FINITE-STATE LANGUAGES AND GRAMMARS.[*] Let V be a finite vocabulary of basic symbols (e.g. phonemes, letters, morphemes, words, etc.; anything that is capable of being concatenated). A string in V is any finite (possibly empty) concatenation of members of V. The class of finite-state languages in V is defined recursively in (1) through (3) (Kleene 1956, Chomsky & Miller 1958).

(1) Every finite set of strings in V is a finite-state language in V.

(2) If X and Y are finite-state languages in V, then so is Z, where $Z = \{z \mid z \in X$ or $z \in Y\}$, or $Z = \{x\, y \mid x \in X$ and $y \in Y\}$.

(3) Nothing else is a finite-state language in V.

For example, let $V = \{a, b\}$, and let x^n represent the string consisting of n repetitions of the substring x. Then $L_1 = \{a^m\ b^n \mid m, n > 0\}$ is a finite-state language in V, but $L_2 = \{a^m\ b^n \mid m > 0$ and $m = n\}$ is not; L_2 is a strictly context-free language in V. [*See* Context-free Languages and Grammars.]

The members of the class of finite-state languages in V are generated by the members of the class of finite-state grammars $<N, V, S, R>$, where N is a finite set of nonterminal symbols, S Í N is a set of start symbols, and R is a finite set of rules of the form (4) or (5), where $P, Q \in N$, and x is a string in V.

(4) $P \rightarrow x\ Q$

(5) $P \rightarrow x$

For example, L_1 is generated by the finite-state grammar $<\{A, B\}, \{a, b\}, \{A\}, \{A \rightarrow a\ A, A \rightarrow a\ B, B \rightarrow b\ B, B \rightarrow b\}\}>$, whereas there is no finite-state grammar that generates L_2. Finite-state languages are accepted by the class of finite automata, and constitute the smallest infinite class of languages in the Chomsky hierarchy of formal languages. [*See* Automata Theory.]

Chomsky (1956) and in many subsequent publications contended that English (and presumably every other natural language) is not a finite-state language over its vocabulary, on the grounds that its grammar must contain rules that are not reducible to schema (4) or (5). For example, to account for the syntactic dependency between the words *if* and *then* in a sentence like *if it rains then it pours*, English grammar must contain a rule like *S → if S then S*. If so, and assuming it contains no other rules regulating the co-occurrence of *if* and *then*, then English also contains the sentences *if if it rains then it pours then it pours* and *if if if it rains then it pours then it pours then it pours*, but not **if it rains then it pours then it pours* nor **if if it rains then it pours*. That is, English contains every sentence of the form $\{(if)^m$ *it rains* $(then\ it\ pours)^n \mid m = n\}$, but no sentence of the form $\{(if)^m$ *it rains* $(then\ it\ pours)^n \mid m \neq n\}$. Hence English is not a finite-state language.

To counter the obvious objection that speakers of English do not distinguish between the case of $m = n$ and $m \neq n$ when m and n are large (they generally reject all such sentences), Chomsky (1965) introduced the competence-performance distinction, arguing that an English speaker's internalized grammar generates all the sentences of the form $(if)^m$ *it rains* $(then\ it\ pours)^n$ in which $m = n$ and none in which $m \neq n$, and that his or her general inability to process such sentences for which m and n are large is a cognitive limitation that lies outside of the internalized

[*] Prepublication version of D. Terence Langendoen (2003) Finite state languages. *Oxford International Encyclopedia of Linguistics*, 2nd ed., ed. by William J. Frawley, vol. 3, pp. 26-28. Oxford University Press.

grammar. For the most part, linguists accepted Chomsky's argument, and their interest in the theory of finite-state languages and grammars, which was never great in the first place, languished.

However, certain aspects of a natural language, at least, are completely analyzable using finite-state methods, in particular its phonotactics (the determination of legitimate sequences of sounds in natural language; Johnson 1970, Kaplan & Kay 1994), and its morphology excluding compound formation (Koskenniemi 1983). Despite the fact that in principle morphological rules can give rise to sets of words that cannot be generated by a finite-state grammar, no such system for natural languages has ever been discovered (Langendoen 1981). Consequently in the areas of speech (and orthographic) analysis and of morphological analysis, finite-state methods have become standard.

The use of finite-state methods in syntax and semantics was somewhat slower to develop, in large measure due to the belief that these aspects of linguistic structure are not fully analyzable in those terms. However, partial syntactic and semantic analyses have been carried out using finite-state methods beginning with the Transformation and Discourse Analysis Project at the University of Pennsylvania in the late 1950s (Joshi & Hopely 1999). The first serious proposal that finite-state methods are fully adequate for syntactic analysis was made by Krauwer & des Tombe (1981), and a sophisticated (augmented) finite-state grammar that handles discontinuous elements was developed by Blank (1989). For recent surveys of what is now a vast and rapidly growing field, see Roche & Schabes (1997), Kornai (1999), Nederhof (2000), and Beesley & Karttunen (forthcoming).

D. TERENCE LANGENDOEN

BIBLIOGRAPHY

BEESLEY, K. R. & L. KARTTUNEN. forthcoming. *Finite state morphology: Xerox tools and techniques*. Cambridge, U.K.: Cambridge University Press.

BLANK, G. D. 1989. A finite and real time processor for natural language. *Communications of the ACM* 32.1174–1189.

CHOMSKY, N. 1956. Three models for the description of language. *IRE Transactions on Information Theory* IT-2, pp. 113-124. Reprinted in Luce, Bush & Galanter 1965, pp. 105–124.

CHOMSKY, N. 1965. *Aspects of the theory of syntax*. Cambridge, Mass.: MIT Press.

CHOMSKY, N. & G. A. MILLER. 1958. Finite state languages. *Information & Control* 1.91–112. Reprinted in Luce, Bush & Galanter 1965, pp. 156–171.

JOHNSON, C. D. 1970. Formal aspects of phonological representation. Ph.D. dissertation, University of California, Berkeley.

JOSHI, A. & P. HOPELY. 1999. A parser from antiquity: An early application of finite state transducers to natural language parsing. In Kornai 1999, pp. 6–15.

KAPLAN, R. M. & M. KAY. 1994. Regular models of phonological rule systems. *Computational Linguistics* 20.331–378.

KLEENE, S. C. 1956. Representation of events in nerve nets and finite automata. In *Automata studies*, edited by C. E. Shannon & J. McCarthy, pp. 3–42. Princeton, N.J.: Princeton University Press.

KORNAI, A., ed. 1999. *Extended finite state models of language*. Cambridge, U.K.: Cambridge University Press.

KOSKENNIEMI, K. K. 1983. *Two-level morphology: A general computational model for word-form recognition and production*. Ph.D. dissertation, University of Helsinki, Finland.

KRAUWER, S. & L. DES TOMBE. 1981. Transducers and grammars as theories of language. *Theoretical Linguistics* 8.173–202.

LANGENDOEN, D. T. 1981. The generative capacity of word-formation components. *Linguistic Inquiry* 12.320–322.

LUCE, R. D., R. R. BUSH, & E. GALANTER, eds. 1965. *Readings in mathematical psychology,* vol. 2. New York: Wiley.

NEDERHOF, M.-J.. 2000. Practical experiments with regular approximation of context-free languages. *Computational Linguistics*, 26.17-44. http://odur.let.rug.nl/~markjan/publications/2000a.pdf

PRINCE, A. S., & P. SMOLENSKY. 1997. Optimality: From neural networks to universal grammar. *Science* 275.1604–1610.

ROCHE, E. & Y. SCHABES, eds. 1997. *Finite-state devices for natural language processing*. Cambridge, Mass.: MIT Press.