behave in linear time on some grammars, in $O(G^2n^4)$ worst time for unambiguous TAGs and in general in $O(G^2n^6)$-time in the worst case. An earlier Earley-type parser that we proposed in 1988 maintains the VPP but at its cost of its worst case complexity ($O(G^2n^9)$-time). To our knowledge, it is the only known polynomial-time general TAG parser that maintains the VPP. Both Earley-style parsers for TAGs use top-down filtering and therefore their behaviors are in practice superior to pure bottom-up parsers (as Joshi's and Vijay-Shanker's adaptation of CKY algorithm to TAG).

In practice, the importance of the VPP varies from grammars and is currently being evaluated on natural language TAG grammars for English and French.

Parallel TAG Parsing on the Connection Machine

Michael Palis, David Wei
Department of Computer and Information Science
School of Engineering and Applied Sciences
R-555 Moore School
University of Philadelphia
220 South Street 33rd Street
Philadelphia, PA 19104-6389, USA
palis@linc.cis.upenn.edu

We present a parallel parsing algorithm for Tree Adjoining Grammars (TAGs) and its implementation on the Connection Machine (CM). The CM TAG parser is designed to handle TAGs of arbitrary size without significant decrease in performance. Specifically, the expected run-time of the parallel algorithm is logarithmic in the grammar size (as opposed to quadratic in a serial implementation).

The CM TAG parser is an emulation of the CRCW PRAM algorithm. The PRAM algorithm is characterized by frequent communication between processors via the shared memory. Moreover, the pattern of inter-processor communication does not have the regular structure often found in many parallel numerical algorithms. Because the CM has a distributed memory, the emulation of the PRAM algorithm can only be realized by explicit message-passing, albeit between non-adjacent processors. Unfortunately, routing messages between non-adjacent processors is time-consuming on the CM. The CM uses a deterministic oblivious routing strategy, which, in the worst-case, can introduce $\sqrt{p}$ delay per emulated step, where p is the number of processors used.

To obtain a more efficient emulation, we employ randomization: i.e., grammar nodes of the TAG are mapped randomly to corresponding CM processors. In theory, this reduces the delay per emulated step to $O(log(p))$ with high probability. In practice, we use randomization as part of a pre-processing step: given a fixed TAG, we generate several random mappings of the TAG to the CM, then choose the most efficient mapping. The most efficient mapping is obtained experimentally by running

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.