Extended Finite State Models of Language

András Kornai (editor)
(BBN Technologies)

Cambridge University Press (Studies in natural language processing), 1999, xii+278 pp and CD-ROM; hardbound, ISBN 0-521-63198-X, $59.95

Reviewed by
Ed Kaiser
Oregon Graduate Institute

In this volume, András Kornai has put together a collection of articles that strongly argue his contention that finite-state approaches to natural language processing (NLP) are now part of the mainstream, both theoretically and computationally. The papers included were first presented at a 1996 workshop, held in Budapest as part of the European Conference on Artificial Intelligence (ECAI '96), and have been chosen for inclusion in this volume to complement previous works on finite-state approaches. Articles referring to the Xerox regular expression calculus and the AT&T Bell Labs system of weighted finite-state transducers that appeared in Roche and Schabes's 1997 *Finite State Language Processing* are followed up by new articles in this book. In addition this volume includes a CD-ROM, which, although it does not contain the Xerox and AT&T Bell Labs toolkits, does have source code and executables for several of the systems described in the book, including an older version of Bruce Watson's FIRE Lite toolkit for constructing and minimizing finite automata.

The attraction of finite-state approaches to NLP is their speed and efficiency. The question is their adequacy—just how powerful a formal system is needed to describe natural language? Consider the following sentence:

A doctor (whom a doctor)^m (hired)^n hired another nurse.

As Gazdar and Mellish point out in their textbook, from which this example is taken, this is a legal English sentence only if m and n are equal (Gazdar and Mellish 1989). Strings of the form a^nb^n are not regular expressions and therefore cannot be represented by finite-state machines. They require a context-free grammar definition and a machine with an unbounded memory, such as the stack of a pushdown automaton. Natural language constructions can require even more complex forms such as $a^nb^nc^n$, which in turn must be defined using indexed grammars and parsed with a machine that has the equivalent of multiple stacks of memory. However "suppose that we had access to hardware that would handle FSTNs [finite-state transition networks] ... ultrafast and observed that in actual occurrences of a^nb^n constructions, the value of n never exceeded 3; then we might decide to compile our RTN [recursive transition network] ... descriptions down into FSTNs subject to an $n = 3$ upper bound (such a compilation is possible for any given finite upper bound on the value of n)" (Gazdar and Mellish 1989). Kornai points out that such finite upper bounds are indeed what is observed in parsing natural languages.

Kornai argues in his introduction that while the surge of thought and development surrounding transformational models in the early 1960s threatened to remove all credibility from finite-state approaches to NLP, the "extraordinary impact" of Thompson's (1968) grep family of Unix tools lead to the pervasive belief that if you wanted to "do something with text you needed to build finite automata." In this regard, Kornai's own chapter on vectorized finite-state automata describes an extremely efficient pattern-matching engine, around which the NewsMonitor system is built. This system extracts relational information, such as "who is where" or "who bought what", from issues of the *Wall Street Journal* (source code and sample data are included on the CD-ROM).

Shortly after the publication of *The Sound Pattern of English* (Chomsky and Halle 1968), Kornai points out, "Johnson (1970) demonstrated that the context-sensitive machinery of *SPE* ... [could] be replaced by a much simpler one, based on finite-state transducers (FSTs); the same conclusion was reached independently by Kaplan and Kay, whose work remained an underground classic until it was finally published in Kaplan and Kay (1994)." These works inspired Koskenniemi's two-level system, and the Xerox rule compiler (Dalrymple et al. 1987). Both are now dominant tools in the fields of computational phonology and morphology, as exemplified by Tateno et al. (Chapter 6), "The Japanese lexical transducer based on stem-suffix style forms" and Kim and Jang (Chapter 7), "Acquiring rules for reducing morphological ambiguity from POS tagged corpus in Korean." The latter includes an algorithm for automatically inferring regular grammar rules for morphological relations directly from part-of-speech tagged corpora.

Although finite-state approaches to NLP were attempted as early as 1958, Kornai comments that finite-state syntax "did not really come in from the cold until the nineties." Chapter 2 of this work, "A parser from antiquity: An early application of finite state transducers to natural language parsing," by Joshi and Hopely, describes that 1958 parser. In a short commentary on that article, Lauri Karttunen points out that "many of the currently popular methods for robust parsing are already present, fully articulated:

- multiword tokens (*because of, in front of*);
- tagging words by ambiguity classes (*cool VA,* [i.e., (V)erb (A)djective class]);
- rule-based disambiguation;
- syntactic markup by finite-state transduction;
- depth-first strategy with backtracking and pushdown store for the analysis of recursive structures;
- default selection of one structure among alternative analyses with option for later revision."

Contemporary systems that incorporate these features in finite-state approaches to parsing and modeling syntax are described in the following papers in the book:

- Chanod and Tapanainen, Chapter 8, "Finite state based reductionist parsing for French";
- Grefenstette, Chapter 9, "Light parsing as finite state filtering";
- Kornai, Chapter 10, "Vectorized finite state automata";
- Roche, Chapter 11, "Finite state transducers: Parsing free and frozen sentences".

The papers by Vilares et al. (Chapter 12), "Text and speech translation by means of subsequential transducers," and Ejerhed (Chapter 13), "Finite state segmentation of discourse into clauses," move the application of finite-state techniques into exciting new areas of machine translation and discourse segmentation.

In Chapter 11, Roche expands on the part-of-speech-tagging and parsing articles of Roche and Schabes (1997). Specifically, he details a finite-state method for syntactically parsing light verbs. Such verbs have complements that allow only a certain degree of variability:

(1) John makes concessions to his friend.

(2) John makes a right turn.

(3) *John makes a right turn to his friend.

In this example the predicative noun *concessions* is the real head of the sentence, governing the number and nature of arguments. The light verb *makes* is in a support role. Roche's paper shows that such constructions, for which "rewriting mechanisms such as context free parsing are at best unnatural," can be processed by finite-state transducer parsing. Kornai points out that such light-verb constructions were seen in the tradition of Chomsky (1970) as "core cases of transformational grammar." Roche's article argues persuasively that finite-state models are not just "an efficient but somewhat inaccurate tool," as might be imagined from a transformational grammar perspective, "but rather one of the best formalisms at hand to represent accurately complex linguistic phenomena."

Kornai suggests also that another "important way in which mainstream syntax is impacted by finite-state techniques can be called 'finite-state to the rescue'. The paper by Schulz and Mikolajewski [Chapter 14, "Between finite state and Prolog: Constraint-based automata for efficient recognition of phrases"] ... describes how constraint-based grammars can be speeded up by finite-state methods, and the paper by Srinivas [Chapter 15, "Explanation-based learning and finite state transducers: Applications to parsing lexicalized tree adjoining grammars"] ... shows how corpus-based acquisition of LTAGs is facilitated by finite-state techniques."

Speech recognition is dominated by statistical techniques, in particular HMMs (which are finite-state mechanisms) and statistical *n*-gram language models. In regards to syntax parsing, Kornai points out that "an important step in bringing rule-based and statistical work closer is the framework of weighted finite-state transducers developed at AT&T Bell Labs, represented in this volume by the Tzoukermann and Radev paper" (Chapter 16, "Use of weighted finite state transducers in part of speech tagging"). This article describes a process of classifying words into "genotypes" through the use of weighted, negative-constraint transducers. All members of a particular genotype share the same set of possible part-of-speech tags. After tagging, genotype *n*-gram models are built. Tzoukermann and Radev state that their system "correctly disambiguates 96% of words in unrestricted texts."

Since the main algorithms and model-building techniques associated with the Xerox and AT&T Bell Labs approaches to finite-state NLP are well described elsewhere, this book examines ways of extending the scope of finite-state machinery. For example, the "more complex formal systems discussed by Csuhaj-Varjú [Chapter 17, "Colonies: A multi-agent approach to language generation"] ..., Nederhof and Bertsch [Chapter 18, "An innovative finite state concept for recognition and parsing of context free languages"] ..., and Ristad [Chapter 19, "Hidden Markov models with finite state supervision"] ... are likely to provide a fertile ground for further experimentation with extended finite state models of language."

Nederhof and Bertsch's article defines a new subclass of context-free languages that combines deterministic languages (also known as the $LR(k)$ languages) within the sequential framework of a finite-state automaton. The result is a class of languages that can be recognized and parsed in linear time, which also allows for *deterministic* center-self-embedding (i.e., rule descriptions that define strings of the form $a^n b^n$ for unbounded values of n). This approach is in some respects similar to lexicalized tree adjoining grammars (Srinivas, Chapter 15), which also have a regular level (i.e., *initial* trees) that contain a context-free lower-level (i.e., *auxiliary* trees—for center-self-embedding). The notion of combining regular grammars with context-free or restricted context-free grammars (like the $LR(k)$ grammars) to create new subclasses of grammars and languages that lend themselves to treatment by finite-state approaches is an exciting area of research that is well represented in this book.

Overall, I believe that this book will be a valued addition to the library of anyone interested in emerging finite-state approaches to NLP.

References

Chomsky, Noam. 1970. Remarks on nominalization. In R. Jacobs and P. Rosenbaum, editors, *Readings in English Transformational Grammar*. Blaisdell, Waltham, MA, pages 184–221.

Chomsky, Noam and Morris Halle. 1968. *The Sound Pattern of English*. Harper and Row, New York.

Dalrymple, Mary, Ronald M. Kaplan, Lauri Karttunen, Kimmo Koskenniemi, Sami Shaio, and Michael Wescoat. 1987. *Tools for morphological analysis*. Center for the Study of Language and Information Report CSLI-87-108. CSLI, Stanford, CA.

Gazdar, Gerald and Chris Mellish. 1989. *Natural Language Processing in Prolog*. Addison Wesley.

Johnson, C. Douglas. 1970. *Formal Aspects of Phonological Representation* Ph.D. thesis, University of California at Berkeley.

Kaplan, Ronald M. and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 21(3):331–378.

Roche, Emmanuel and Yves Schabes, editors. 1997. *Finite-State Language Processing*. The MIT Press, Cambridge, MA.

Thompson, Ken. 1968. Regular expression search algorithm. *Communications of the ACM*, 11(6):419–422.

Ed Kaiser is interested in integrating structural language models directly into the speech recognition process. He has developed a robust, finite-state parser (inspired by CMU's Phoenix system) as a first step towards that integration, and published two papers on the compilation techniques involved. Kaiser's address is Computer Science and Engineering Department, Oregon Graduate Institute, P.O. Box 91000, Portland, OR, 97291-1000; e-mail: kaiser@cse.ogi.edu