the CM parser for the different mappings. Note that the pre-processing step is only performed once – at the time the grammar is defined.

In the current CM implementation, a "coarse-grain" emulation of the PRAM algorithm is used. More specifically, the number of CM processors used in $| G |^2$ and the run-timeis $O(n^6 log | G |)$. The motivation behind this coarse-grain mapping is that for NL parsing, $| G | \gg n$; in particular, n is rarely more than 20. (This is direct contrast to parsing programming languages where $| G |$ is small but n, the length of the program to be parsed, can be arbitrarily large.)

The CM parser currently being developed is for a small grammar consisting of 55 trees which expands to approximately 200 grammar nodes (Yves Schabes' Small English Lexicalised TAG). Initial performance measurements indicate that the run-time is linear in n, rather than the theoretical $O(n^6)$ run-time. The next stage of the project is to enlarge the TAG and to measure the run-time of the CM with respect to both grammar size and sentence length. ¿From this experimental data, we hope to verify the logarithmic behavior of the run-time with respect to grammar size.

Tree Adjoining Grammar, Segment Grammar and Incremental Sentence Generation

Gerard Kempen, Koenraad DeSmedt
NICI
Department of Psychology
University of Nijmegen
NL/6525 HR Nijmegen, Netherlands
KEMPEN or DESMEDT@KUNPV1.PSYCH.KUN.NL

The cognitive process of syntactic structure formation is **lexically guided**, both in production and in parsing. "Lexicalized" grammars are therefore likely to figure prominently in psycholinguistic processing models. Tree Adjoining Grammars (TAG) and Segment Grammars (SG) are two such formalisms. They are similar in that they both use subsentential structures as building blocks: elementary trees (or mobiles) which are larger than individual nodes. At least one terminal node of a building block is a lexical node (as implied by the definition of lexicalized grammars).

A second property of human syntactic structure formation is **incremental generation**. This feature imposes special demands on the syntactic processor and its associated grammar. In our talk we evaluated TAG and SG from the point of view of the following three demands:

1. The processor should be capable of incrementing the current (incomplete) syntactic structure in any direction (leftward or rightward) and by any method (upward expansion, downward expansion and insertion).

2. Not only phrase- and clause-sized incrementsshould be allowed, but word-sized increments as well.

3. Syntactic coordination (inclusive of reduction phenomena such as gapping) should closely resemble the treatment of self-repair in spontaneous speech. (For example, the repair text can **refer back** to the reparandum text; this suggests that, during the computation of the repair text, the reparandum's structure is not destroyed. We hypothesize that reparandum and repair are "coordinated" in a way similar to the members of a conjunction).

We explained the workings of SG, compared it to lexicalized TAG and evaluated both in terms of the three demands.

The discussion and informal conversation revealed that the most important difference between SG and TAG resides in the fact that TAG uses only one level of syntactic representation, whereas SG distinguishes two levels: Functional (or F-) structures and Constituent (or C-) structures. Y. Schabes suggested informally that the mapping between C- and F-structures could be formalized in terms of S. Shieber's & Y. Schabes' **Synchronous TAG**. We added to this t he suggestion that one might consider a system performing a double mapping: Between Semantic and F-structures, and hetween F- and C-structures, and that this, in turn, could considerably simplify the complexity of the (TAG-style) syntactic structures in the 'middle' layer. For instance, we suspect that only 'canonical' trees suffice (as in SG F-structure), and that their expansion to tree families is no longer needed: this work is replaced by the F-to-C-structure mapping. These ideas deserve further scrutiny.

Incremental Natural Language Generation with TAGs in the WIP Project

Wolfgang Finkler
DFKI
Stuhlsatzenhausweg 3
W-6600 Saarbrücken 11
finkler@dfki.uni-sb.de

In my talk, I argued that lexicalized Tree Adjoining Grammars with unification are useful for the incremental processing at the syntactic level of description. In order to motivate the need for incremental natural language generation in the WIP project I gave a short overview of the system to present the specific requirements upon its natural language generation component.

Incremental generation means the immediate verbalization of the parts of a stepwise computed message. It is psychologically evident that humans often start speaking before they know exactly what the whole contents of their utterance will be. Because the WIP system shall be usable in scenarios where information to be presented is continuously supplied by an application system and where such information must be simultaneously presented in a condensed way to assist human decision makers – there is a need for incremental presentation.