Special Issue: Finite State Methods in Natural Language Processing

Lauri Karttunen, Kimmo Koskenniemi, Gertjan van Noord

Finite state methods have been in common use in various areas of natural language processing (NLP) for many years. A series of specialized workshops in this area illustrates this. In 1996, András Kornai organized a very successful workshop entitled *Extended Finite State Models of Language*. One of the results of that workshop was a special issue of Natural Language Engineering (Volume 2, Number 4). In 1998, Kemal Oflazer organized a workshop called *Finite State Methods in Natural Language Processing*. A selection of submissions for this workshop were later included in a special issue of *Computational Linguistics* (Volume 26, Number 1). Inspired by these events, Lauri Karttunen, Kimmo Koskenniemi and Gertjan van Noord took the initiative for a workshop on finite state methods in NLP in Helsinki, as part of the European Summer School in Language, Logic and Information. As a related special event, the 20th anniversary of two-level morphology was celebrated. The appreciation of these events led us to believe that once again it should be possible, with some additional submissions, to compose an interesting special issue of this journal.

The popularity of finite state methods can be explained by the attractive properties of finite state automata. Finite state automata are well-understood, and inherently *efficient* models of simple languages. In addition, there is an efficient minimization algorithm which constructs for every (deterministic) finite state automaton an equivalent, unique, *minimal* finite state automaton. The article by Bruce Watson and Jan Daciuk describes an alternative minimization algorithm. Unlike the standard algorithm, their algorithm is *incremental*: the algorithm successively constructs smaller (equivalent) automata. If for some reason (for very big automata, or in situations where very little CPU-time is available) the algorithm cannot be completed, then the latest intermediate result is useful as it is a smaller, equivalent, alternative for the input automaton.

Perhaps the most important attractive property of finite state automata is that they can be combined in various interesting ways, with the guarantee that the result again is a finite state automaton. This property is perhaps most clearly exploited in regular expressions. Regular expressions can be seen as recipes which indicate how highly complicated finite state machines can be built from very simple ones. The type of regular expressions used in modern NLP applications has evolved dramatically from the regular expressions found in standard Computer Science textbooks. This development was initiated in Kaplan and Kay (1994). One of their innovations was the use of regular expressions for the description of finite state *transducers*. The regular expression calculus was extended in various ways in Karttunen (1995, 1996, 1998), Mohri and Sproat (1996), Karttunen et al. (1996), Kempe and Karttunen (1996) and Gerdemann and van Noord (1999). In these papers, various high level regular expression operators are defined (such as contexted replacement operators). The availability of more and more abstract operators make the regular expression notation more and more attractive. The regular expression notation has become a succinct, declarative notation for regular languages and regular relations which abstracts away from many details of imlementation of the actual finite state machines. The remaining five contributions in this special issue clearly exploit the regular expression calculus in this way.

The various replacement operators referred to above sometimes have subtle differences in semantics, and the precise definition of these operators are typically fairly complicated. Nathan Vaillette provides simpler definitions of such operators by exploiting the connection between regular sets and monadic second-order logic. He incorporates monadic second-order logic formulas into the regular expression calculus, and illustrates that this provides the machinery for definitions of replacement operators with straightforward proofs of correctness.

In Karttunen (1998) a regular expression operator *lenient composition* was introduced. The operator, motivated for modelling optimality-theoretic analyses of phonology, is used to filter out transductions in a lenient way: remove undesirable outputs for a given input, unless this input has no alternative outputs. Kemal Oflazer integrates the lenient composition operator in two-level morphology, and illustrates the benefit of lenient two-level morphology with examples from Basque and Turkish.

Gerhard Jaeger investigates the complexity of a specific variant of optimality theory known as *bidirectional optimality theory*. He introduces a related regular expression operator called *generalized lenient composition*. This operator generalizes Karttunen's lenient composition operator as well as the optimality operator introduced in Gerdemann and van Noord (2000). The generalized lenient composition operator is useful in modeling an important class of (classical, uni-directional) OT constraints that may be violated multiple times. Jaeger shows that in bi-directional OT, constraints that can be violated multiple times remain problematic. In fact, bi-directional OT is shown to be more powerful than finite-state, even if all of the ingredients (GEN and the various constraints) are finite-state.

Gosse Bouma studies finite state models for hyphenation. He describes how a finite state hyphenation method can be constructed on the basis of syllable structure. Transformation-based learning can be exploited to improve this hyphenator. An alternative finite state hyphenation system can be constructed by compiling the hyphenation patters used in TEX to finite state transducers. In both cases, the required operations are specified by means of regular expressions.

In a practically motivated article, Aldezabal and collegues describe a finite state syntactic grammar of Basque. The purpose of this grammar is the extraction of verb subcategorization information from large text corpora. The finite state grammar is specified by a variety of regular expressions which perform clause recognition and disambiguation and which implement various agreement constraints.

The tradition of workshops on finite state methods in NLP will be extended by another workshop on Finite State Methods in NLP, organized by Eric Laporte and Ken Beesley in the context of the EACL in Budapest, in April 2003.

We would like to express our thanks to the regular members of the editorial board that contributed to this enterprise as well as to the reviewers for the workshop and the additional editorial board members for this special issue (including Ken Beesley, Gosse Bouma, Sascha Brawer, Ted Briscoe, John Carroll, Jan Daciuk, Jason Eisner, Tamas Gáal, Dale Gerdemann, Gregory Grefenstette, Martin Kay, André Kempe, Ronald Kaplan, Lucia Helena Machado Rino, Franck Thollard, Bruce Watson, Annie Zaenen), and journal editor John Tait.

References

Gerdemann, Dale and Gertjan van Noord. 1999. Transducers from rewrite rules with backreferences. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, Bergen Norway.

Gerdemann, Dale and Gertjan van Noord. 2000. Approximation and exactness in finite state optimality theory. In *Coling Workshop Finite State Phonology*, Luxembourg.

Kaplan, Ronald and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–379.

Karttunen, Lauri. 1995. The replace operator. In *33th Annual Meeting of the Association for Computational Linguistics*, M.I.T. Cambridge Mass.

Karttunen, Lauri. 1996. Directed replacement. In *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz.

Karttunen, Lauri. 1998. The proper treatment of optimality theory in computational phonology. In *Finite-state Methods in Natural Language Processing*, pages 1–12, Ankara.

Karttunen, Lauri, Jean-Pierre Chanod, Gregory Grefenstette, and Anne Schiller. 1996. Regular expressions for language engineering. *Natural Language Engineering*, 2(4):305–238. http://www.rxrc.xerox.com/research/mltt/fst/articles/jnle-97/rele.html.

Kempe, André and Lauri Karttunen. 1996. Parallel replacement in the finite-state calculus. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, Copenhagen, Denmark.

Mohri, Mehryar and Richard Sproat. 1996. An efficient compiler for weighted rewrite rules. In *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz.