BOOLEAN MATRIX MULTIPLICATION AND TRANSITIVE CLOSURE[†]

M.J. Fischer and A.R. Meyer

Massachusetts Institute of Technology
Cambridge, Massachusetts

Summary

Arithmetic operations on matrices are applied to the problem of finding the transitive closure of a Boolean matrix. The best transitive closure algorithm known, due to Munro, is based on the matrix multiplication method of Strassen. We show that his method requires at most $O(n^{\alpha} \cdot P(n))$ bitwise operations, where $\alpha = \log_2 7$ and $P(n)$ bounds the number of bitwise operations needed for arithmetic modulo n+1. The problems of computing the transitive closure and of computing the "and-or" product of Boolean matrices are shown to be of the same order of difficulty. A transitive closure method based on matrix inverse is presented which can be used to derive Munro's method.

Application of the Strassen Algorithm to Boolean Matrix Operations

Strassen's algorithm[7] for computing the product of two n×n integer matrices in $O(n^{\alpha})$ operations, where $\alpha = \log_2 7 \simeq 2.8$, can be applied to the problem of forming the "and-or" product* of two Boolean matrices: form the real integer product using Strassen's method and obtain the Boolean result by normalizing all non-zero entries to ones. The transitive closure** of a Boolean matrix A can then be obtained by forming $(I \vee A)^K$ for any number $K \geq n$. By the method of repeated squaring, this can be accomplished in $\log_2 n$ Boolean matrix multiplications, yielding a total of $O(n^{\alpha} \cdot \log_2 n)$ "operations" in the worst case, a significant improvement over the n^3 operations required in the worst case by Warshall's method.[8]

This observation has been made independently by Furman and Munro[3,5], though both their constructions have a minor lacuna: operations in Warshall's and Strassen's methods are not the same. Strassen's method counts arithmetic operations on integers while Warshall's method counts "bitwise" operations.

To analyze the cost of forming the Boolean matrix product using Strassen's method, we must bound the size of the numbers involved. Clearly no entry in the product exceeds n (given that the operands are 0-1 valued matrices), but a more careful analysis of the algorithm is required to verify that the intermediate results remain small.

A proof by induction shows that in fact the integers appearing as intermediate results in Strassen's algorithm applied to compute the integer product of 0-1 valued n×n matrices never grow larger than $16n^2$. A more elegant observation suggested to us by M. Rabin ensures that intermediate results remain between zero and n by carrying out the arithmetic operations modulo n+1. The validity of Strassen's algorithm is based only on the ring properties of integers, and since the values of the integer product lie between zero and n, operations modulo n+1 will yield the correct integer product matrix.

Let P(n) bound the number of bitwise operations needed to add or multiply integers modulo n+1. We assume for convenience that P is nondecreasing. (The familiar algorithms for modular arithmetic imply $P(n) \leq O(\log_2{}^2 n)$, and if one appeals to some of the recent fast multiplication techniques one can show $P(n) = o((\log_2 n)^{1+\varepsilon})$ for any $\varepsilon > 0$.)[4] We now have

Theorem 1: The "and-or" product of two n×n Boolean matrices is computable with at most $O(n^{\alpha} \cdot P(n))$ bitwise operations.

By the remark above about repeated squaring, we conclude

Corollary 1: The transitive closure of an n×n Boolean matrix is computable with at most $O(n^{\alpha} \cdot \log_2 n \cdot P(n))$ bitwise operations.

Transitive Closure in Multiplication Time

Given a directed graph with n nodes (or equivalently, given an n×n Boolean connection matrix), all cycles can be "removed" in $O(n^2)$ steps, that is, a new graph can be formed in which all nodes of the original graph belonging to the same cycle are merged together to form a single new node.[2,5,6] The new graph is necessarily acyclic, permitting its connection matrix to be expressed in upper triangular form (and this transformation also requires only $O(n^2)$ steps).

Arlazarov et. al. use this idea in a proof that transitive closure can be computed in $O(n^3/\log_2 n)$ steps[1] (although their definition of step differs slightly from ours). Munro[5] uses it to establish that transitive closure can be computed in essentially the time required to perform one Boolean matrix multiplication.

Theorem 2 (Munro): If the "and-or" product of any two n×n Boolean matrices is computable within M(n) bitwise operations and M is monotonic and satisfies $M(kn) \geq k^{\beta} \cdot M(n)$ for some $\beta > 2$ and all $k,n \geq 1$, then the transitive closure of any n×n Boolean matrix can be computed in O(M(2n)) bitwise operations.

Proof: We may base a recursive procedure for computing the transitive closure on the following identity, where A (and hence A_{11} and A_{22}) are upper triangular:

$$A^{*} = \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix}^{*} = \begin{pmatrix} A_{11}{}^{*} & A_{11}{}^{*}A_{12}A_{22}{}^{*} \\ 0 & A_{22}{}^{*} \end{pmatrix}.$$

[†] 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.

*The "and-or" product C of two Boolean matrices A and B is defined by $C_{ij} = \bigvee_{k=1}^{n} (A_{ik} \wedge B_{kj})$.

**The transitive closure A^{*} of a Boolean matrix A $= I \vee A \vee A^2 \vee \ldots$, where I is the identity matrix.

Assuming $n = 2^m$, this gives rise to the recursive relation

$$T(2^m) \le 2\cdot M(2^{m-1}) + 2\cdot T(2^{m-1}) + O(m) \qquad (1)$$

where T(n) is the number of bitwise operations needed to compute the transitive closure of an n×n Boolean matrix.

Solving (1) using the fact that T(1) = 0, we get that

$$T(2^m) \le \sum_{k=1}^{m} 2^k\cdot M(2^{m-k}) + O(m\cdot 2^m). \qquad (2)$$

By assumption, $M(2^m) \ge 2^{k\beta}\cdot M(2^{m-k})$, so from (2),

$$T(2^m) \le \sum_{k=1}^{m} 2^k\cdot 2^{-k\beta}\cdot M(2^m) + O(m\cdot 2^m)$$

$$< M(2^m)\cdot \sum_{k=1}^{\infty} 2^{(1-\beta)k} + O(m\cdot 2^m).$$

Since $2^{1-\beta} < 1$, the series converges, so taking c to be the limit, we get

$$T(2^m) < c\cdot M(2^m) + O(m\cdot 2^m) = O(M(2^m)).$$

The result for arbitrary n follows by padding a given n×n matrix with zeros to make it of order $2^m \times 2^m$, $m = \lceil \log_2 n \rceil$, and taking the transitive closure of that.

This provides an improvement to Corollary 1:

Corollary 2: The transitive closure of an n×n Boolean matrix is computable with cost at most $O(n^\alpha\cdot P(2n))$.

Proof: We must show that the function $M(n) = c\cdot n^\alpha\cdot P(n)$ satisfies the conditions of Theorem 2. By the monotonicity of P,

$$\begin{aligned} M(kn) &= c\cdot k^\alpha n^\alpha\cdot P(kn) \\ &\ge c\cdot k^\alpha n^\alpha\cdot P(n) \\ &= k^\alpha\cdot M(n), \end{aligned}$$

and since $\alpha > 2$, the corollary follows.

Matrix Product Using Transitive Closure

We now prove a partial converse to Theorem 2, showing that transitive closure and Boolean matrix product are of the same general order of difficulty.

Theorem 3: If the transitive closure of any n×n Boolean matrix is computable in $O(n^\beta)$ bitwise operations for some β, then the "and-or" product of two n×n Boolean matrices can also be computed in $O(n^\beta)$ bitwise operations.

Proof: We observe that

$$\begin{pmatrix} 0 & A & 0 \\ 0 & 0 & B \\ 0 & 0 & 0 \end{pmatrix}^{*} = \begin{pmatrix} I & A & AB \\ 0 & I & B \\ 0 & 0 & I \end{pmatrix},$$

and hence product takes at most $O((3n)^\beta) = O(n^\beta)$ operations.

Transitive Closure Using Matrix Inverse

Another apparently different method suggested by M.S. Paterson is motivated by the formal identity

$$(I-B)^{-1} = \frac{1}{1-B} = 1 + B + B^2 + \ldots$$

Choosing $\varepsilon > 0$ sufficiently small, and letting $B = \varepsilon A$ will ensure that the series on the right converges to $(I-\varepsilon A)^{-1}$. The transitive closure of A can then be obtained by inverting $I-\varepsilon A$ and normalizing.

Theorem 4: For any n×n Boolean matrix A and $0 < \varepsilon < \frac{1}{n}$, the matrix $I-\varepsilon A$ is nonsingular and $A^* = (I-\varepsilon A)^{-1}$ normalized.

Proof: It is easy to show by induction that every element of $(\varepsilon A)^k$ is non-negative and bounded by $\varepsilon\cdot(n\varepsilon)^{k-1}$. Since $(n\varepsilon) < 1$, $C = \sum_{k=0}^{\infty} (\varepsilon A)^k$ converges. Also, $C\cdot(I-\varepsilon A) = I$, so $(I-\varepsilon A)$ must be nonsingular with inverse C. Clearly, $(C)_{ij} > 0$ iff $(A^*)_{ij} = 1$, so $A^* = (I-\varepsilon A)^{-1}$, normalized.

Strassen also shows that matrix inversion can be performed in $O(n^\alpha)$ arithmetic operations, so that Theorem 4 would seem to provide another $O(n^\alpha)$ transitive closure algorithm. Strassen notes that his method may fail even on nonsingular matrices because certain intermediate matrices in the calculation are singular. However Paterson has pointed out that the conditions of Theorem 4 are sufficient for Strassen's method to be applicable to $I-\varepsilon A$.

This idea of taking inverses to compute transitive closures might also be useful in situations where standard matrix inversion programs were readily available. Unfortunately our best current estimates suggest that the arithmetic precision required to invert $I-\varepsilon A$ is prohibitive both for Strassen's method and for Gaussian elimination. Thus it appears that in this case the cost of arithmetic operations is no longer a minor lacuna as it was in Theorem 1, although we conjecture that this difficulty can be overcome. For example, if a Boolean matrix is in upper triangular form, R. Mandl has observed that the method of Theorem 4 works nicely:

Let A be an upper triangular zero-one valued matrix with zero diagonal. It is easy to verify that the series

$$I + A + A^2 + \ldots$$

converges and hence equals $(I-A)^{-1}$. Moreover, Strassen's inverse formula simplifies to

$$(I-A)^{-1} = \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix}^{-1} = \begin{pmatrix} A_{11}^{-1} & -A_{11}^{-1}A_{12}A_{22}^{-1} \\ 0 & A_{22}^{-1} \end{pmatrix}$$

where A_{11} and A_{22} are similarly upper triangular. Using this formula recursively to compute $(I-A)^{-1}$ and using Strassen's product method modulo n+1 to compute $A_{11}^{-1}A_{12}A_{22}^{-1}$ requires $O(n^\alpha)$ operations on integers modulo n+1. (Note that we are not assuming that inverses mod n+1 exist since the only element inverted at the bottom level of the recursion is one.) By normalizing $(I-A)^{-1}$ we again obtain the transitive closure A^*. This is essentially another derivation of

Theorem 2, since the recurrence above becomes identical to the relation used to prove Theorem 2 if we simply replace the inverse symbol by "*" and erase the minus sign!

Conclusion

When we began to study operations on Boolean matricies, we hoped that simple "graph-theoretic" methods for multiplication and transitive closure would emerge which might then lead to improved algorithms for real matrices. We were surprised to discover that the general "algebraic" methods when specialized to Boolean matrices not only achieve the best time bounds known for Boolean multiplication and transitive closure, but in fact generate the only known algorithms for achieving these bounds. Looked at differently, restricting attention to the special case of Boolean matrices has not enabled us to exploit any particular special properties to obtain better algorithms, and in fact, it is difficult even to explain the algorithms of Theorem 1 and Theorem 2 directly in graph-theoretic terms. We believe that such an explanation would provide new insight into the case of real matrix multiplication.

References

1. Arlazarov, V.L., Dinic, E.A., Kronod, M.A., and Faradžev, I.A., "On Economical Construction of the Transitive Closure of an Oriented Graph," Dokl. Akad. Nauk SSSR 194, 3 (1970) = Soviet Math. Dokl. 11, 5 (1970), 1209-1210.

2. Faradžev, I.A., "Effective Algorithms for the Solution of Certain Problems in Directed Graphs," Ž. Vyčisl. Mat. i Mat. Fiz. 10 (1970), 1049-1054.

3. 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) = Soviet Math. Dokl. 11, 5 (1970), 1252.

4. Knuth, D.E., The Art of Computer Programming, volume 2, Addison-Wesley, Reading, Mass., 1969.

5. Munro, I., "Efficient Determination of the Strongly Connected Components and Transitive Closure of a Directed Graph," manuscript.

6. Purdom, P., "A Transitive Closure Algorithm," BIT 10, 1 (1970), 76-94.

7. Strassen, V., "Gaussian Elimination is not Optimal," Numer. Math. 13 (1969), 354-356.

8. Warshall, S., "A Theorem on Boolean Matrices," JACM 9, 1 (1962), 11-12.