with incremental input by generating repairs [2], but further research has to be carried out in order to investigate the detailed consequences.

## 4 Conclusion

Incrementality and interactivity are central properties of the human speech understanding device. We have shown that these features are also advantageous for artificial mechanisms for speech processing, especially if one considers simultaneous interpretation as a task performable by machines. We have dealt with the formalization of incrementality from various viewpoints and described a chart-based transfer component capable of handling transfer problems incrementally like in the case of head switching. We listed properties of the system and described the two-dimensional structure of the search space.

The system we have built consists of a chart parser to generate the input specifications to the transfer component, and of the transfer component itself [1]. It is implemented using Common Lisp and CLOS. Later on, we expect the system to be an integral part of the interactive prototype built within the architectural subproject of the German joint research project *Verbmobil*.

## References

- [1] Jan W. Amtrup. *Transfer and Architecture: Views from Chart Parsing*. *Verbmobil Report 8*, University of Hamburg, March 1995.

- [2] Wolfgang Finkler and Anne Schauder. Effects of Incremental Output on Incremental Natural Language Generation. In *Proc. of the 10<sup>th</sup> ECAI*, pages 505–507, Vienna, Austria, August 1992.
- [3] Andreas Hauenstein and Hans Weber. An Investigation of Tightly Coupled Speech Language Interfaces Using an Unification Grammar. In *Proceedings of the Workshop on Integration of Natural Language and Speech Processing at AAAI '94*, pages 42–50, Seattle, WA, 1994.
- [4] Martin Kay. Algorithmic Schemata and Data Structures in Syntactic Processing. Technical Report CSL-80-12, Xerox Palo Alto Research Center, Palo Alto, 1980.
- [5] Anne Kilger. Using UTAGS for Incremental and Parallel Generation. *Computational Intelligence*, 10(4):591–603, November 1994.
- [6] Wolfgang Menzel. Parsing of Spoken Language under Time Constraints. In T. Cohn, editor, *Proc. of the 11<sup>th</sup> ECAI*, pages 560–564, 1994.
- [7] Michael Niv. *A Computational Model of Syntactic Processing: Ambiguity Resolution from Interpretation*. PhD thesis, Univ. of Pennsylvania, 1993.
- [8] Richard Shillcock and Ellen Gurman Bard. Modularity and the Processing of Closed-class Words. In Gerry T.M. Altmann and Richard Shillcock, editors, *Cognitive Models of Speech Processing: The Second Sperlonga Meeting*, chapter 9, pages 163–185. Lawrence Erlbaum, Hove, UK, 1993.
- [9] Henry S. Thompson. Chart parsing for loosely coupled parallel systems. In *Proc. International Parsing Workshop*, pages 320–328, Pittsburgh, Pa., 1989. Carnegie Mellon University.
- [10] P. Zwitserlood. The locus of effects of sentential-semantic context in spoken-word processing. *Cognition*, 32:25–64, 1989.