GENERAL CONTEXT-FREE RECOGNITION
IN LESS THAN CUBIC TIME

Leslie G. Valiant

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

January 1974

ABSTRACT

An algorithm for general context-free recognition is given that requires less than n^3 time asymptotically for input strings of length n.

INTRODUCTION

We shall exhibit a succession of reductions to show that general context-free recognition can be carried out at least as fast as Boolean matrix multiplication. Since the latter is known to be computable in $O(n^{2.81})$ bit operations by means of Strassen's algorithm for matrix multiplication in a ring [7], an indirect $O(n^{2.81})$ algorithm for context-free recognition can be derived. The resulting procedure is asymptotically more efficient than any of the best previously known recognition schemes (Kasami [5], Younger [8], Earley [2]), all of which require $O(n^3)$ time in the worst case.

PRELIMINARIES

Since every context-free grammar can be transformed into an equivalent one in Chomsky normal form [1], we need only consider grammars that are specified by quadruples (N, Σ, P, A_1) of the following type. N is a set of non-terminals $\{A_1,\ldots,A_h\}$ of which A_1 is the starting symbol, Σ is a set of terminals, and P is a set of productions, each of which has one of the following forms:

(i) $A_i \rightarrow A_j A_k$,

(ii) $A_i \rightarrow x$ for $x \in \Sigma$,

(iii) Λ (denoting that the null string is in the language)

We define a binary operation on arbitrary subsets N_1, N_2 of N as follows.

$$N_1 . N_2 = \{A_i \mid \exists\, A_j \in N_1,\ A_k \in N_2 \text{ such that } (A_i \rightarrow A_j A_k) \in P\}.$$

In terms of this we can define some operations on matrices that have subsets of N as elements. Thus we define matrix multiplication, $a.b = c$, for a and b of suitable size, as

$$c_{ik} = \bigcup_{j=1}^{n} a_{ij} . b_{jk} .$$

The transitive closure of a square matrix a can then be defined as

$$a^{+} = a^{(1)} \cup a^{(2)} \cup \ldots$$

where
$$a^{(i)} = \bigcup_{j=1}^{i-1} a^{(j)} . a^{(i-j)} \text{ and } a^{(1)} = a.$$

Observe that the case $N = \{A_1\}$, $P = \{(A_1 \rightarrow A_1A_1)\}$ gives the conventional multiplication and transitive closure operations for Boolean matrices. Note, however, the significant difference that while the Boolean "and" operator is associative, our more general binary operator is not in general.

For any algorithm we can define a complexity function that relates the input size to the number of elementary operations executed by the algorithm (for worst case inputs). $M(n)$ and $T(n)$ will denote such functions for algorithms for the problems of performing multiplication and transitive closure respectively on $n \times n$ upper triangular* matrices. $BM(n)$ will do the same for the multiplication of arbitrary $n \times n$ Boolean matrices. Our count of basic operations can be regarded as representing to within a constant factor the total number of bit operations required on a conventional computer. Since an $n \times n$ matrix may contain n^2 bits of information, all these complexity functions can be assumed to be of at least this magnitude.

For the grammar we have specified we use the conventional notation

$$A_i \overset{*}{\rightarrow} w$$

to denote that the strings $w \in \Sigma^*$ can be derived from A_i by some sequence of applications of productions from P. The number of elementary operations required to determine whether an arbitrary word w of length n belongs to the language (i.e, is derivable from A_1) we denote by $R(n)$.

REDUCING RECOGNITION TO TRANSITIVE CLOSURE

Let the input string be $x_1 \ldots x_n \in \Sigma^*$. First compute the $(n+1) \times (n+1)$ upper triangular matrix b defined by

$$b_{i,i+1} = \{A_k \mid (A_k \to x_i) \in P\}, \text{ and}$$
$$b_{i,j} = \emptyset \quad \text{for } j \neq i+1$$

From the definition of the multiplication operation, it is inductively evident that the elements of the transitive closure b^+ will be just those that have the property that

$$A_k \in b_{ij} \Leftrightarrow A_k \overset{*}{\to} x_i \ldots x_{j-1}.$$

We can therefore determine whether $A_1 \overset{*}{\to} x_1 \ldots x_n$ by computing $a = b^+$ and asking whether $A_1 \in a_{1,n+1}$. Taking into account the overheads of setting up the matrix b, we obtain the following.

THEOREM 1. $R(n) \leq T(n+1) + O(n^2)$. □

REDUCING TRANSITIVE CLOSURE TO MULTIPLICATION

We shall describe a recursive procedure for computing the transitive closure of an upper triangular matrix, that can be shown to be of about the same complexity as matrix multiplication. Several analogous procedures for the special case of Boolean matrix multiplication are known ([3], [4], [6]). However, these all assume associativity, and are therefore not applicable here. Instead of the customary method of recursively splitting into disjoint parts, we now require a more complex procedure based on "splitting with overlaps". Fortunately, and perhaps surprisingly, the extra cost involved in such a strategy can be made almost negligible.

Let b be an upper triangular $n \times n$ matrix. Define $b^{+(r:s)}$ to be the result of the following operations: (i) collapse b by removing all elements b_{ij} where $r < i \leq s$ or $r < j \leq s$, (ii) compute the transitive closure of the remaining $(n+r-s) \times (n+r-s)$ matrix, and (iii) expand the matrix back to its original size by restoring the elements that were removed to their rightful place.

The key observation on which our reduction depends is the following. If the submatrices of b specified by $[1 \leq i,j \leq s]$, and $[r < i,j \leq n]$ are both already transitively closed, and if $s \geq r$, then

$$b^+ = (b \cup b.b)^{+(r:s)}.$$

This expresses the facts that (i) to obtain b^+ from b we only need to complete the submatrix $[1 \leq i \leq r,\ s < j \leq n]$, and that (ii) all the new contributions that can arise directly from a pair of elements both outside this submatrix, can be obtained by squaring b just once. (N.B. An item at (i,j) can only contribute to one at (k,ℓ) if $k \leq i$ and $\ell \geq j$.)

Denote by P_k the operation of closing a matrix b of which the submatrices $[1 \le i,j \le n-n/k]$ and $[n/k < i,j \le n]$ are already closed. We can define these tasks for $k = 2,3$ and 4, recursively as follows.

P_2: (i) Apply P_2 to submatrix $[n/4 < i,j \le 3n/4]$

(ii) Apply P_3 to submatrices $[1 \le i,j \le 3n/4]$ and $[n/4 < i,j \le n]$ of the result of (i)

(iii) Apply P_4 to the result of (ii)

P_3: (i) Compute $b \cup b.b$

(ii) Apply $+(n/3, 2n/3)$ to the result of (i) using P_2

P_4: (i) Compute $b \cup b.b$

(ii) Apply $+(n/4, 3n/4)$ to the result of (i) using P_2

If $T_i(n)$ is the time bound on procedure P_i when applied to an $n \times n$ matrix, the recursive definitions give immediately that

$$\begin{aligned}
T_2(n) &\le T_2(n/2) + 2T_3(3n/4) + T_4(n),\\
T_3(n) &\le M(n) + T_2(2n/3) + O(n^2), \text{ and}\\
T_4(n) &\le M(n) + T_2(n/2) + O(n^2).
\end{aligned}$$

Eliminating T_3 and T_4 gives

$$T_2(n) \le 4T_2(n/2) + 3M(n) + O(n^2).$$

Assuming that n is a power of 2, and that there is some growth factor $\gamma \ge 2$ such that for all m, $M(2^{m+1}) \ge 2^{\gamma}M(2^m)$, we obtain that

$$T_2(n) \le O(n^2 \log n) + 3M(n) \cdot \sum_{m=0}^{\log n} 2^{(2-\gamma)m}.$$

Since $\sum_{m=0}^{\infty} x^m$ converges if $|x| < 1$, we conclude that if there is a suitable $\gamma > 2$, then

$$T_2(n) \leq M(n).\ const.,$$

and

$$T_2(n) \leq M(n).\log n.const.$$

otherwise (i.e. if only $\gamma = 2$ is possible).

We can compute the transitive closure by closing the submatrices $[1 \leq i,j \leq n/2]$ and $[n/2 < i,j \leq n]$, and then applying P_2. This gives

$$T(n) \leq 2T(n/2) + T_2(n) + O(n^2).$$

Assuming now that T_2 is also well behaved (i.e. that there is a $\delta \geq 2$ such that for all m $T_2(2^{m+1}) \geq 2^{\delta} T_2(2^m)$), we obtain

$$T(n) \leq O(n^2) + T_2(n) \cdot \sum_{m=0}^{\log n} 2^{(1-\delta)m} \leq T_2(n).const.$$

If n is not a power of 2, we can pad the matrix with null sets to increase its size to the next power of two, and then apply the above procedure. If we assume in addition that $M(n) \geq M(2^{\lceil \log_2 n \rceil}).const.$, we can deduce that the above obtained bounds also hold for arbitrary n. We therefore conclude the following.

THEOREM 2. If $M(n)$ and $T_2(n)$ are well behaved (in the sensesstated above), then, if there exists a growth factor $\gamma > 2$ for $M(n)$ then

$$T(n) \leq M(n).const.,$$

and

$$T(n) \leq M(n).\log n.const. \text{ otherwise.} \quad \Box$$

It is observed by Fischer and Meyer [4] that the closure of a $3n \times 3n$ Boolean matrix that is zero everywhere except for the partitions $[1 \le i \le n, n < j \le 2n]$ and $[n < i \le 2n, 2n < j \le 3n]$, gives the product of these partitions. This is clearly applicable here also and provides a converse inequality

$$M'(n) \le T(3n) + O(n^2).$$

In conclusion we note that the purpose of using P_2, P_3 and P_4 was to derive the tight bounds of Theorem 2. A looser bound, that still leads to a sub-cubic recognition algorithm, can be obtained from the following simpler procedure: (i) compute $b^{+(2n/3,n)}$ and $b^{+(0,n/3)}$, (ii) square the union of the results of (i), and (iii) apply $+(n/3, 2n/3)$ to the union of the results of (i) and (ii).

REDUCING MULTIPLICATION TO BOOLEAN MULTIPLICATION

Given matrices a and b (both assumed $n \times n$ for simplicity) we want to compute c such that

$$c_{ij} = \bigcup_{k=1}^{n} a_{ik} \cdot b_{kj}.$$

First compute the Boolean matrices a',$(hn \times n)$, and b',$(n \times hn)$, from a and b respectively (h being the size of N) such that

$$a'_{pq} = 1 \text{ iff } A_i \in a_{rq} \text{ for } i = p \bmod h \text{ and } r = \lceil p/h \rceil, \text{ and}$$
$$b'_{pq} = 1 \text{ iff } A_i \in b_{pr} \text{ for } i = q \bmod h \text{ and } r = \lceil q/h \rceil.$$

The Boolean product c' of a' and b' then has the property that $c'_{pq} = 1$ iff for some s $a'_{ps} = 1$ and $b'_{sq} = 1$, i.e. iff for some s

$$A_i \in a_{rs} \text{ where } i = p \bmod h \text{ and } r = \lceil p/h \rceil, \text{ and}$$
$$A_j \in b_{st} \text{ where } j = q \bmod h \text{ and } t = \lceil q/h \rceil.$$

By definition therefore

$$c_{rt} = \bigcup_{\substack{1 \le i \le h \\ 1 \le j \le h}} \{A_k \mid (A_k \to A_i A_j) \in P \text{ and } c'_{rh-h+i,th-h+j} = 1\}$$

Thus we compute ab by generating a' and b', performing Boolean multiplication on them, and abstracting c from the result according to the above relation. Since only the multiplication can require more than $O(n^2)$ time, we can deduce the following by means of a padding argument.

THEOREM 3. $M(n) \le BM(n).\text{const}$. □

SUMMARY

The last two theorems establish the following intermediate result: if the complexity functions grow uniformly as assumed then the problem of computing the transitive closure of a "parse" matrix is of essentially the same difficulty as that of Boolean matrix multiplication. The difference between their complexities can be bounded by a multiplicative constant, unless they grow more slowly than $n^{2+\epsilon}$ for any ϵ, in which case the gap is still no more than a factor of log n.

To reach our main conclusion we use the known fact that Boolean matrix multiplication does not require time $O(n^3)$. Treating the Boolean elements as integers modulo n+1, applying Strassen's algorithm [7], and reducing the non-zero elements to one in the result gives the Boolean product in $O(n^{2.81})$ bit operations [3]. We can therefore deduce from Theorems 1, 2, and 3 that context-free languages can be recognized in time $O(n^{2.81})$.

We have therefore arrived indirectly at an algorithm for general context-free recognition that is asymptotically more efficient than any previously known.

ACKNOWLEDGMENT

I should like to thank Mario Schkolnick for his helpful comments on a draft of this paper.

REFERENCES

[1] CHOMSKY, N. On certain formal properties of grammars. Inf. and Control, 2, 137-167 (1959).

[2] EARLEY, J. An efficient context-free parsing algorithm. CACM 13, 2, 94-102 (1970).

[3] FISCHER, M. J., and MEYER, A. R. Boolean matrix multiplication and transitive closure. IEEE Conf. Rec. 12th Symp. on Switching and Automata Theory (1971).

[4] FURMAN, M. E. Application of a method of fast multiplication of matrices in the problem of finding the transitive closure of a graph. Dokl. Akad. Nauk SSSR 194, 3 (1970) = Sov. Math. Dokl. 11, 5 (1970), 1252.

[5] KASAMI, J. An efficient recognition and syntax analysis algorithm for context-free languages, Report of Univ. of Hawaii,(1965).

[6] MUNRO, I. Efficient determination of the transitive closure of a directed graph. Inf. Proc. Letters, 1, 56-58 (1971).

[7] STRASSEN, V. Gaussian elimination is not optimal. Numer. Math. 13, 354-456 (1969).

[8] YOUNGER, D. H. Recognition and parsing of context-free languages in time n^3. Inf. and Control, 10, 189-208 (1967).