INFORMATION AND CONTROL 25, 123–133 (1974)

The Equivalence Problem for Deterministic Finite-Turn Pushdown Automata*

LESLIE G. VALIANT

Department of Computer Science, Carnegie-Mellon University, Schenley Park, Pittsburgh, Pennsylvania 15213

It is proved that there is an effective procedure for deciding whether two deterministic finite-turn pushdown automata are equivalent.

INTRODUCTION

A deterministic pushdown automaton (dpda) is described as finite-turn if there is a bound on the number of times the direction of the stack movement can change in the set of all derivations from the starting configuration. The purpose of this paper is to show that there exists a procedure for deciding whether two such finite-turn machines recognize the same language. By virtue of a direct correspondence between a restricted class of one-turn dpda and deterministic two-tape acceptors (Valiant, 1973), our proof also provides a solution to the equivalance problem for the latter, alternative to that of Bird (1973). Since some of the ideas we introduce are not related exclusively to the finite-turn property, or even to pushdown machines, it is hoped that our methods can be adapted for constructing equivalance tests for other classes of deterministic automata.

Our main technique can be regarded as a generalization in several directions of one introduced by Rosenkrantz and Stearns (1970). They consider a class of pushdown automata for which a natural valuation can be placed on each stack segment and deduce that, for any input word, two equivalent machines in that class must have closely related stack movements. They show how, under such circumstances, for any two machines a single pushdown automaton can be constructed to simulate them both and used to decide their equivalence. What we shall show is that, even for a class with no such stack valuation known and in which two equivalent machines can have totally dissimilar

* This research was supported by the Science Research Council of the United Kingdom.

stack movements, pushdown automata, now nondeterministic, can be found to perform the required simulations.

Definitions

The class of finite-turn machines can be described as the intersection of the class of deterministic pushdown automata (Ginsburg and Greibach, 1966) with the class of nondeterministic finite-turn pushdown automata (Ginsburg and Spanier, 1966). For convenience we shall define and deal with a normal form for these machines. We observe, however, that it is decidable whether a pushdown automaton has the finite-turn property and that, if it has and is in addition deterministic, then it can be transformed in a number of easy steps to an equivalent machine in this normal form (Valiant, 1973).

A dpda M is a one-tape one-way deterministic acceptor with a pushdown stack and a finite state control for storage. It can be specified by a sextuple $(Q, \Gamma, \Sigma, \Delta, c_s, F)$ where Q, Γ, and Σ are respectively finite sets of *states* $\{s, \ldots\}$, *stack symbols* $\{A, \ldots\}$, and *input symbols* $\{a, \ldots\}$, and Δ, c_s, and F are as defined below. Typically, we shall denote words from Γ^* and Σ^* by ω and α, respectively, and the length of ω by $| \omega |$. A *configuration* c is a pair (s, ω) from $Q \times \Gamma^*$. A *mode*, designated either a *reading mode* or an ϵ *mode* is a pair $Q \times (\Gamma \cup \{\Omega\})$, where Ω is a special empty stack symbol. Δ is the set of *transitions*, each of the form

$$(s, A) \xrightarrow{\pi} (s', \omega),$$

where $\pi \in \Sigma \cup \{\epsilon\}$, such that,

(i) if (s, A) is a reading mode, then for each $a \in \Sigma$ it has a unique transition with $\pi = a$ but none with $\pi = \epsilon$ and,

(ii) if (s, A) is an ϵ mode, then it has just one transition and in this $\pi = \epsilon$ and ω is null.

This machine makes a *move*

$$(s, \omega A) \xrightarrow{\pi} (s', \omega\omega')$$

if and only if there is some transition

$$(s, A) \xrightarrow{\pi} (s', \omega').$$

If $\pi \in \Sigma$, then this symbol is considered to have been read.

A *derivation* is a sequence of such moves through successive configurations and is said to read the word α if α is the concatenation of the symbols read by the constituent moves.

The set F of *accepting modes* is a set of reading modes. A word α is *accepted* from the configuration c if and only if for some c' with mode (i.e., state, top stack symbol) belonging to F, there is a derivation from c to c' that reads α. Two configurations c_1 and c_2 are *equivalent*, $c_1 \equiv c_2$, if the same set of strings is accepted from both. The language accepted by M is the set of words accepted from c_s, its *starting configuration*, and is denoted by $L(M)$.

The finite-turn property is guaranteed by defining Q to be the disjoint union of sets $Q_0, Q_1, \ldots, Q_n$, where for all transitions $(s, A) \to^{\pi} (s', \omega)$, with $s \in Q_i$, if i is even then

$$|\omega| \geqslant 1 \Rightarrow s' \in Q_i \text{ and}$$
$$|\omega| < 1 \Rightarrow s' \in Q_{i+1},$$

and if i is odd then

$$|\omega| > 1 \Rightarrow s' \in Q_{i+1}, \text{ and}$$
$$|\omega| \leqslant 1 \Rightarrow s' \in Q_i .$$

A configuration in state $s \in Q_i$ is said to be in an *upstroke* if i is even and in a *downstroke* if i is odd. A move from a state in Q_i to one in Q_{i+1} constitutes a *turn.*

Parallel Stacking

The argument for decidability that we shall use is embodied in the following lemma. This depends only on the well-known facts that both emptiness and membership are decidable for languages recognized by (nondeterministic) pushdown acceptors.

Lemma. *If there is an effective procedure which, given any M_1 and M_2 belonging to a class T of pushdown acceptors, can enumerate a family $P(M_1, M_2)$ of pushdown automata with the properties that*

(i) $L(M_1) = L(M_2) \Rightarrow \exists M' \in P(M_1, M_2)$ *such that $L(M')$ is empty, and*

(ii) $L(M_1) \neq L(M_2) \Rightarrow \forall M' \in P(M_1, M_2)$ $L(M')$ *is nonempty,*

then equivalence is decidable in T.

Proof. The conditions of the lemma guarantee that equivalence is partially decidable. For, by enumerating the family $P(M_1, M_2)$ and testing each

member for emptiness, we shall be sure to verify equivalence, if there is equivalence, when we reach the machine that accepts nothing.

However, inequivalence is also partially decidable. The set of all words over the input alphabet can be enumerated in some lexicographic manner, and each one can be tested for acceptance by exactly one of M_1 and M_2.

Running the two partial procedures concurrently constitutes the required decision procedure. ▮

To generate each member of the family $P(M_1, M_2)$ for a pair of deterministic finite-turn machines, we use what we call the parallel stacking construction. For ease of description it is convenient to first create a single machine M from the disjoint union of the specifications of M_1 and M_2 (the choice of c_s in M being irrelevant) so that we need only discuss configurations from a single machine. We shall now describe the basic form of a typical machine M' in the class $P(M_1, M_2)$ in terms of the machine M.

A configuration of M' has a stack which can be regarded as having a left track and a right track. The stack is segmented into lengths of bounded size by special symbols, called *ceilings*, occupying both tracks. The finite state control of M' is able to manipulate directly all of the *top segment* (i.e., the part of the stack above the topmost ceiling). Each track in the top segment is associated with a state of M.

In each segment below the top one, both tracks contain nonnull words over the stack alphabet of M. Into each ceiling there is encoded the following information about the previous history of the computation:

(1) a *quadruple* (s_1, A_1, s_2, A_2), which states that at the time the ceiling was created the left and right tracks had modes (s_1, A_1) and (s_2, A_2) respectively, and

(2) an *indicator* $(X, Y) \in \{L, R\}^2$, specifying that the left track above the ceiling is to be associated with the X track below and the right track above is to be associated with the Y track below.

Each configuration of M' is to be interpreted as corresponding to two configurations of M in the obvious way; i.e., the M configuration can be recreated by taking each track in the top segment of M' and concatenating it with the appropriate words in the segments below, as specified successively by the indicators in the ceilings, and by adopting the associated state. Note that if $X = Y$ in one of the ceilings, then in each segment below it only one of the tracks will contribute to these M configurations.

The basic operations of M' are to mimic simultaneously for any input the transitions appropriate to both of the simulated configurations of M. In order

to be able to do this while at the same time maintaining an upper bound on the length of the segments that can arise, the machine M' can also do, on occasions depending on the contents of the top stack segment, one of the following operations without reading inputs:

(a) If one of the tracks in the top segment is empty and the other contains a word from a certain set of "short" words, then the ceiling below these is removed and the tracks immediately below and above this ceiling are fused, in the manner specified by the indicator, into one segment.

(b) If both tracks have more than one symbol, then a ceiling is placed to be just below the top symbol of each track. The indicator (L, R) and the quadruple corresponding to the modes of the tracks are encoded into this ceiling.

(c) If one track has just one symbol while the other contains a word from some specified set of "long" words and is associated with a reading mode, then M' has the following *nondeterministic* choice of moves: A *replacement* stack word, with the same mode as the large track, is introduced to replace one or the other of the tracks of the top segment. The simulation is then to be allowed to continue to compare the newly introduced configuration with whichever of the old ones is left. The replacement word is "short" in the above sense but of length greater than one, and it is uniquely specified by the M' configuration.

Examples of each of these three kinds of moves are illustrated in Fig. 1. We observe that once a ceiling has been created, although its quadruple cannot be modified, its indicator can be changed by moves of type (a) or (c). We also note that the simulation of a pair of ϵ derivations causes no problems if we ensure that M' pursues them both at the same time until either both terminate or just one terminates and a replacement occurs.

We define the starting configuration of M' to be that containing the starting configurations of M_1 and M_2 in the two tracks. The conditions for acceptance by M' are that either (i) both tracks of the top segment are in reading modes, exactly one of which is an accepting mode of M, or (ii) the top segment is about to exceed the allowed segment bound.

We shall prove that for any finite-turn machines M_1 and M_2, the parallel stacking simulators for M_1 and M_2 with arbitrarily guessed segment bound and replacement operations form a class satisfying the conditions of the Lemma. It is evident that this class is effectively enumerable and that each of its elements can be transformed by an appropriate encoding to a pushdown automaton of the standard kind.

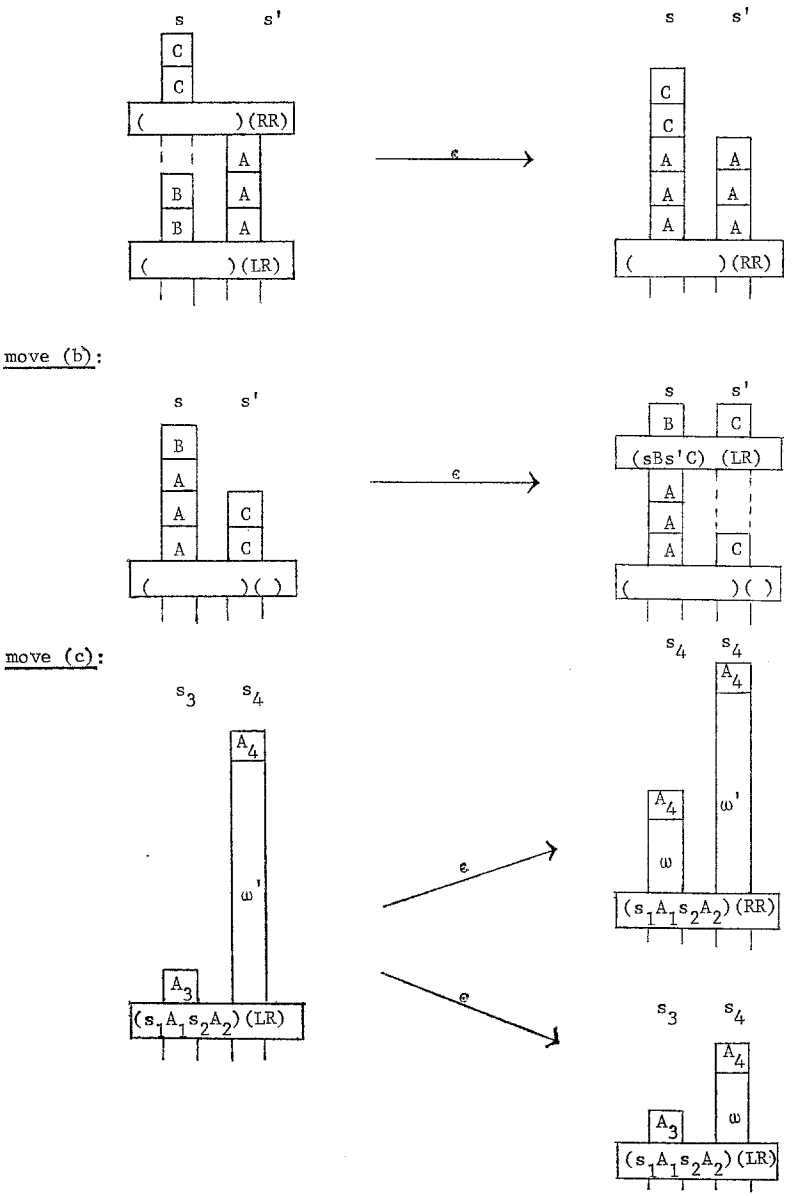
move (a):
s
s'
C
C
( )(RR)
B
B
A
A
A
( )(LR)
ε
s
s'
C
C
A
A
A
A
A
( )(RR)
move (b):
s
s'
B
A
A
A
C
C
( )( )
ε
s
s'
B
C
(sBs'C) (LR)
A
A
A
C
( )( )
move (c):
s_3
s_4
A_4
ω'
A_3
$(s_1A_1s_2A_2)$ (LR)
ε
ε
s_4
s_4
A_4
A_4
ω
ω'
$(s_1A_1s_2A_2)$ (RR)
s_3
s_4
A_4
A_3
ω
$(s_1A_1s_2A_2)$ (LR)

FIGURE 1

Simulation of Equivalent Machines

We show that for any pair of equivalent configurations of M there exists a simulating pushdown automaton M' of the kind described, which, when started with these configurations in its tracks, will accept no input word.

Clearly, any pair of configurations of M each reached via the same input string from one of a pair of equivalent configurations will be equivalent. The crucial part of the argument is to show that, when the tracks get too much out of step in M', we can by making appropriate replacements get them back within a finite bound while still maintaining the equivalence of the pairs of configurations being simulated.

We now define the function Rep over

$$(Q \times \Gamma)^4 \times \{1, 2\}$$

to determine these replacements.

Definition. $\mathrm{Rep}(s_1, A_1, s_2, A_2, s_3, A_3, s_4, A_4, \theta) = \omega$, where ω, if defined, is a shortest nonnull stack word with the property that, for any $\omega_1, \omega_2 \in \Gamma^*$,

$$(s_1, \omega_1 A_1) \equiv (s_2, \omega_2 A_2) \Rightarrow (s_3, \omega_\theta A_3) \equiv (s_4, \omega_2 \omega A_4).$$

In M', if the top ceiling contains (s_1, A_1, s_2, A_2) and (L, R) and if the left track above contains (s_3, A_3) and the right track contains some $(s_4, \omega' A_4)$, then we make a nondeterministic replacement by $(s_4, \omega A_4)$, where

$$\mathrm{Rep}(s_1, A_1, s_2, A_2, s_3, A_3, s_4, A_4, 1) = \omega,$$

provided ω is shorter than ω' and that (s_4, A_4) is a reading mode. This is illustrated in Fig. 1(c). If the indicator were (L, L), then the value of Rep for $\theta = 2$ would have been appropriate while, if the top tracks were interchanged, we would treat them analogously by symmetry.

We show that under the circumstances specified for a replacement the function Rep must be defined by observing that ω' is itself a suitable candidate for its value. By construction the equivalence of the pair of configurations being simulated will be preserved in any derivation of M'. As the computational history subsequent to the creation of the current topmost ceiling is dependent only on that ceiling and could be repeated for any other stack with the same ceiling on top, it follows that ω' must satisfy the conditions required of a value of Rep (except possibly minimality).

We do not need to construct this function or to determine the arguments for which it is defined. It suffices to observe that a correct guess of this

function exists and that, since its domain is finite, the words in its range are bounded in length by some number ρ.

A finite state control in M' can carry out these correctly guessed replacements provided that it can always manipulate the whole of the top segment. It therefore remains to prove that there is an upper bound to the lengths of the segments that can arise in the computations of M'. It is here that the finite-turn property is required.

We first consider the various eventualities that may arise:

(1) If both tracks are steadily increasing, then new ceilings will be frequently added and no segment will become large.

(2) If one track is increasing steadily in an upstroke while the other stays at a stationary height, then after a stack growth bounded by a constant μ in the one a valid replacement must be possible, for there must be two instants in any suitably long such derivation at which the mode of the growing track and the configuration of the stationary one both repeat. A replacement word can then be obtained by cutting out a segment from the growing track. Thus, for however long this track is trying to grow, the effect on M' will be to make nondeterministic moves repeatedly so as to keep the segment lengths bounded.

(3) If one track is decreasing, then situations can arise in which the length of the segment created depends in a bounded way on the size of the previously existing segments and is not bounded a priori by ρ. There are just two ways in which a track can grow "out of bounds" without being arrested immediately by a nondeterministic replacement:

(i) If one track in the top segment is emptied but the other is not quite long enough to have qualified for a replacement, then the ceiling below will be removed by an (a) move. The resulting fusion may create a top segment that is suddenly longer by y than before, where y is the bound on the segments previously occurring in the configuration.

(ii) A track may grow larger and larger without a replacement being possible if the other track does not become a singleton in the meantime. This second track can be assumed to be steadily decreasing since, as in (2), stationary periods do not contribute to growth. Thus a decreasing track initially of size y may cause to arise a segment of length no more than μy.

The crucial observation is that the gains in length that can be achieved by either of methods (3i) or (3ii) can be exploited to achieve further gains only after further turns have been made by the simulated configurations.

More precisely, we define the *order* of an M' configuration to be the pair (i,j) iff the states of the tracks are $s \in Q_i$ and $s' \in Q_j$, respectively. The effect of a replacement on a configuration of M' of order (i,j) is to produce one of order (i,j), (i,i), or (j,j), while an ordinary simulating move would lead to one of order (i,j), $(i+1,j)$, $(i,j+1)$, or $(i+1,j+1)$. We have to ensure that the simulating machine cannot enter any "loop" that would cause the segments to increase in size indefinitely. Considerations (1)–(3) above show that this behavior certainly cannot occur for any derivation through configurations of just one order. Thus, we have to consider derivations from configurations of M' of a given order to others of the same order via ones of different orders. Clearly, these must involve replacements that substitute for the track with the singleton. Since by definition the modes of the tracks are the same after any such replacement, either both tracks will be in an upstroke or both will be in a downstroke. In the former case, it is obvious that, before the two tracks can again exhibit different behavior, turns must be made by both tracks. In the latter case the height of the top segment can be exploited by method (3ii) for further growth after a turn has been made by just one of the tracks. However, the new gains will only be achieved in the track that is now in the upstroke and cannot be exploited for further growth until the other track has turned or been replaced.

It follows that the times when successive gains can be made in the size of segments in excess of the bound ρ can be regarded as occurring at periods during which M' has configurations whose orders form a monotonic increasing sequence under the ordering defined by

$$(i,j) > (i',j') \Leftrightarrow i,j > i', \text{ or } i,j > j', \text{ or } i > i' \text{ and } j \geqslant j', \text{ or } i \geqslant i' \text{ and } j > j'.$$

If M can only make n turns, then such a monotonic sequence of pairs has no more than $(n+1)^2$ elements. Hence, we conclude that there is a finite bound on the size of segments that may occur in any computation of M'.

Consequently, a finite state control is sufficient to specify for each top segment that may arise whether moves of type (a), (b), or (c) are appropriate and what form these should take and to carry out the normal simulation otherwise.

This concludes the proof that condition (i) of the Lemma is satisfied.

Simulation of Inequivalent Machines

We have now to show that, if a parallel stacking machine M' is constructed with inequivalent configurations initially in its tracks, then whatever replace-

ment function and segment bound M' incorporates it will always accept some input word, even if it never attempts to exceed the specified bound.

Consider the simulation by M' of a string α distinguishing the inequivalent configurations. Either the two are simulated directly to their different conclusions, causing M' to accept α by definition, or else a replacement must occur after a part of α has been read. In the latter case, one of the new pairs of configurations must clearly be distinguished by the remainder of α. Also, since any sequence of replacements made without any input being read must cause the stack to decrease, it must be finite. It follows that α will be eventually accepted via some finite derivation of M'.

Conclusion

Since the previous sections have established that for any finite-turn automata M_1 and M_2 the class $P(M_1, M_2)$ generated by parallel stacking satisfies the conditions of the Lemma, we can deduce our main result.

Theorem. *The equivalence problem for deterministic finite-turn pushdown automata is decidable.* ∎

It can be verified that there exist pairs of equivalent dpda's without the finite-turn property, for which the particular stacking construction described cannot keep the segment lengths bounded. However, we have not found any such equivalent pair for which a simulating machine of a closely related nature does not exist. To keep the segments bounded, we may require only to store more information in the ceilings and to take a more macroscopic view of the stack movements. It therefore appears plausible that our construction may be adapted to obtain procedures for other classes of dpda's once certain deeper properties of these machines have been established. One particular such property is suggested by our conjecture that a related construction, called alternate stacking (Valiant, 1973), resolves the equivalence problem for the class of dpda's that have no ϵ moves and only empty-stack acceptance (Harrison and Havel, 1972).

Acknowledgment

The author wishes to thank Michael Paterson for many illuminating discussions on the subject of decision problems.

Received: October 16, 1973

References

BIRD, M. R. (1973), The equivalence problem for deterministic two-tape automata, *JCSS* **7**, 218–236.

GINSBURG, S. AND GREIBACH, S. A. (1966), Deterministic context-free languages, *Inf. and Control* **9**, 620–648.

GINSBURG, S. AND SPANIER, E. H. (1966), Finite-turn pushdown automata, *SIAM J. on Control* **4**, 423–434.

HARRISON, M. A. AND HAVEL, I. M. (1972), Real time strict deterministic languages, *SIAM J. on Computing*.

ROSENKRANTZ, D. J. AND STEARNS, R. E. (1970), Properties of deterministic topdown grammars, *Inf. and Control* **17**, 226–255.

VALIANT, L. G. (1973), Decision procedures for families of deterministic pushdown automata, University of Warwick Computer Centre, Report No. 7.