A Simpler Construction for Showing the Intrinsically Exponential Complexity of the Circularity Problem for Attribute Grammars

MEHDI JAZAYERI

University of North Carolina, Chapel Hill, North Carolina

ABSTRACT. The recognition problem for alternating Turing machines is reduced to the circularity problem for attribute grammars, and thus an inherently exponential lower bound for the complexity of the circularity problem is derived Although the result is already known, the use of alternation allows a simpler construction

KEY WORDS AND PHRASES: alternating Turing machines, attribute grammars, circularity problem, computational complexity, exponential time

CR CATEGORIES: 4.12, 5.25

1. *Introduction*

The circularity problem for attribute grammars was shown to be of inherently exponential complexity in [3]. The proof was based on the recognition problem for writing pushdown acceptors. By basing our proof on alternating Turing machines instead we are able to achieve a simpler construction here. We assume the reader is familiar with the problem of circularity. For a brief review, see [3, Secs. 1 and 2].

2. *The Result and Background*

The following theorem is the result of our construction.

THEOREM. *The lower bound on the complexity of the circularity problem for attribute grammars is* $2^{cn/\log n}$, *where n is the size of the grammar description. That is, there is a constant* $c > 0$ *such that any correct algorithm must run for* $2^{cn/\log n}$ *steps for infinitely many n's.*

The theorem will be established by the usual reduction method. We reduce a known exponential problem to the circularity problem. In [3] the known problem was the recognition problem for writing pushdown acceptors. The handling of the stack complicated the construction considerably.

The present proof is based on alternating Turing machines (ATMs) introduced by Chandra and Stockmeyer [1] (see also [2]). In an alternating Turing machine some

This work was supported in part by the National Science Foundation under Grant MCS 77-03729 and in part by a Fulbright Grant at the University of Helsinki, Helsinki, Finland, in the summer of 1979.

Author's present address: Synapse Computer Corporation, 801 Buckeye Court, Milpitas, CA 95035.

states are designated as *universal* and others as *existential.* In contrast, all states in a nondeterministic Turing machine are existential. The difference between an existential state and a universal state is that for an existential state to lead to an acceptance of the input it is sufficient that *only one* of the possible next moves leads to an accepting configuration; for a universal state, *all* of the possible next moves must lead to an accepting configuration.

Formally, an alternating Turing machine is a 6-tuple

$$M = (Q, \Sigma, \delta, q_0, F, U),$$

where

Q = set of states,
q_0 = initial state,
F = set of accepting states,
Σ = input and tape alphabet,
$\delta = (Q, \Sigma) \times (Q, \Sigma, D)$ is the next move relation, where $D = \{left, right\}$,
U = set of universal states,
$Q - U$ = set of existential states.

A machine move is represented in the form

$$\delta(q, x) = \{(q_1, w, D_1), (q_2, y, D_2), \ldots, (q_n, z, D_n)\},$$
$$\text{where} \quad q, q_1, q_2, \ldots, q_n \in Q; \quad D_1, D_2, \ldots, D_n \in \{left, right\}; \quad x, w, y, z \in \Sigma.$$

The meaning of the move is: In state q, scanning symbol x, take any one of the following actions if q is an existential state; take all of them (simultaneously) if q is a universal state:

(1) Rewrite x as w, move the tape head one position in the direction of D_1, and change the state to q_1.
(2) Rewrite x as y, move the tape head one position in the direction D_2, and change the state to q_2.
⋮
(n) Rewrite x as z, move the tape head one position in the direction D_n, and change the state to q_n.

A *configuration* of machine M consists of the state, head position, and contents of the tape. A configuration is existential (respectively, universal, accepting, initial) if the state of the machine in that configuration is an existential (respectively, universal, accepting, initial) state. A configuration is *accepting* if either

(i) it contains an accepting state;
(ii) it is existential and at least one of its δ-successors is accepting; or
(iii) it is universal and all of its δ-successors are accepting.

An input[1] $\propto$ is *accepted* by machine M if the initial configuration, consisting of the initial state q_0, scanning the leftmost symbol of $\propto$, is accepting.

ATMs have many interesting properties. The one we will use in this paper is that the time complexity of the acceptance problem for ATMs as defined above (i.e., linear space-bounded) is exponential in terms of deterministic Turing machines [1, Th. 3.2; 2, Th. 3].

[1] As usual, the input is assumed to be delineated by endmarkers No move is possible if the head runs beyond the endmarkers at either end.

3. *The Construction*

To show the exponential lower bound on the complexity of the circularity problem for attribute grammars, we reduce the recognition problem for ATM to the circularity problem. In particular, given a machine M and an input $\propto$, we show how to construct a grammar $G(M, \propto)$ such that M accepts $\propto$ iff $G(M, \propto)$ is circular. Throughout the construction we assume that $\Sigma = \{a, b\}$.

The grammar $G(M, \propto)$ has one nonterminal for each state q of the machine M. It also has a starting nonterminal S. Each nonterminal $\langle q \rangle$ has $3(2n - 1)$ attributes where $n = |\propto|$. Each move rule and each final state of M gives rise to one production in G. The starting production of G is derived from the starting configuration of M.

The attributes of the nonterminal symbol are used to represent the current tape configuration. Let C be the set of these current attributes for a nonterminal symbol. C is partitioned into $2n - 1$ cells, $C(1), \ldots, C(j), \ldots, C(2n - 1)$. Each cell contains three attributes, which we name $C(j, a)$, $C(j, b)$, and $C(j, *)$. In a circular dependency network the fact that $C(j, *)$ participates in a circuit will imply that cell $C(j)$ represents a current tape square. It will turn out that in this case exactly one of $C(j, a)$ and $C(j, b)$ also participates in the circuit, so that the tape square will contain either an a or a b. At a nonterminal node the active cells in a circuit will always form a block of n contiguous cells $C(p), \ldots, C(p + n - 1)$. Such a block must always include cell $C(n)$ because there are only $2n - 1$ cells in all. Cell $C(n)$ will thus always represent the position of the head relative to the beginning of the tape; a "move left" instruction will cause a "shift right" in the active cell block, so that $C(p + 1)$ becomes the first active cell and $C(n)$ is one space closer to the first active cell. (For an idea of the general form of a circuit corresponding to an accepting computation, see Figure 1.)

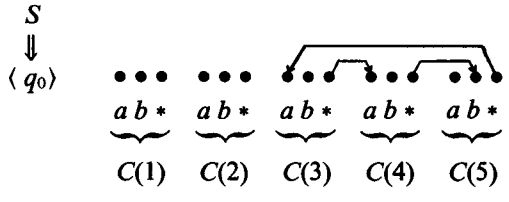
In each cell $C(j)$ the inherited attributes are $C(j, a)$ and $C(j, b)$; $C(j, *)$ is synthesized.[2] In a production $X^0 \rightarrow X^1 X^2$ let C^0 be the attributes associated with X^0, C^1 with X^1, and C^2 with X^2.

We are now ready to associate grammar productions with move rules of the ATM. A simple example is used to illustrate the construction.

(1) Let q_0 be the initial state of M. In the initial configuration of M the tape head is at square 1 of the tape, and the contents are $\propto = x_1, \ldots, x_n$. The following production and semantic rules "record" the initial configuration in the grammar:

$$\begin{aligned} &\langle S \rangle \rightarrow \langle q_0 \rangle \\ &\quad C(n, x_1) := C(2n - 1, *) \\ &\quad C(j, x_j) := C(j - 1, *) \quad j = n + 1, n + 2, \ldots, 2n - 1. \end{aligned}$$

Pictorially, the effect can be seen clearly. Assume that $\propto = aab$. Then the attribute connections would be



[2] In terms of a parse tree, an inherited attribute is a function of attributes of parent or sibling nodes; a synthesized attribute is a function of attributes of children nodes.

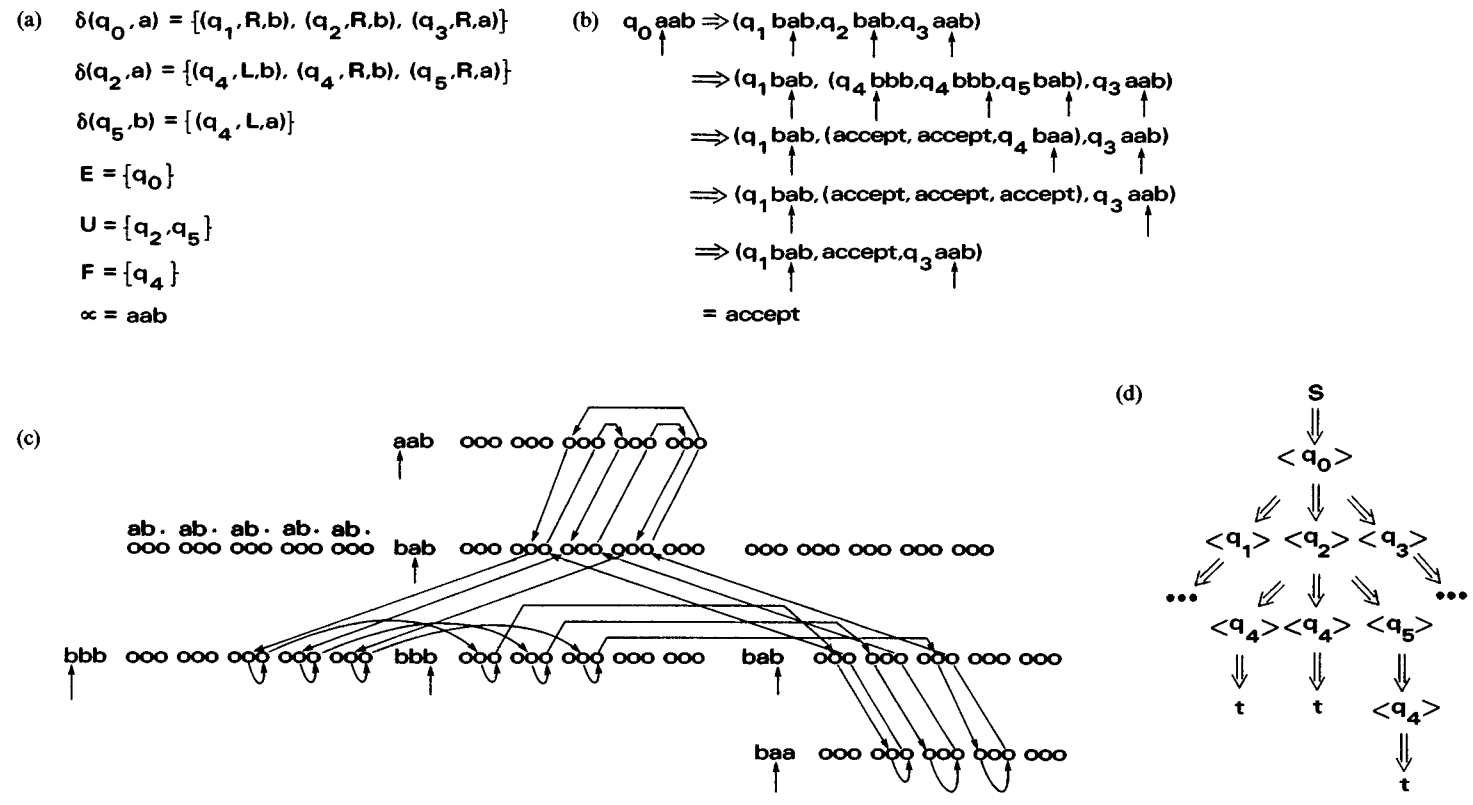


FIG. 1. A circular dependency network corresponding to a machine acceptance. (a) The description of a machine *M*. (b) A particular set of machine moves leading to the acceptance of the input *aab*. (c) Corresponding circular dependency network, the simulated machine configuration is indicated with each attribute set. (d) Parse tree for the string with circular dependency network.

(2) Consider a tape move of the form

$$\delta(q, x) = \{(q_1, y, R), (q_2, z, L), (q_3, w, L)\},$$

where q is an existential state. We generate the following production and rules:

$$\begin{aligned}
&\langle q\rangle^0 \to \langle q_1\rangle^1\langle q_2\rangle^2\langle q_3\rangle^3\\
&\quad C^1(n-1, y) := C^0(n, x)\\
&\quad C^2(n+1, z) := C^0(n, x)\\
&\quad C^3(n+1, w) := C^0(n, x)\\
&\quad C^1(j-1, v) := C^0(j, v) \quad \text{for each} \quad v \in \Sigma, \quad j = 2, 3, \ldots, 2n-1, \quad j \neq n,\\
&\quad C^2(j+1, v) := C^0(j, v) \quad \text{for each} \quad v \in \Sigma, \quad j = 2, 3, \ldots, 2n-2, \quad j \neq n,\\
&\quad C^3(j+1, v) := C^0(j, v) \quad \text{for each} \quad v \in \Sigma, \quad j = 2, 3, \ldots, 2n-2, \quad j \neq n,\\
&\quad C^0(j, *) := C^1(j-1, *) + C^2(j+1, *)\\
&\qquad\qquad + C^3(j+1, *) \quad j = 1, 2, \ldots, 2n-1,
\end{aligned}$$

where $C(k, *)$ is defined to be some constant for $k > 2n-1$ and $k < 1$.

(3) For a tape move of the form

$$(q, x) = \{(q_1, y, L), (q_2, z, L), (q_3, w, R)\},$$

where q is a universal state, we create the following production and rules:

$$\begin{aligned}
&\langle q\rangle^0 \to \langle q_1\rangle^1\langle q_2\rangle^2\langle q_3\rangle^3\\
&\quad C^1(n+1, y) := C^0(n, x)\\
&\quad C^2(n+1, z) := C^1(n+1, *)\\
&\quad C^3(n-1, w) := C^2(n+1, *)\\
&\quad C^0(n, *) := C^3(n-1, *)\\
&\quad C^1(j+1, v) := C^0(j, v) \quad \text{for each} \quad v \in \Sigma, \quad j = 1, 2, \ldots, 2n-2, \quad j \neq n,\\
&\quad C^2(j, v) := C^1(j, *) \quad \text{for each} \quad v \in \Sigma, \quad j = 1, 2, \ldots, 2n-1, \quad j \neq n+1,\\
&\quad C^3(j-1, v) := C^2(j+1, *) \quad \text{for each} \quad v \in \Sigma, \quad j = 2, 3, \ldots, 2n-2, \quad j \neq n,\\
&\quad C^0(j, *) := C^3(j-1, *) \quad \text{for each} \quad v \in \Sigma, \quad j = 1, 2, \ldots, 2n-2, \quad j \neq n.
\end{aligned}$$

(4) Finally, for each final state q_f we create a production and rules of the form

$$\begin{aligned}
&\langle q_f\rangle \to t\\
&\quad C(j, *) := C(j, a) + C(j, b) \quad \text{for} \quad j = 1, 2, \ldots, 2n-1,
\end{aligned}$$

where t is the only terminal symbol in the grammar.

This completes the construction. A proof of correctness is similar to but simpler than that given in [3] and is not given here. Figure 1 shows a circular dependency network corresponding to an accepting computation of some machine.

The complexity analysis is quite similar to that in [3] except that there are fewer objects to count here. We have half as many attributes ($(S+1)(2n-1)$ where $S = |\Sigma|$), and there are many fewer productions. The size of the grammar, however, is still $Kn \log n$, where K is some constant dependent on the machine M. The theorem is thus established.

4. *Conclusions*

The power of alternation was used to give a simpler construction for the exponential lower bound on the complexity of the circularity problem for attribute grammars. The concept of a universal state, which is easily simulated by the attribute grammar, removes the need for simulating a pushdown stack, which was the source of difficulty in the proof in [3]. It appears that alternating Turing machines correspond naturally

to grammars: existential states are analogous to different alternatives for a nonterminal, and universal states correspond to a single right-hand side for a nonterminal. ATMs should thus be helpful in obtaining complexity bounds for other grammar problems.

ACKNOWLEDGMENTS. I would like to thank Richard Ladner, Donald Stanat, Nancy Lynch, and an anonymous referee for their help.

REFERENCES

1. CHANDRA, A., AND STOCKMEYER, L.J. Alternation. Proc. 17th IEEE Symp. on Foundations of Computer Science, Houston, Texas, 1976, pp. 98–108.
2. KOZEN, D. On parallelism in Turing machines. Proc. 17th IEEE Symp. on Foundations of Computer Science, Houston, Texas, 1976, pp. 89–97.
3. JAZAYERI, M., OGDEN, W.F., AND ROUNDS, W.C. The intrinsically exponential complexity of the circularity problem for attribute grammars. *Commun. ACM 18*, 12 (Dec. 1975), 697–706.

RECEIVED NOVEMBER 1979; REVISED MAY 1980; ACCEPTED JUNE 1980