Parallel Parsing:
Different Distribution Schemata for Charts

Jan W. Amtrup, Univ. of Hamburg
e-mail: amtrup@informatik.uni-hamburg.de

Introduction

We are going to present results from two experiments designed for parallel parsing within the chart paradigm. Parallel processing gains more relevance as applications become increasingly complex and nets of workstations as well as dedicated parallel computers are widely available. A chart-based parser is well suited for approaches to parallelism due to the identification of almost independent data objects that a chart is made of. A parallelization based on tasks of an agenda is only suitable for shared memory systems with a tight coupling, the choice of individual edges as autonomous agents may result in too many processes. But both nodes of a chart and rules of a grammar may provide sufficient possibilities for parallelization in a loosely coupled framework.

Context-free parsing in a net of workstations

These latter two aspects — node-based and grammar-based approaches — of a data-driven parallelization of chart parsing have been pursued in two experiment sets (for other experiments, confirm e.g. Thompson (1994)). The first system was designed to parse written input on a workstation cluster. It operates with a context-free grammar and uses chart nodes as basis for the distribution of work. A set of chart nodes is assigned to each processor (workstation). It is stored combined with all edges emerging from the given vertices. The configuration and initial synchronization is carried out by an additional process that at the same time functions as a user interface. We use a complete graph as the connection scheme between the processing components. Communication is effected by extensions of the interpretative Scheme language and is based on UNIX sockets.

The chosen corpus consists of simple sentences from the blocks world (like "The green pyramid lies on the red case"), sentence length varying from 2 to 14 words producing 35 to 172 edges for a parse. We experimented with different configurations of up to three machines. Analysis times for the sample varied from 22 to 52 seconds, a speedup of 2.36 could be observed by using three processors compared to one.

Unification-based parsing with transputers

The results of the first experiment set have to be critically reviewed for different reasons: First, due to the low execution speed of the Scheme dialect used and the resulting long processing time, the speedup values seem to be unrealistically high (extremely slow programs win a lot by

duplication of resources). Second, the results may not generally be valid because of the limited usefulness of a context-free formalism for adequate NLP applications.

Thus, we designed a second system (Amtrup, 1992) that uses an efficient, compiling programming language (C) and a unification-based formalism for grammar and lexicon. A transputer system became the hardware basis for the system which was configured to a ring topology. The program consists of computational processes, a dedicated user interface process that connects the transputer subsystem to the outer world, and routing facilities on each processor to support data flow within the communication network.

We examined two different styles of parallelization. Additionaly to the node based distribution schema described above, we introduced the possibility to assign subsets of the grammar rules to different processors. The system is able to parse sentences from a medical domain (German thorax radiology reports) which mostly consist of complex noun phrases (e.g. "Strahlentransparenz der linken Lunge ohne Hinweise auf frische Infiltrate"). The formalism used to describe grammar and lexicon is a slight modification of PATR II, implemented in C and using a linear unification algorithm. Feature structures are stored as compact arrays of feature node definitions and can be transferred from one process to another without the necessity of linearization and reconstruction.

The grammar used for the tests described here contains 30 context-free rules with some 70 constraint equations. The lexicon consists of 248 entries. The test sentences varied in length between 4 and 18 words, resulting in 152 up to 1382 edges when parsed on a monoprocessor. Again, we studied different configurations using up to seven transputers. The time span necessary for analyzing the sentences varied from 2 to 22 seconds using a large configuration. The speedup values ranged between 1.34 and 3.08 when using seven transputers instead of one. These ratios are naturally lower than that of the context-free parser, but nevertheless promising. It turned out that a parallelization based on a hand-partitioned grammar was more stable than a node-based parallelization which to a certain degree depends on the actual input sentence.

Conclusion

We have shown that parallelization may increase the performance of chart parsers. While one experiment was conducted to get insights into the mechanisms in principle, the other one aimed at results that could be of practical relevance, too. The speedup ratio of 3.08 with sevenfold resources seems promising. Nevertheless, further investigation is necessary for evaluating the application of the described techniques to speech input. Speedup values grew with the sentence length in the second example, which would suggest that a parallel chart-parsing model can render a better performance when processing speech-data magnitudes larger than written input.

References

Amtrup, Jan W. 1992. Parallele Strukturanalyse Natürlicher Sprache mit Transputern. ASL-TR 44-92/UHH, Univ. of Hamburg.

Thompson, Henry S. 1994. Parallel Parsers for Context-Free Grammars — Two Actual Implementations Compared. In Geert Adriaens and Udo Hahn, editors, *Parallel Natural Language Processing*. Ablex Publishing Corp., Norwood, NJ, chapter 3, pages 168–187.