Efficient Large-Scale Parsing — a Survey

John Carroll
Cognitive and Computing Sciences
University of Sussex
Brighton BN1 9QH, UK
johnca@cogs.susx.ac.uk

Stephan Oepen
Computational Linguistics
Saarland University
66041 Saarbrücken, Germany
oe@coli.uni-sb.de

Abstract

We survey work on the empirical assessment and comparison of the efficiency of large-scale parsing systems. We focus on (1) grammars and data used to assess parser efficiency; (2) methods and tools for empirical assessment of parser efficiency; and (3) comparisons of the efficiency of different large-scale parsing systems.

1 Background

Interest in large-scale, grammar-based parsing has recently seen a large increase, in response to the complexities of language-based application tasks such as speech-to-speech translation, and enabled by the availability of more powerful computational resources, and by efforts in large-scale and collaborative grammar engineering and also in the induction of statistical grammars/parsers from treebanks.

There are two main paradigms in the evaluation and comparison of the performance of parsing algorithms and implemented systems: (i) the formal, complexity-theoretic analysis of how an algorithm behaves, typically focussing on worst-case time and space complexity bounds; and (ii) the empirical study of how properties of the parser and input (possibly including the grammar used) affect actual, observed run-time efficiency.

It has been shown (Maxwell and Kaplan, 1993; Carroll, 1994; van Noord, 1997) that the theoretical study of algorithms alone does not (yet) suffice to provide an accurate prediction about how a specific algorithm will perform in practice, when used in conjunction with a specific grammar (or type of grammar), and when applied to a particular domain and task. Therefore, empirical assessment of practical parser performance has become an established technique and continues to be the primary means of comparison among algorithms. At the same time, system competence (i.e. coverage and overgeneration with respect to a particular grammar and test set) cannot be decoupled from the evaluation of parser performance, because two algorithms can only be compared meaningfully when they really solve the same problem. This typically means that they either directly use the same grammar, or at least achieve demonstrably similar competence on the same test set.

In the next section, we briefly describe large-scale grammars and test suites that have been used in evaluations of parser efficiency. Section 3 discusses methods and computational tools that have been used in such evaluations, and Section 4 surveys research comparing the efficiency of different parsers or parsing strategies with large-scale grammars.

2 Grammars and Data

A number of large-scale, general-purpose grammars have been used in evaluations of parser efficiency. We describe their main characteristics briefly below.[1]

- The Alvey NL Tools (ANLT) contains a large, wide-coverage sentence grammar of English (Grover, Carroll, & Briscoe, 1993), written in a unification-based metagrammatical formalism resembling GPSG. The grammar expands out to an object grammar of 780 DCG-

[1] While it is the case that most current large-scale grammar-based parsing systems construct constituent structure representations that are capable of supporting semantic interpretation, the English Constraint Grammar (Karlsson, Voutilainen, Heikkilä, & Anttila, 1995) and the Link Grammar (Sleator & Temperley, 1993) systems are exceptions. Thus, since the motivations behind these grammars are different we do not consider them here.

like rules, each category containing on average around 30 nodes. Associated with the grammar is a test suite, originally written by the grammarian to monitor coverage during grammar development, containing around 1,400 (mostly grammatical) items.

- The SRI Core Language Engine (CLE) grammar (Alshawi, 1992) is also GPSG-inspired, but with different treatments of a number of central syntactic phenomena, such as subcategorisation and unbounded dependencies. The grammar contains of the order of 150 rules which map fairly directly into a DCG.
- The LinGO English grammar (Flickinger & Sag, 1998) is a broad-coverage HPSG developed at CSLI Stanford. The grammar contains roughly 8,000 types and 64 lexical and grammar rules, with an average feature structure size of around 300 nodes. Three main test sets have been used for parser evaluation with this grammar, the largest—containing 2,100 items—having been extracted from VerbMobil corpora of transcribed speech and balanced with respect to sentence length. (Comparable grammars of German and Japanese, again originally developed in VerbMobil, are shortly to be available).
- In the Xerox-led ParGram collaboration (Butt, King, Niño, & Segond, 1999), wide-coverage grammars of English, French, German and a number of other languages are being developed in parallel in the LFG framework, all of the grammars based on a common set of linguistic principles, with a commonly-agreed-upon set of grammatical features. Each grammar consists of an atomic-categoried phrase-structure backbone augmented with feature annotations.
- The trees in the Penn Treebank induce a large context-free grammar containing 15,000 rules. A recent comparison of context-free parsing strategies (Moore, 2000) has used this grammar, a second one derived from an ATIS treebank (with 4,600 productions), and a third (24,500 productions) produced by computing an atomic-categoried backbone from a unification-based phase structure grammar. Test sentences for these grammars were derived either from the associated corpora, or artificially, by using the grammar to stochastically generate random strings.
- The XTAG system grammar (XTAG, 1995) is a large-scale lexicalised tree adjoining grammar of English, developed by several researchers over the past ten years or so. The grammar contains of the order of 500 elementary tree schemata, organised into families; each lexeme is associated with a number of these families. Nodes in the tree schemata are augmented with feature structures so that information can be passed non-locally between elementary trees.

Test suites supplied with grammars have typically been written by the grammar developers themselves for the purpose of monitoring over- and under-generation as the grammar is changed. However, the test suites have also been found to be of some value for evaluating parser efficiency. A major drawback in this context, though, is that each test suite item usually only contains very limited ambiguity (easing the task of checking the resulting parses), and is relatively short (so that only one or two constructions are tested at a time). This is also the case for independently-developed test suites, such as the TSNLP suites for English, French and German (Oepen, Netter, & Klein, 1997). Therefore, in some parser evaluation work, new suites of longer sentences have had to be constructed manually or extracted specially from corpora.

Another important issue is the degree to which the grammars are available to the general NL processing research community. Those developed within companies are in general more difficult to obtain, although use for parser evaluation may be easier to negotiate than use within an actual application system, for instance.

3 Methods and Tools

Previous work on the assessment and comparison of large-scale parsers has mostly been concerned with evaluation of parser (or grammatical) coverage, and with correctness of the analyses produced. So, for example, coverage has been expressed in terms of lists of grammatical phenomena for which an analysis is provided; over- and under-generation as the percentage of grammatical or ungrammatical items from a given reference set that are or are not assigned

some sort of analysis; and degree of ambiguity of a grammar in terms of the 'parse base', the expected number of parses for a given input length (Carroll, Briscoe, & Sanfilippo, 1998). Work on quantifying parse correctness has used various measures of structural consistency with respect to constituent structure annotations of a corpus (e.g. exact match, crossing brackets, tree similarity, and others—see Black et al., 1991, Black, Garside, & Leech, 1993, Grisham, Macleod, & Sterling, 1992, and Briscoe & Carroll, 1993); recently, more general schemes have been advocated that deploy functor – argument (dependency) relations as an abstraction over different phrase structure analyses that a parser may assign (Lin, 1995; Lehmann et al., 1996; Carroll et al., 1998). The Penn Treebank and the SUSANNE corpus are well-established resources for the evaluation of parser accuracy.

In a sharp contrast, there is little existing methodology, let alone established reference data or software tools, for the evaluation and contrastive comparison of parser efficiency. Although most grammar development environments and large-scale parsing systems supply facilities to batch-process a test corpus and record the results produced by the system, these are typically restricted to processing a flat, unstructured input file (listing test sentences, one per line) and outputting a small number of processing results to a log file.[2] Additionally, no metrics exist that allow the comparison of parser efficiency across different grammars and sets of reference data. We therefore note a striking methodological and technological deficit in the area of precise and systematic assessment of grammar and parser behaviour.

Recently though, a new methodology, termed *competence & performance profiling* (Oepen & Flickinger, 1998; Oepen & Carroll, 2000), has been proposed that aims to fill this gap. Profiles are rich, precise, and structured snapshots of parser competence (coverage and correctness) and performance (efficiency), where the production, maintenance, and inspection of profiles is supported by a specialised software package called [incr tsdb()].[3] Profiles are stored in a relational database that serves as the basis for flexible report generation, visualisation, data analysis via basic descriptive statistics, and of course comparison to other profiles. The [incr tsdb()] package has so far been interfaced with some eight unification-based grammar development and/or parsing systems, and has served as the 'clearing house' in a multi-site collaborative effort on parser benchmarking (Flickinger, Oepen, Tsujii, & Uszkoreit, 2000), resulting in useful feedback to all participating groups.

4 Efficiency Comparisons

Many parsing algorithms suitable for NL grammars have been proposed over the years, their proponents often arguing that the number of computational steps are minimised with respect to alternative, competing algorithms. However, such arguments can only be made in the case of very closely related algorithms; qualitatively different computations can only reliably be compared empirically. So, for example, generalised LR parsing was put forward as an improvement over Earley-style parsing (Tomita, 1987), with a justification made by running implementations of the two types of parser on a medium-sized CF grammar with attribute-value augmentations. However, comparisons of this type have to be done with care. The coding of different strategies must use exactly equivalent techniques, and to be able to make any general claims, the grammar(s) used must be large enough to fully stress the algorithms. In particular, with grammars admitting less ambiguity, parse time is likely to increase more slowly with increasing input length, and also with smaller grammars rule application can be constrained tightly with relatively simple predictive techniques. In fact, a more recent evaluation (Moore, 2000) using a number of large-scale CF grammars has shown conclusively that generalised LR parsing is less efficient than certain left-corner parsing strate-

[2] Some (Meta-)Systems like PLEUK (Calder, 1993) and HDrug (van Noord & Bouma, 1997) that facilitate the exploration of multiple descriptive formalisms and processing strategies come with slightly more sophisticated benchmarking facilities and visualisation tools. However, they still largely operate on monolithic, unannotated input data sets, restrict accounting of system results to a small number of parameters (e.g. number of analyses, overall processing time, memory consumption, possibly the total number of chart edges), and only offer a limited, predefined choice of analysis techniques.

[3] See 'http://www.coli.uni-sb.de/itsdb/' for the (draft) [incr tsdb()] user manual, pronunciation guidelines, and instructions on obtaining and installing the package.

gies.

Moore and Dowding (1991) document a process of refining a unification-based (purely bottom-up) CKY parser (forming part of a speech understanding system) by incorporating top-down information to prevent it hypothesising constituents bottom-up that could not form part of a complete analysis, given the portions of rules already partially instantiated. An important step was reducing the spurious prediction of gaps by means of grammar transformations. The refinement process was guided throughout by empirical measurements of parser throughput on a test corpus.

Improvements in efficiency can be gained by specialising a general-purpose grammar to a particular corpus. Samuelsson and Rayner (1991) describe a machine learning technique that is applied to the CLE grammar to produce a version of the grammar that parses ATIS corpus sentences much faster than the original grammar. In general there are more rules in the specialised grammar than in the original, but they are more specific and can thus be applied more efficiently.

Maxwell and Kaplan (1993) investigate the interaction between parsing with the CF backbone component of a grammar and the resolution of functional constraints, using a precursor of the English ParGram grammar. A number of parsing strategies are evaluated, in combination with two different unifiers, on a small set of test sentences. There is a wide gap between the best and worst performing technique; the differences can be justified intuitively, but not with any formal analyses of computational complexity.

Carroll (1994) discusses the throughput of three quite distinct unification-based parsing algorithms running with the ANLT grammar. The main findings were that exponential parsing algorithm complexities with respect to grammar size have little impact on the performance of the parsers, since they all achieved relatively good throughput, and parse table sizes were also quite manageable. Increases in parse times with longer inputs were also fairly controlled, being roughly only quadratic. In another experiment, running the ANLT grammar with the CLE parser resulted in very poor performance, suggesting that the parallel development of the software and grammars had inadvertently caused them to become 'tuned' to one another.

van Noord (1997) presents an efficient implementation of head-corner parsing, as used in a prototype spoken language dialogue system. Memoisation and goal-weakening techniques are used to reduce parser space requirements; the head-corner parser also runs faster than implementations of left-corner, bottom-up and LR parsers in evaluations using a DCG of Dutch with speech recogniser word-graph input. A further set of evaluations use the ANLT grammar, allowing a tentative cross-system comparison with the ANLT parser to be made.

In work concerned with parsing with large-scale CF grammars, Moore (2000) investigates empirically the interactions between various types of grammar factoring and versions of the left-corner parsing algorithm that differ in the details of precisely how and in what order top-down filtering information is applied. Using three very different grammars, one of the parser/factoring combinations was found to be consistently and significantly better than the alternatives, despite being only minimally different from the other variants. This strategy was also shown to outperform several other major approaches to CF parsing.

Sarkar (2000) evaluates the efficiency of a chart-based head-corner parsing algorithm on a corpus of 2,250 Wall Street Journal sentences, using a large-scale grammar (containing 6,800 elementary tree schemata) extracted automatically from the Penn Treebank. For each sentence, parse times were found to correlate roughly exponentially with the number of lexicalised elementary trees selected; there was little correlation between sentence length and parse time.

Oepen and Carroll (2000) describe and argue for a strategy of *performance profiling* in the engineering of parsing systems for wide-coverage linguistic grammars. The aim is to characterise system performance at a very detailed technical level, but at the same time to abstract away from idiosyncracies of particular processing systems. Based on insights gained from detailed performance profiles of various parsing strategies with the LinGO English grammar, a novel 'hyper-active' parsing strategy is synthesised and evaluated.

A number of other empirically-driven research efforts into efficient parsing are described in the same journal special issue (Flickinger et al., 2000). These include grammar-writing techniques for improved parser efficiency, new efficient algorithms for feature structure operations, fast pre-unification filtering, and techniques for the extraction of CF grammars and abstract machine compilation for HPSGs.

5 Conclusions

Recent interest in large-scale, grammar-based parsing (in response to the demands of complex language-based application tasks) has led to renewed efforts to develop wide-coverage, general-purpose grammars, and associated research efforts into efficient parsing with these grammars. Some initial progress has been made towards precise empirical assessment of parser efficiency. However, more work is needed on methods, standard reference grammars and test data to facilitate improved comparability.

Acknowledgements

The first author is supported by a UK EPSRC Advanced Fellowship, and the second by the *Deutsche Forschungsgemeinschaft* Collaborative Research Division *Resource-Adaptive Cognitive Processes* (SFB 378) project B4 (PERFORM).

References

Alshawi, H. (Ed.). (1992). *The Core Language Engine.* Cambridge, MA: MIT Press.

Black, E., Abney, S., Flickenger, D. P., Gdaniec, C., Grishman, R., Harrison, P., Hindle, D., Ingria, R., Jelinek, F., Klavans, J., Liberman, M., Marcus, M., Roukos, S., Santorini, B., & Strzalkowski, T. (1991). A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the 4$^{\text{th}}$ DARPA speech and natural language workshop.* Pacific Grove, CA: Morgan Kaufmann.

Black, E., Garside, R., & Leech, G. (Eds.). (1993). *Statistically-driven computer grammars of English. The IBM – Lancaster approach.* Amsterdam, The Netherlands: Rodopi.

Briscoe, E., & Carroll, J. (1993). Generalised probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics, 19 (1)*, 25 – 60.

Butt, M., King, T. H., Niño, M.-E., & Segond, F. (1999). *A grammar writer's cookbook.* Stanford, CA: CSLI Publications.

Calder, J. (1993). Graphical interaction with constraint-based grammars. In *Proceedings of the 3rd Pacific Rim Conference on Computational Linguistics* (pp. 160 – 169). Vancouver, BC.

Carroll, J. (1994). Relating complexity to practical performance in parsing with wide-coverage unification grammars. In *Proceedings of the 32nd Meeting of the Association for Computational Linguistics* (pp. 287 – 294). Las Cruces, NM.

Carroll, J., Briscoe, E., & Sanfilippo, A. (1998). Parser evaluation: a survey and a new proposal. In *Proceedings of the 1st International Conference on Language Resources and Evaluation* (pp. 447 – 454). Granada, Spain.

Flickinger, D., Oepen, S., Tsujii, J., & Uszkoreit, H. (Eds.). (2000). *Journal of Natural Language Engineering. Special Issue on Efficient processing with HPSG: Methods, systems, evaluation.* Cambridge, UK: Cambridge University Press. (in press)

Flickinger, D. P., & Sag, I. A. (1998). Linguistic Grammars Online. A multi-purpose broad-coverage computational grammar of English. In *CSLI Bulletin 1999* (pp. 64 – 68). Stanford, CA: CSLI Publications.

Grisham, R., Macleod, C., & Sterling, J. (1992). Evaluating parsing strategies using standardized parse files. In *Proceedings of the 3rd ACL Conference on Applied Natural Language Processing* (pp. 156 – 161). Trento, Italy.

Group, T. X. R. (1995). *A lexicalized tree adjoining grammar for english* (Tech. Rep. No. IRCS Report 95-03). The Institute for Research in Cognitive Science, University of Pennsylvania.

Grover, C., Carroll, J., & Briscoe, E. (1993). *The Alvey Natural Language Tools grammar*

(4th release) (Technical Report No. 284). University of Cambridge: Computer Laboratory.

Karlsson, F., Voutilainen, A., Heikkilä, J., & Anttila, A. (1995). *Constraint grammar: a language-independent system for parsing unrestricted text.* Berlin, Germany: Mouton de Gruyter.

Lehmann, S., Oepen, S., Regnier-Prost, S., Netter, K., Lux, V., Klein, J., Falkedal, K., Fouvry, F., Estival, D., Dauphin, E., Compagnion, H., Baur, J., Balkan, L., & Arnold, D. (1996). TSNLP — Test Suites for Natural Language Processing. In *Proceedings of the 16th International Conference on Computational Linguistics* (pp. 711–716). Kopenhagen, Denmark.

Lin, D. (1995). A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 1420–1425). Montreal, Canada.

Maxwell III, J. T., & Kaplan, R. M. (1993). The interface between phrasal and functional constraints. *Computational Linguistics, 19 (4)*, 571–590.

Moore, R. (2000). Improved left-corner chart parsing for large context-free grammars. In *Proceedings of the 6th International Workshop on Parsing Technologies* (pp. 171–182). Trento, Italy.

Moore, R., & Dowding, J. (1991). Efficient bottom-up parsing. In *DARPA Speech and Natural Language Workshop* (pp. 200–203). Asilomar, CA.

van Noord, G. (1997). An efficient implementation of the head-corner parser. *Computational Linguistics, 23 (3)*, 425–456.

van Noord, G., & Bouma, G. (1997). Hdrug. A flexible and extendible development environment for natural language processing. In *Proceedings of the Workshop on Computational Environments for Grammar Development and Linguistic Engineering* (pp. 91–98). Madrid, Spain.

Oepen, S., & Carroll, J. (2000). Performance profiling for parser engineering. *Natural Language Engineering, 6 (1) (Special Issue on Efficient Processing with HPSG)*, 81–97.

Oepen, S., & Flickinger, D. P. (1998). Towards systematic grammar profiling. Test suite technology ten years after. *Journal of Computer Speech and Language, 12 (4) (Special Issue on Evaluation)*, 411–436.

Oepen, S., Netter, K., & Klein, J. (1997). TSNLP — Test Suites for Natural Language Processing. In J. Nerbonne (Ed.), *Linguistic Databases* (pp. 13–36). Stanford, CA: CSLI Publications.

Samuelsson, C., & Rayner, M. (1991). Quantitative evaluation of explanation-based learning as an optimization tool for a large-scale natural language system. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence* (pp. 609–615). Sydney, Australia.

Sarkar, A. (2000). Practical experiments in parsing using tree adjoining grammars. In *5th international workshop on tree adjoining grammars and related formalisms* (pp. 193–198). Paris France.

Sleator, D., & Temperley, D. (1993). Parsing English with a link grammar. In *Proceedings of the 3rd International Workshop on Parsing Technologies* (pp. 277–292). Tilburg, The Netherlands.

Tomita, M. (1987). An efficient augmented-context-free parsing algorithm. *Computational Linguistics, 13 (1)*, 31–46.