A Theorem on Boolean Matrices*

STEPHEN WARSHALL†

Computer Associates, Inc., Woburn, Massachusetts

Given two boolean matrices A and B, we define the boolean product $A \wedge B$ as that matrix whose (i, j)th entry is $\mathbf{v}_k(a_{ik} \wedge b_{kj})$.

We define the boolean sum $A \vee B$ as that matrix whose (i, j)th entry is $a_{ij} \vee b_{ij}$.

The use of boolean matrices to represent program topology (Prosser [1], and Marimont [2], for example) has led to interest in algorithms for transforming the $d \times d$ boolean matrix M to the $d \times d$ boolean matrix M' given by:

$$M' = \mathop{\mathbf{v}}_{i=1}^{d} M^i \quad \text{where we define } M^1 = M \text{ and } M^{i+1} = M^i \wedge M.$$

The convenience of describing the transformation as a boolean sum of boolean products has apparently[1] suggested the corresponding algorithms, the running times of which increase—other things being equal—as the cube of d. While refraining from comment on the area of utility of such matrices, we prove the validity of an algorithm whose running time goes up slightly faster than the square of d.

THEOREM. *Given a square $(d \times d)$ matrix M each of whose elements m_{ij} is 0 or 1. Define M' by $m'_{ij} = 1$ if and only if either $m_{ij} = 1$ or there exist integers $k_1, \cdots, k_n$ such that $m_{ik_1} = m_{k_1k_2} = \cdots = m_{k_{n-1}k_n} = m_{k_nj} = 1$; $m'_{ij} = 0$, otherwise.[2] Define M^* by the following construction:[3]*

0. Set $M^* = M$.
1. Set $i = 1$.
2. $(\forall_j \ni : m^*_{ji} = 1)(\forall k)$ set $m^*_{jk} = m^*_{jk} \vee m^*_{ik}$.
3. Increment i by 1.
4. If $i \leqq d$, go to step 2; otherwise, stop.

We assert $M^* = M'$.

PROOF. Trivially, $m^*_{ij} = 1 \Rightarrow m'_{ij} = 1$. For, either m^*_{ij} was unity initially $(m_{ij} = 1)$—in which case m'_{ij} is surely unity—or m^*_{ij} was set to unity in step two. That is, there were, at the L_0th application of step two, $m^*_{iL_0} = m^*_{L_0j} = 1$.

* Received September, 1960; revised February, 1961.

† This work was performed by the author at Technical Operations, Inc., under Department of the Air Force Contract AF 33(600)-35190.

[1] Prosser, op. cit. In his definition of Boolean sum and product, Prosser uses "$\vee$" for product and "$\wedge$" for sum. This is apparently a typographical error, for his subsequent usage is consistent with ours.

[2] This definition of M' is trivially equivalent to the previous one.

[3] This definition by construction is equivalent to the recursive definition: 0. $(m_{ij})_0 = m_{ij}$; 1. $(m_{ij})_{n+1} = (m_{ij})_n \vee ((m_{i,n+1})_n \wedge (m_{n+1,j})_n)$; 2. $m^*_{ij} = (m_{ij})_d$.

Each of these, similarly, either came directly from M or from a previous application of step two. Since there are exactly d applications of step two, this procedure is finite and leads to $m^*_{iL_A} = m^*_{L_A L_{A-1}} = \cdots = m^*_{L_2 L_1} = m^*_{L_1 L_0} = m^*_{L_0 R_1} = \cdots = m^*_{R_B j} = 1$, where all the corresponding entries in M were unity. This is exactly the sequence required in the definition of M' (to within redundant elements which may simply be struck out) to imply that $m'_{ij} = 1$.

We have yet to prove that $m'_{ij} = 1 \Rightarrow m^*_{ij} = 1$. Assume this is false. Then there is a sequence of integers $i \neq k_1 \neq k_2 \neq \cdots \neq k_n \neq j$ such that $m_{ik_1} = m_{k_1 k_2} = \cdots = m_{k_n j} = 1$, but $m^*_{ij} = 0$. Let $L = \{ x \mid (1 \leqq x \leqq n) \text{ and } m^*_{ik_x} = 1 \}$. Let λ be the largest element of L. Surely $m^*_{ik_\lambda}$ must have been changed from zero to unity by an application of step two (for if $m_{ik_\lambda} = 1$, since $m_{k_\lambda k_{\lambda+1}} = 1$, $m^*_{ik_{\lambda+1}} = 1$ by the k_λth step 2, which would contradict the definition of λ), the γth, say. This γ must be less than k_λ; for immediately after the k_λth iteration of step two, $(\forall p) m^*_{pk_\lambda} = 1 \Rightarrow m^*_{pk_{\lambda+1}} = 1$. Any p_0 such that $m^*_{p_0 k_\lambda}$ is set to one *after* this will result from the p_1th iteration of step two when $m^*_{p_1 k_\lambda} = m^*_{p_0 p_1} = 1$ leads to $m^*_{p_0 k_\lambda} = 1$. But if $m^*_{p_1 k_\lambda} = 1$ at this time, then either $m^*_{p_1 k_\lambda} = 1$ at the time of the k_λth iteration (in which case $m^*_{p_1 k_{\lambda+1}} = 1$ also), or $m^*_{p_1 k_\lambda}$ is set to one at the p_2th iteration where $k_\lambda < p_2 < p_1$. We thus generate a finite ordered set $p_1 > p_2 > \cdots > p_q > k_\lambda$, where $m^*_{p_q k_\lambda} = 1$ at the time of the k_λth iteration, whence $m^*_{p_q k_{\lambda+1}} = 1$ immediately after that iteration. Then the sequence of iterations designated by the p's will surely result in $m^*_{p_0 k_{\lambda+1}} = 1$ after the p_1th iteration. Since p_0 was an arbitrary element, this is true if, in particular, $p_0 = i$. Thus, if $\gamma \geqq k_\lambda$, $m^*_{ik_{\lambda+1}} = 1$, a contradiction.

But if $\gamma < k_\lambda$, then $m^*_{ik_\lambda} = m^*_{k_\lambda k_{\lambda+1}} = 1$ before the k_λth iteration, whence $m^*_{ik_{\lambda+1}} = 1$ after that iteration of step two, a contradiction. Therefore, the assertion is true. Q.E.D.

REFERENCES

1. Prosser, Reese T. Applications of Boolean matrices to the analysis of flow diagrams. Proc. Eastern Joint Comput. Conf. No. 16 (1959), 133.
2. Marimont, Rosalind B. A new method of checking the consistency of precedence matrices. *J. ACM 6*, (1959) 164.