ECONOMY OF DESCRIPTION BY AUTOMATA, GRAMMARS, AND FORMAL SYSTEMS†

A. R. Meyer and M. J. Fischer
Massachusetts Institute of Technology
Cambridge, Massachusetts

1. Introduction

We propose in this paper an approach to the classification of automata, grammars, and related formal systems which has until now received little explicit attention. Our aim is to analyse the size of systems for specifying mathematical objects as opposed to the power of such systems.

As an illustration of the kind of investigations we propose, consider the following assortment of formal systems for specifying regular sets:

Deterministic and Nondeterministic:
1) finite automata[13]
2) two-way finite automata[14]
3) one pebble finite automata[5]
4) linear-time one-tape off-line Turing machines[7]
5) regular expressions[9]
6) extended regular expressions (allowing set complementation),
7) right linear grammars.

Each of these systems is equally powerful in the sense that the class of sets of words specifiable by any one of them is precisely the regular sets. Nevertheless, specification of regular sets may be much more economical in one system than another. A familiar example of this point is the contrast between deterministic and nondeterministic finite automata. It is well known that nondeterministic automata can be exponentially more succinct. In the next section we show that the gain in economy achievable by nondeterministic compared to deterministic finite automata can be exactly determined.

Analysis of the size of systems provides a new outlook for comparing and explaining the utility of different systems. Context-free grammars are a good example. The widespread use of context-free grammars for describing programming languages seems almost paradoxical when one surveys the large collection of theoretical results (for example, inherent ambiguity and slow recognition times [6]) which suggest that context-free languages are poor models for programming languages. The power of context-free grammars is actually a disadvantage in specifying efficient parser-compilers. A good deal of recent effort has been spent in finding methods for transforming grammars of more or less general appearance into equivalent special form grammars for efficient parsing [1,10]. Nevertheless the reason for using general context-free grammars even for specifying fairly special kinds of languages is obvious. The general grammars are usually much shorter, easier to understand, and easier to write than restricted form grammars. Our goal is to prove mathematically this observation that general grammars are better at describing simple languages, and moreover to quantify the advantage that general grammars provide. As a preliminary step in this program we describe in section three some results about the size of general context-free grammars, deterministic grammars, and finite automata for describing finite sets and regular sets.

By the size of an automaton or grammar we mean, roughly speaking, the number of symbols needed to specify it. Thus the number of productions in a grammar is an unrealistic size measure, since a single production might be enormously large. We define instead the size of a grammar to be the sum of the lengths of the productions. Similarly the number of states in an automaton may be a poor measure, and we use instead the number of arcs in the state diagram.

This research is still in its preliminary state. The proofs below are incomplete, and we expect that all of our results can be significantly sharpened with further effort. Two theorems are labelled "strong conjectures" because we have not yet had time to carry out what we expect to be a routine but tedious formal verification. Our techniques are familiar in automata theory with the emphasis diverted to obtain results about size instead of power of formal systems.

2. Size of Finite Automata

We begin by comparing the size of deterministic and nondeterministic finite automata. The standard subset construction shows for any n-state nondeterministic automaton, there exists an equivalent deterministic automaton with at most 2^n states. The following example shows that this bound is optimal.

Proposition 1. For every $n > 0$, there is a regular set $R_n \subset \{0,1\}^*$ such that the reduced finite automaton accepting R_n has exactly 2^n states (and size 2^{n+1}), but there is an n-state nondeterministic finite automaton of size 3n-2 accepting R_n. Moreover, the reversal of R_n is recognizable by a 2n-state deterministic machine (of size 4n).

Proof. The nondeterministic machine for R_n has states $\{0,1,\ldots,n-1\}$. Input 1 takes state i to state (i+1) mod n. Input 0 takes state i to itself and to state 0 for $i \neq 0$. State 0 is both start and final state. □

(As far as we know the optimality of the subset construction for converting from nondeterministic to deterministic finite automata has not appeared in the literature. Rabin[12] mentions this as an open problem. Our example is a simplification of an example in an unpublished report by G. Ott. We subsequently noted that this example for n = 4 appears as an exercise in Hennie[8].

†Work reported herein was conducted at Project MAC and at the Artificial Intelligence Laboratory, M.I.T. research programs supported in part by the Advanced Research Projects Agency of the Department of Defense under the Office of Naval Research Contracts Numbers N00014-70-A-0362-0001 and -0002.

An even simpler example suggested by Paterson is close to optimal: let $R'_n \subset \{0,1\}^*$ be the set of strings whose n^{th} from the last digit is 1. Then R'_n is recognized by an n+1 state nondeterministic machine, the reduced deterministic machine has 2^n states, and the reduced machine for the reversal of R'_n has n+2 states. Moreover, $R'_n \cap \{0,1\}^n\{0,1,\lambda,\}^n$ is a finite event with similar properties.

As an immediate consequence of the observation about reversal we note that two-way finite automata are exponentially more succinct than finite automata for two (or three depending on conventions about end-of-tape markers) symbol regular events. Moreover, an n+4 state two-way automaton for R'_n can accept and reject strings of length ℓ in only ℓ+2n steps, so that in this case there is essentially no sacrifice in processing time for the sake of program size economy. This is an improvement over a similar example recently published by Barnes[3].

Shepherdson[14] shows that any n-state two-way finite automaton is equivalent to a finite automaton with at most $(n+2)^{n+1}$ states. The next result shows that an economy of roughly this kind can actually be achieved.

Proposition 2. For every $n > 1$, there is a regular set $F_n \subset \{0,1,2\}^*$ which can be recognized by a two-way finite automaton with at most 5n+5 states, but the reduced finite automaton accepting F_n has at least n^n states. Moreover F_n is a finite set.

Proof. Let $F_n = \{0^{i_1}10^{i_2}1\cdots 10^{i_n}2^k0^{i_k} \mid 1 \le k \le n \text{ and } 1 \le i_j \le n \text{ for } j = 1,\ldots,n\}$.

Meyer's proof that two-way deterministic one-pebble automata recognize only regular sets[5] actually reveals that any such n-state automaton is equivalent to a finite automaton with roughly n^{n^n} states. That at least a doubly exponential improvement is possible follows from

Proposition 3. For each $n > 0$ there is a regular event $P_n \subset \{0,1,2\}^*$ which is recognizable by a deterministic one pebble automaton with at most 3n+5 states. The reduced finite automaton accepting P_n has at least 2^{2^n} states. Moreover, P_n is finite.

Proof. $P_n = \{x_1 2 x_2 2 \ldots 2 x_k 2 2 x_i \mid x_j \in \{0,1\}^n$ for $1 \le j \le k$, $i \le k$, and regarded as binary integer $x_j < x_{j+1}$ for $1 \le j < k\}$. □

Such improvements are not possible for singleton sets.

Proposition 4. For any language consisting of a single word of length n, the reduced finite automaton accepting it has n+2 states and every nondeterministic finite automaton accepting it has at least n+1 states.

Strong Conjecture. For languages consisting of a single word, the improvement of each of the systems (1)-(7) mentioned in the introduction over deterministic finite automata is at most exponential. For singleton languages over a one symbol alphabet, there is essentially no improvement.

3. Context-free Grammars and Pushdown Automata

One can convert between nondeterministic pushdown machines and context-free grammars, and between deterministic pushdown store machines and LR(1) grammars with a change in size bounded by a linear factor. Hence the results below about size of machines carry over to grammars directly.

Proposition 5. If a finite set is recognized by a deterministic (nondeterministic) pushdown automaton with n states and s pushdown store symbols, then it can be recognized by a deterministic (nondeterministic) finite automaton with $0(s^{(sn)^2})$.

Proof. Using an argument similar to the proof of the xuwvy pumping lemma for context free grammars [2], one can show that the length of the store cannot be greater than $(sn)^2$ in accepting computations by the pushdown automaton, if the automaton accepts only finitely many inputs. A finite automaton with $s^{(sn)^2}$ states can therefore "remember" the symbols on the store and simulate the pushdown automaton. □

Thus the gain in economy of pushdown automata over finite automata for describing finite sets is at most roughly exponential. Simple examples such as the length n palindromes show that this much economy can be approximately achieved. Rather suprisingly the situation for infinite regular sets is drastically changed.

First we note the deep results of Stearns that if a deterministic pushdown automaton with n states and s pushdown symbols accepts a regular set, then the regular set is acceptable by a deterministic finite automaton with at most s^{n^n} states[15]. Whether this bound is achievable remains open, but we can show

Proposition 6. For each $n > 0$ there is an infinite regular set I_n whose reduced finite automaton contains at least 2^{2^n} states, but I_n can be recognized by a deterministic pushdown store machine of size $0(n^3)$.

Proof. I_n consists of words in $\{0,1,a_1,\ldots,a_n\}^*\{0,1\}^n$ accepted by a deterministic pushdown store machine which operates as follows:

1) Copy the input onto the store until input a_1 is encountered. If a_1 does not occur, reject the input.
2) Set i = 2.
3) If the next input is zero, pop the store until the first occurrence of a_i. If the next input is a one, pop the store to the second occurrence of a_i. If any other input is encountered, or the occurrences of a_i are not found, reject the input.
4) Increment i by one.
5) If $i \le n$, repeat step (3).
6) If the digit on top of the store is 1 and there are no more input symbols, accept the input. Otherwise reject the input.

The deterministic pushdown automaton described above has roughly n states, n pushdown symbols and n inputs, and so is of size propostional to n^3.

Consider the set A_n of reverse Polish prefix expressions of depth n involving binary operators $a_1,\ldots,a_n$ over $\{0,1\}$ generated by the grammar

$$S \to S_2 S_2 a_1$$

$$S_k \to S_{k+1} S_{k+1} a_k,\ k = 2,\ldots,n$$

$$S_{n+1} \to 0 \mid 1.$$

Each element of A_n corresponds in an obvious way to a binary tree of depth n with leaf labels zero and one. A particular leaf can be described by a binary word of length n which describes the path from the root to the leaf, where a zero or one in the i^{th} digit of the binary word indicates whether the path goes left or right at the i^{th} level.

Let $B_n = \{xy \mid x \in A_n,\ y \in \{0,1\}^n$, and the leaf of x determined by y is labelled with 1$\}$. It is easy to see that a finite automaton accepting B_n requires at least 2^{2^n} states, since each word $x \in A_n$ must lead to a different state in the course of accepting B_n.

But I_n is defined so that $B_n = I_n \cap \{xy \mid x \in A_n\}$, and it follows that a finite automaton for I_n must also have 2^{2^n} states because A_n is recognizable by a finite automaton with approximately 2^n states. We omit the proof that I_n is in fact regular. □

Thus Proposition 5 and 6 together show that the improvement in size over finite automata provided by deterministic pushdown store machines is still greater on infinite regular sets than on finite sets.

Comparison of the size of finite automata and general context-free grammars for infinite regular sets reveals a qualitatively new phenomenon. The gain in economy can be arbitrary.

Proposition 7. For any recursive function f and for arbitrarily large integers n, there is a context-free grammar of size n describing a regular (in fact co-finite) set whose reduced finite automaton has at **least** f(n) states.

Proof. Given any Turing machine T with states $q_1,\ldots,q_k$ and symbols $x_1,\ldots,x_\ell$ which halts started on blank tape started in state q_1, let h(T) be the sequence of successive instanteneous descriptions (id's) of the computation of T on blank tape with successive id's separated by \$ and alternately reversed. That is h(T) is of the form

$$(id)_1\ \$(id)_2^R\ \$(id)_3\ \$(id)_4^R\ \$\cdots\$(id)_m\ \$$$

where superscript "R" indicates reversal.

It is not hard to show that $\{q_1,\ldots,q_k,x_1,\ldots,x_\ell,\$\}^* - \{h(T)\}$ is a context-free language definable by a grammar of size approximately the same as T.* On the other hand the minimum size finite automaton recognizing this language must have as many states as the length of h(T). Since there is no recursive function f bounding the length of h(T) and T, the proposition follows. □

*Cf. Hartmanis[16].

Finite automata can be regarded as a special case of pushdown automaton, so finite automaton descriptions of sets are never smaller than minimal descriptions by pusdhown automata. However, pushdown automata are not always more succinct than all the other systems for regular sets mentioned in the introduction. In fact, deterministic pushdown automata and one pebble finite automata are exponentially more succinct than each other on different classes of finite sets.

Strong Conjecture. The finite event P_n of Proposition 3 (which can be recognized by a one pebble automaton with 3n+5 states) cannot be recognized by any deterministic pushdown store of size less than $0(2^n)$. On the other hand the singleton language $\{a^{2^n}\}$ can be recognized by a deterministic pushdown store automaton with size 0(n), but cannot be recognized by any one pebble automaton of size less than 2^n.

4. Further Results

It is natural to consider context-sensitive grammars and Turing machines next in our discussion. We merely note here that context-sensitive grammars may be arbitrarily more succinct (in the sense of Proposition 7) than context-free grammars, and that Turing machines may be arbitrarily more succinct than context-sensitive grammars for describing finite sets. The proofs of these observations use arguments of recursive function theory rather than automata theory[4, 11].

Acknowledgement

We are indebted to Michael S. Paterson for his help and interest in this paper.

References

1. Aho, A. and Ullman, J., The Sensuous Compiler, Prentice Hall, Englewood Cliffs, N.J., to appear.
2. Bar-Hillel, Y., Perles, M., and Shamir, E., On Formal Properties of Simple Phrase Structure Grammars, Z. Phonetik, Sprach. Kommunikationsforsch., Vol. 14, (1961) 143-172; also appears as Chap. 9 in Y. Bar-Hillel, Language and Information, Addison-Wesley, Reading, Mass., 1964.
3. Barnes, B., A Two-Way Automaton with Fewer States than Any Equivalent One-Way Automaton, IEEE Trans. on Comp. C-20, 4 (April 1971), 474-475.
4. Blum, M., On the Size of Machines, Inf. and Control 11 (1967), 257-265.
5. Blum, M. and Hewitt, C., Automata on a 2-Dimensional Tape, Proceedings of the Eighth IEEE Symposium on Switching and Automata Theory, 1967, 155-160.
6. Ginsburg, S., The Mathematical Theory of Context-Free Languages, McGraw-Hill, N.Y. 1966.
7. Hennie, F., One-tape Off-line Turing Machine Computations, Inf. and Control 8 (1965), 553-578.
8. Hennie, F., Finite-State Models for Logical Machines, John Wiley and Sons, Inc., New York, 1968.
9. Kleene, S., Representation of Events in Nerve Nets and Finite Automata, Automata Studies, Annals of Mathematics Studies, No. 34, Princeton University Press, Princeton, N.J., 1956, 3-41.

10. Lewis, P. and Rosenkrantz, D., An ALGOL Compiler, Designed Using Automata Theory, presented at the Polytechnic Institute of Brooklyn Symposium on Computers and Automata, April 1971. Proceedings to appear as volume XXI of the MRI Symposia Series, Fall 1971.

11. Meyer, A., The Size of Programs in Restricted Programming Languages, to appear (1971).

12. Rabin, M., Mathematical Theory of Automata, Proc. Sympos. Appl. Math., Vol 19, Amer. Math. Soc., Providence, R.I., 1967, 153-175.

13. Rabin, M. and Scott, D., Finite Automata and their Decision Problems, IBM J. Res. Dev. 3 (1959), 114-125.

14. Shepardson, J., The Reduction of Two-way Automata to One-way Automata, IBM J. Res. Dev. 3 (1959), 198-200.

15. Stearns, R., A Regularity Test for Pushdown Machines, Inf. and Control 11 (1967), 323-340.

16. Hartmanis, J., Context-Free Languages and Turing Machine Computations, Proc. Sympos. Appl. Math., Vol. 19, Amer. Math. Soc., Providence, R.I., 1967, 42-51.