AN IMPROVED ALGORITHM FOR BOOLEAN MATRIX MULTIPLICATION

N. Santoro* and J. Urrutia**

SCS-TR-38
January 1984

* School of Computer Science, Carleton University, Ottawa, Canada. K1S 5B6
** Computer Science Department, University of Ottawa, Ottawa, Canada. K1N 5B4

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant No. A.2415.

1. INTRODUCTION

In parallel with the analysis of asymptotically fast methods, the research on Boolean matrix multiplication has also focused on the determination of 'efficient' bounds; that is, finding non-optimal but practical algorithms that outperform the asymptotic algorithms for a bounded matrix size or for special classes of matrices. Example of these results are the $O(N^2)$ algorithms for multiplying $N \times N$ sparse or dense Boolean matrices [2], and the $O(N^3/\log N)$ algorithm for multiplying arbitrary $N \times N$ Boolean matrices [1]. The latter algorithm, known as the "Four Russians' Method", unfortunately requires $O(N^3/\log N)$ additional bits to store the $O(N/\log N)$ sets, each containing N rows of N bits each.

In this paper, a new algorithm for Boolean matrix multiplication is presented; this algorithm is based on some properties of the product matrix, and it is shown to require $O(N^3/\log N)$ bit operations (thus, achieving the Four Russian' bound) but only $O(N \log N)$ bits of additional storage.

The paper is organized as follows. In the next section, some properties of the product matrix are discussed; based on these properties, in Section 3, an algorithm is presented which multiplies a $p \times N$ matrix by a $N \times N$ matrix in $O(N^2)$ bit operations using $O(N \log N)$ bits of additional storage where $p \leq [\log_2 N]$; finally, in Section 4, this algorithm is employed to obtain the claimed result. In the following, all logarithms are in base two and (P,Q,R) denotes the problems of multiplying a $P \times Q$ by a $Q \times R$ Boolean matrix.

2. PROPERTIES OF THE PRODUCT MATRIX

Consider the product $C = A \times B$ where A is a $p \times N$ Boolean matrix, $p \leq [\log N]$, and B is a $N \times N$ Boolean matrix.

Consider the sets

$$Z^{i+1}_{2k-1} = \{x \in Z^i_k : A[i+1, x] = 1\} \tag{1a}$$

and

$$Z^{i+1}_{2k} = \{x \in Z^i_k : A[i+1, x] = 0\} \tag{1b}$$

$(0 \leq i < p,\ 1 \leq k \leq 2^i)$, where

$$Z^0_1 = \{1,2,\ldots, N\}.$$

Obviously, $Z^{i+1}_{2k-1} \cap Z^{i+1}_{2k} = \emptyset$ and $Z^{i+1}_{2k} \cup Z^{i+1}_{2k} = Z^i_k$ $(0 \leq i < p,\ 1 \leq k \leq 2^i)$

Example 1 Consider the matrix

$$A = \begin{matrix} 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1 \\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0 \\ 1\ 1\ 0.0\ 1\ 0\ 0\ 1 \end{matrix}$$

In this case, the sets Z^i_k's are as follows

$Z^0_1 = \{1,\ldots,8\}$;

$Z^1_1 = \{2,3,6,8\}$, $Z^1_2 = \{1,4,5,7\}$;

$Z^2_1 = \{2,3\}$, $Z^2_2 = \{6,8\}$, $Z^2_3 = \{4,5\}$, $Z^2_4 = \{1,7\}$;

$Z^3_1 = \{2\}$, $Z^3_2 = \{3\}$, $Z^3_3 = \{8\}$, $Z^3_4 = \{6\}$, $Z^3_5 = \{5\}$, $Z^3_6 = \{4\}$,

$Z^3_7 = \{1\}$, $Z^3_8 = \{7\}$.

Let $f^i = \{1,2,\ldots,N\} \to \{1,2,\ldots,N\}$ $(0 \leq i \leq p)$ be the mapping

$$f^i(x) = k \text{ iff } x \in Z^i_k \tag{2}$$

A bijection $\pi:\{1,2,\ldots,N\} \to \{1,2,\ldots,N\}$ is said to be

<u>i-canonical</u> $(0 \leq i \leq p)$ if for all $x,y \in \{1,2,\ldots,N\}$

$$\pi^{-1}(x) < \pi^{-1}(y) \quad \text{if} \quad f^i(x) < f^i(y) \tag{3}$$

Obviously, the identity function $\mathcal{I}$ is 0-canonical.

Given a bijection $\pi: \{1,\ldots,N\} \to \{1,\ldots,N\}$, let

$$\pi[i,k] = \{\pi^{-1}(x) : x \in Z^i_k\}$$

$$\ell_\pi[i,k] = \min\{a \in \pi[i,k]\}$$

$$h_\pi[i,k] = \max\{a \in \pi[i,k]\}.$$

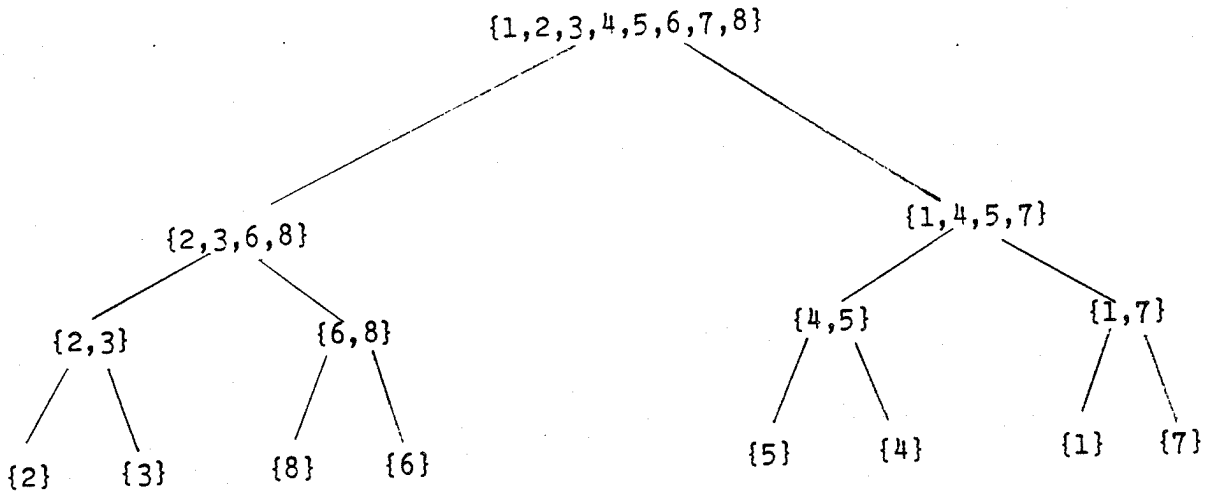
<u>Lemma 1</u> Let π be i-canonical. Then, for all non-empty Z^i_k $(1 \leq k \leq 2^i)$

$$Z^i_k = \{\pi(\ell_\pi[i,k]),\ \pi(\ell_\pi[i,k]+1),\ \ldots,\ \pi(h_\pi[i,k])\}$$

Proof. By definition of i-canonical mapping, each $x \in Z^i_k$ is such that $\pi^{-1}(w) < \pi^{-1}(x) < \pi^{-1}(z)$ for all $w \in Z^i_{k'}$, $z \in Z^i_{k''}$ where $k' < k < k''$. Hence, $\pi[i,k]$ is a sequence of consecutive integers. By definition of $\ell_\pi[i,k]$ and $h_\pi[i,k]$, and since π is a bijection the Lemma follows. [ ]

The sets Z^i_k may conveniently be imagined as lying in a binary tree, each Z^i_k having Z^{i+1}_{2k-1} and Z^{i+1}_{2k} as left and right children. The i-th level of the tree then consists of the sets Z^i_k $(k = 1,2,\ldots,2^i)$ which form a partition (possibly with some empty parts) of $\{1,2,\ldots,N\}$; the function f^i is the characteristic function of the partition. If the nodes at the i-th level, $Z^i_1, Z^i_2, \ldots$, are sets of size $m^i_1, m^i_2, \ldots$, an i-canonical bijection is one that maps the first m^i_1 positive integers onto Z^i_1, the next m^i_2 integers onto Z^i_2, and so on.

In the case of the example above, we have the tree



Given an i-canonical bijection π_i, consider the following construction

Construction 2.1

1. For all $k(1 \leq k \leq 2^i)$, if $Z_k^i \neq \emptyset$

   a) Partition $\pi[i,k]$ into two sets P_k^i and Q_k^i such that

   $P_k^i = \{a \in \pi[i,k] : A[i+1, \pi_i(a)] = 1\}$ and

   $Q_k^i = \{a \in \pi[i,k] : A[i+1, \pi_i(a)] = 0\}$

   b) Define $\psi_k : \pi[i,k] \to \pi[i,k]$ to be a bijection such that for all $a \in P_k^i$, $b \in Q_k^i$ $\psi_k(a) < \psi_k(b)$.

2. Define $\pi_{i+1} : \{1,2,\ldots, N\} \to \{1,2,\ldots, N\}$ to be the bijection defined by

$$\pi_{i+1}^{-1}(x) = \psi_k(\pi_i^{-1}(x)) \quad \underline{\text{iff}} \quad \pi_i^{-1}(x) \in \pi[i,k]$$

Example 2

It is easy to verify that the bijection π_2 defined as follows

$$1 \to 2,\ 2 \to 3,\ 3 \to 6,\ 4 \to 8,\ 5 \to 4,\ 6 \to 5,\ 7 \to 1,\ 8 \to 7$$

is 2-canonical for the matrix of Example 1. The sets $\pi_2[i,k]$ are as follows $\pi[2,1] = \{1,2\}$, $\pi[2,2] = \{3,4\}$, $\pi[2,3] = \{5,6\}$, $\pi[2,4] = \{7,8\}$;

the partitions P_k^i, Q_k^i obtained by applying Construction 2.1 are

$P_1^2 = \{1\}$, $P_2^2 = \{4\}$, $P_3^2 = \{6\}$, $P_4^2 = \{7\}$;

$Q_1^2 = \{2\}$, $Q_2^2 = \{3\}$, $Q_3^2 = \{5\}$, $Q_4^2 = \{8\}$;

and the bijections ψ_k are

$\psi_1 : 1 \to 1,\ 2 \to 2;\ \psi_2: 4 \to 3,\ 3 \to 4;\ \psi_3 \to 6 \to 5,\ 5 \to 6;\ \psi_4: 7 \to 7,\ 8 \to 8$

The bijection π_3 obtained by step 2 of Construction 2.1 is then

$\pi_3: 1 \to 2,\ 2 \to 3,\ 3 \to 8,\ 4 \to 6,\ 5 \to 5,\ 6 \to 4,\ 7 \to 1,\ 8 \to 7.$

Lemma 2 Let π_i be i-canonical, and let π_{i+1} be the bijection obtained by Construction 2.1. Then, π_{i+1} is (i+1)-canonical.

Proof. It is not difficult to see that if π_i is i-canonical, then π_{i+1} is i-canonical. Hence, it sufficies to prove that $\pi_{i+1}^{-1}(x) < \pi_{i+1}^{-1}(y)$ if $f^{i+1}(x) < f^{i+1}(y)$. Let $x \in Z_{k'}^{i+1}$, $y \in Z_{k''}^{i+1}$ with $k' < k''$. Two cases may arise. Case 1 ($k' = k''-1 = 2k-1$). In this case, $\pi_i^{-1}(x) \in P_k^i$ and $\pi_i^{-1}(y) \in Q_k^i$; by Step 1(b) of Construction 3.1 $\pi_{i+1}^{-1}(x) = \psi_k(\pi_i^{-1}(x)) < \psi_k(\pi_i^{-1}(y)) = \pi_{i+1}^{-1}(y)$. Case 2. Let $c' = [k'/2]$ and $c'' = [k''/2]$, obviously $c' < c''$. By definition (1), $x \in Z_{c'}^i$, and $y \in Z_{c''}^i$; since π_{i+1} is i-canonical, then $\pi_{i+1}^{-1}(x) < \pi_{i+1}^{-1}(y)$.[ ]

Let $\phi_k^i(j)$ be the Boolean function ($1 \le i < p$, $1 \le k \le 2^i$, $1 \le j \le N$)

$$\phi_k^i(j) = \begin{cases} \bigvee_{x \in Z_k^i} B[x,j] & \text{if } Z^i \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

where $\vee$ denotes the Boolean OR.

Lemma 3 Let π be p-canonical. Then

$$\phi_k^p(j) = \begin{cases} \bigvee_{s=\ell_\pi[p,k]}^{h_\pi[p,k]} B[\pi(s), j] & \text{if } Z_k^p \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

for $1 \leq k \leq 2^i$, $1 \leq j \leq N$.

Proof. By Lemma 1.

Lemma 4 For $1 \leq i \leq p$, $1 \leq k \leq 2^i$, $1 \leq j \leq N$

$$\phi_k^i(j) = (\phi_{2k-1}^{i+1}(j) \vee \phi_{2k}^{i+1}(j))$$

Proof. By observing that $Z_k^i = Z_{2k-1}^{i+1} \nu Z_{2k}^{i+1}$. [ ]

Theorem 1 $C[i,j] = 1$ iff $\bigvee_{r=1}^{2^{i-1}} \phi_{2r-1}^i(j) = 1$ $(1 \leq i \leq p,\ 1 \leq j \leq N)$

Proof. By (1), it follows that $A[i,\ell] = 1$ iff $r \in \{1,\ldots,2^{i-1}\}: \ell \in Z_{2r-1}^i$.

Then, by (4) it follows that

$C[i,j] = 1$ iff $\exists\, \ell \in \{1,\ldots,N : A[i,\ell] = B[\ell,j] = 1$ iff

$\exists\, r \in \{1,\ldots,2^{i-1}\}, \ell \in \{1,\ldots,N\} : \ell \in Z_{2r-1}^i \wedge B[\ell,j] = 1$ iff

$\exists\, r \in \{1,\ldots,2^{i-1}\} : \bigvee_{x \in Z_{2r-1}} B[x,j] = 1$ iff

$\exists\, r \in \{1,\ldots,2^{i-1}\} : \phi_{2r-1}^i(j) = 1$ iff

$\bigvee_{r=1}^{2^{i-1}} \phi_{2r-1}^i(j) = 1$.

3. AN ALGORITHM FOR (p,N,N) WITH $p \leq \lceil \log N \rceil$

Theorem 1 shows that to compute the entries of the j-th column of the product matrix $C = A \times B$ it is sufficient to compute the OR of the Boolean functions $\phi_k^i(j)$'s over the appropriate indices. Lemma 4 gives a method for computing the $\phi_k^i(j)$'s once the $\phi_k^{i+1}(j)$'s have been computed; Lemma 3 shows how to compute the starting values $\phi_k^p(j)$'s once a p-canonical bijection π and the values $h_\pi[p,k]$'s and $\ell_\pi[p,k]$'s are known. Finally, Construction 2.1 provides a method for determining a $(i+1)$-canonical bijection once a i-canonical bijection is known. Observe that, by Lemma 1, each set $\pi_i[i,k]$ is uniquely defined by the values $\ell_{\pi_i}[i,k]$ and $h_{\pi_i}[i,k]$ $(1 \leq k \leq 2^i)$.

The above considerations lead to the following algorithm for computing the product $C = A \times B$.

Algorithm 3.1

Step 1. (Computation of a p-canonical bijection: Initialization)

Set π_o to be the identity function on $\{1,\ldots,N\}$; set $\ell_{\pi_o}[0,1] := 1$, $h_{\pi_o}[0,1] := N$, $i := 1$.

Step 2. (Computation of a p-canonical bijection: Iteration)

a) Compute an i-canonical bijection π_i from π_{i-1} using Construction 2.1. (Note: the set $\pi_i[i-1,k]$ is uniquely defined by the values $\ell_{\pi_{i-1}}[i-1,k]$ and $h_{\pi_{i-1}}[i-1,k]$, $1 \leq k \leq 2^{i-1}$);

b) Compute the values $\ell_{\pi_i}[i,k]$ and $h_{\pi_i}[i,k]$ $(1 \leq k \leq 2^i)$;

c) Set $i := i + 1$; if $i \leq p$, repeat Step 2.

Step 3. (Computation of product matrix: Initialization)

Set j: = 1

Step 4. (Computation of j-th column of product matrix: Initialization)

a) Set i: = p;

b) Compute $\phi_k^p(j)$ $(1 \leq k \leq 2^p)$ using Lemma 3.

Step 5. (Computation of j-th column of product matrix: Iteration)

a) Compute C[i,j] using Theorem 1;

b) Compute $\phi_k^{i-1}(j)$ $(1 \leq k \leq 2^{i-1})$ using Lemma 4;

c) Set i: = i-1; if $i \geq 1$, repeat this step.

Step 6. (Computation of product matrix: Iteration)

Set j: = j+1; if $j \leq N$, goto Step 4.

Theorem 2 Algorithm 3.1 correctly computes the product C = A x B within finite time.

Proof. Finiteness is obvious. Correctness follows from Theorem 1 and Lemmas 2-4.[ ]

Lemma 5 Given an i-canonical bijection π_i, and the values $\ell_{\pi_i}[i,k]$ and $h_{\pi_i}[i,k]$ for each non empty Z_k^i $(1 \leq k \leq 2^i)$, an (i+1)-canonical bijection can be computed by Construction 2.1 using O(N log N) bit operations.

Proof: Each non-empty $\pi[i,k]$ is composed (see Lemma 1) of the consecutive integers between $\ell_{\pi_i}[i,k]$ and $b_{\pi_i}[i,k]$. To construct P_k^i and Q_k^i, it is sufficient to test each entry $A[i+1,\pi_i(x)]$ $(\ell_{\pi_i}[i,k] \leq x \leq h_{\pi_i}[i,k])$; if the entry is zero, then x is added to Q_k^i; otherwise, it is added to P_k^i. Note that since $x \leq h_{\pi_i}[i,k] \leq N$, the addition of x to either P_k^i or Q_k^i would require $\lceil \log N \rceil$ bit operations. Hence, the construction of P_k^i and Q_k^i requires a total of $|\pi_i[i,k]|(\lceil \log N \rceil + 1)$ bit operations; since ϕ_k can be

constructed by simply examining the sets P_k^i and Q_k^i (in that order), and assigning consecutive integers between $\ell_{\pi_i}[i,k]$ and $h_{\pi_i}[i,k]$, an additional $|\pi_i[i,k]|\lceil \log N \rceil$ bit operations suffice. Assuming that testing on whether $\pi_i[i,k]$ is empty can be done in $\lceil \log N \rceil$ bit operations (note: this can be obviously achieved by setting $\ell_{\pi_i}[i,k] = h_{\pi_i}[i,k] = 0$ for empty $\pi_i[i,k]$), the execution of Step 2 requires at most

$$\sum_{k=1}^{2^i} |\pi_i[i,k]|(2\lceil \log N \rceil + 1) + \lceil \log N \rceil$$

bit operations. Since the $\pi_i[i,k]$'s are all disjoint and $\bigcup_k \pi_i[i,k] = \{1,\ldots,N\}$, then $O(N \log N)$ bit operations are required in total by Step 1. Step 2 can be obviously performed in an additional $O(N \log N)$ bit operations; hence, the Lemma holds. [ ]

Lemma 6 The total execution of Step 2 of Algorithm 3.1 requires $O(p\ N \log N)$ bit operations.

Proof: By Lemma 5, the i-th iteration of Step 2(a) requires $O(N \log N)$ bit operations. Since the values $\ell_{\pi_i}[i,k]$ and $h_{\pi_i}[i,k]$ (in Step 2(b)) can be computed from the sets P_k^{i-1} and Q_k^{i-1} in $|\pi_{i-1}[i-1,k]| \log N$ bit operations, and since $\sum_k |\pi_{i-1}[i-1,k]| = N$ it follows that the i-th iteration of Step 2(b) requires $O(N \log N)$ bit operations. Since Step 2(a) and 2(b) are performed P times, the Lemma holds. [ ]

Lemma 7 For a given j, $1 \leq j \leq N$, the total execution of Step 5 of Algorithm 3.1 requires $O(N)$ bit operations.

Proof: The i-th execution of Step 5(a) (for a fixed j) requires at most $2^{i-1}-1$ bit operations (see Theorem 1); Step 5(b) requires 2^{i-1} bit operations, one for each $\phi_k^{i-1}(j)$ (see Lemma 4). Since Steps 5(a) and 5(b) are executed for $i\in\{0,\ldots,p\}$, $\sum_{i=0}^{p} (2^{i-1}-1 + 2^{i-1}) = 2^{p+1}-p-1 = 0(N)$ bit operations are required in total. [ ]

Theorem 3 Algorithm 3.1 computes the product $C = A \times B$ using at most $0(N^2)$ bit operations.

Proof: By Lemmas 6 and 7, and by observing that each execution of Step 4 requires $0(N)$ bit operations (see Lemma 4), and that Steps 4 and 5 are executed N times while Step 2 is executed only once. [ ]

Algorithm 2.1 can be implemented so to employ only $0(N \log N)$ bits of additional storage. The basic ideas of this implementation are the following.

An i-canonical bijection π_i can be obviously stored in an integer array of N elements, where the j-th entry contains $\pi_i(j)$. Since $\pi_{i+1}(j)$ is constructed only afterwards, the storage area for $\pi_i(j)$ can be reused for $\pi_{i+1}(j)$. Since $P_k^i \vee Q_k^i \subseteq \{1,\ldots,N\}$ and $P_k^i \cap Q_k^i = \emptyset$, each of the auxiliary sets P_k^i and Q_k^i can obviously be stored in an integer array of N elements; furthermore, the same storage area can be used to store P_{k+1}^i and Q_{k+1}^i once

ψ_k has been computed. Since the mapping ψ_k is only a 'fragment' of π_{i+1} (see Step 2 of Construction 2.1), it is implicitly contained in π_{i+1}. To store all values $\ell_{\pi_i}[i,k]$ and $h_{\pi_i}[i,k]$, two integer arrays of size $2N$ each suffice (recall, there are $2^{p+1}-1 \leq 2N-1$ possible ℓ's, and the same number of h's). Since all these integers are in $\{1,\ldots,N\}$, $O(N \log N)$ bits of additional storage in total suffice to implement Steps 1 and 2 of Algorithm 2.1.

For a fixed j, $1 \leq j \leq N$, all the $2^{p+1}-1$ values $\phi_k^i(j)$'s can be stored in a Boolean array of $2N$ elements (recall, $2^{p+1}-1 \leq 2N-1$); the same array can obviously be employed for successive j's. Hence, $O(N)$ bits of additional storage in total are sufficient to implement the remaining Steps 3-6 of Algorithm 2.1. Therefore

Theorem 4 Algorithm 2.1 can be implemented so to use at most $O(N \log N)$ bits of additional space.

3. AN ALGORITHM FOR (N,N,N)

Consider now the product of two $N \times N$ Boolean matrices, A and B. Let $m = \lfloor \log N \rfloor$ and $k = \lceil N/m \rceil$. Partition matrix A into matrices $A_1, A_2, \ldots, A_k$, where matrix A_i $(1 \leq i < k)$ consists of rows $m(i-1)+1,\ldots,mi$ of A, and A_k consists of the remaining rows. The product $C_i = A_i \times B$ $(1 \leq i < k)$ is a $m \times N$ matrix which consists of the rows $m(i-1)+1,\ldots,mi$ of the product matrix $C = A \times B$; $C_k = A_k \times B$ consists of the remaining rows. Therefore, to compute C, it sufficies to compute the products $A_i \times B$ $(1 \leq i \leq k)$ using the matrix multiplication algorithm proposed in Section 2. Since $m = \lfloor \log N \rfloor$, the computation of each product will take $O(N^2)$ bit operations; hence $O(N^3/\log N)$ bit operations will be performed in total. The additional storage required when multiplying $A_i \times B$ can obviously be employed in the computation of $A_{i+1} \times B$;

hence, by Theorem 4, $O(N \log N)$ bits of additional storage suffice to compute the product matrix C.

4. CONCLUSIONS

A new algorithm for computing the product of two arbitrary $N \times N$ Boolean matrices has been presented. It has been shown that the proposed algorithm requires $O(N^3/\log N)$ bit operations (thus, achieving the Four Russians' bound) but only $O(N \log N)$ bits of additional storage. It should be pointed out that unlike the Four Russians' Method, the proposed algorithm cannot be directly executed on a vector computer.

Acknowledgements

The authors wish to thank Mike Atkinson for the helpful discussions.

REFERENCES

[1] Arlazarof, V.L., Dinic, E.A., Kronrod, M.A., and Faradzev, I.A.: On economical construction of the transitive closure of a directed graph. Doki. Akad.Nauk. SSR 194. 487-488. (1970),

[2] Santoro, N.: Four $O(N^2)$ multiplication methods for sparse and dense Boolean matrices. In: Proc. 10th Conf. Numerical Mathematics and Computing. Winnipeg, Manitoba, 241-253, 1980.

CARLETON UNIVERSITY

School of Computer Science

BIBLIOGRAPHY OF SCS TECHNICAL REPORTS

SCS-TR-1 — **THE DESIGN OF CP-6 PASCAL**
Jim des Rivieres and Wilf R. LaLonde, June 1982.

SCS-TR-2 — **SINGLE PRODUCTION ELIMINATION IN LR(1) PARSERS: A SYNTHESIS**
Wilf R. LaLonde, June 1982.

SCS-TR-3 — **A FLEXIBLE COMPILER STRUCTURE THAT ALLOWS DYNAMIC PHASE ORDERING**
Wilf R. LaLonde and Jim des Rivieres, June 1982.

SCS-TR-4 — **A PRACTICAL LONGEST COMMON SUBSEQUENCE ALGORITHM FOR TEXT COLLATION**
Jim des Rivieres, June 1982.

SCS-TR-5 — **A SCHOOL BUS ROUTING AND SCHEDULING PROBLEM**
Wolfgang Lindenberg, Frantisek Fiala, July 1982.

SCS-TR-6 — **ROUTING WITHOUT ROUTING TABLES**
Nicola Santoro, Ramez Khatib, July 1982.

SCS-TR-7 — **CONCURRENCY CONTROL IN LARGE COMPUTER NETWORKS**
Nicola Santoro, Hasan Ural, July 1982.

SCS-TR-8 Out of Print — **ORDER STATISTICS ON DISTRIBUTED SETS**
Nicola Santoro, Jeffrey B. Sidney, July 1982.

SCS-TR-9 — **OLIGARCHICAL CONTROL OF DISTRIBUTED PROCESSING SYSTEMS**
Moshe Krieger, Nicola Santoro, August 1982.

SCS-TR-10 — **COMMUNICATION BOUNDS FOR SELECTION IN DISTRIBUTED SETS**
Nicola Santoro, Jeffrey B. Sidney, September 1982.

SCS-TR-11 — **SIMPLE TECHNIQUE FOR CONVERTING FROM A PASCAL SHOP TO A C SHOP**
Wilf R. LaLonde, John R. Pugh, November 1982.

SCS-TR-12 — **EFFICIENT ABSTRACT IMPLEMENTATIONS FOR RELATIONAL DATA STRUCTURES**
Nicola Santoro, December 1982.

SCS-TR-13 — **ON THE MESSAGE COMPLEXITY OF DISTRIBUTED PROBLEMS**
Nicola Santoro, December 1982.

SCS-TR-14 **A COMMON BASIS FOR SIMILARITY MEASURES INVOLVING TWO STRINGS**
R. L. Kashyap and B. J. Oommen, January 1983.

SCS-TR-15 **SIMILARITY MEASURES FOR SETS OF STRINGS**
R. L. Kashyap and B. J. Oommen, January 1983.

SCS-TR-16 **THE NOISY SUBSTRING MATCHING PROBLEM**
R. L. Kashyap and B. J. Oommen, January 1983.

SCS-TR-17 **DISTRIBUTED ELECTION IN A CIRCLE WITHOUT A GLOBAL SENSE OF ORIENTATION**
E. Korach, D. Rotem, N. Santoro, January 1983.

SCS-TR-18 **A GEOMETRICAL APPROACH TO POLYGONAL DISSIMILARITY AND THE CLASSIFICATION OF CLOSED BOUNDARIES**
R. L. Kashyap and B. J. Oommen, January 1983.

SCS-TR-19 **SCALE PRESERVING SMOOTHING OF POLYGONS**
R. L. Kashyap and B. J. Oommen, January 1983.

SCS-TR-20 **NOT-QUITE-LINEAR RANDOM ACCESS MEMORIES**
Jim des Rivieres, Wilf LaLonde and Mike Dixon,
August 1982, Revised March 1, 1983.

SCS-TR-21 **SHOUT ECHO SELECTION IN DISTRIBUTED FILES**
D. Rotem, N. Santoro, J. B. Sidney, March 1983.

SCS-TR-22 **DISTRIBUTED RANKING**
E. Korach, D. Rotem, N. Santoro, March 1983.

SCS-TR-23 **A REDUCTION TECHNIQUE FOR SELECTION IN DISTRIBUTED FILES : I**
N. Santoro, J. B. Sidney, April 1983.

SCS-TR-24 **LEARNING AUTOMATA POSSESSING ERGODICITY OF THE MEAN : THE TWO ACTION CASE**
M. A. L. Thathachar and B. J. Oommen, May 1983.

SCS-TR-25 **ACTORS - THE STAGE IS SET**
John R. Pugh, June 1983.

SCS-TR-26 **ON THE ESSENTIAL EQUIVALENCE OF TWO FAMILIES OF LEARNING AUTOMATA**
M. A. L. Thathachar and B. J. Oommen, May 1983.

SCS-TR-27 **GENERALIZED KRYLOV AUTOMATA AND THEIR APPLICABILITY TO LEARNING IN NONSTATIONARY ENVIROMENTS**
B. J. Oommen, June 1983.

SCS-TR-28 **ACTOR SYSTEMS: SELECTED FEATURES**
Wilf R. LaLonde, July 1983.

SCS-TR-29 **ANOTHER ADDENDUM TO KRONECKER'S THEORY OF PENCILS**
M. D. Atkinson, August 1983.

SCS-TR-30 **SOME TECHNIQUES FOR GROUP CHARACTER REDUCTION**
M. D. Atkinson and R. A. Hassan, August 1983.

SCS-TR-31 **AN OPTIMAL ALGORITHM FOR GEOMETRICAL CONGRUENCE**
M. D. Atkinson, August 1983.

SCS-TR-32 **MULTI-ACTION LEARNING AUTOMATA POSSESSING ERGODICITY OF THE MEAN**
B. J. Oommen and M. A. L. Thathachar, August 1983.

SCS-TR-33 **FIBONACCI GRAPHS, CYCLIC PERMUTATIONS AND EXTREMAL POINTS**
N. Santoro and J. Urrutia, December 1983

SCS-TR-34 **DISTRIBUTED SORTING**
D. Rotem, N. Santoro, and J. B. Sidney, December 1983

SCS-TR-35 **A REDUCTION TECHNIQUE FOR SELECTION IN DISTRIBUTED FILES: II**
N. Santoro, M. Scheutzow, and J. B. Sidney, December 1983

SCS-TR-36 **THE ASYMPTOTIC OPTIMALITY OF DISCRETIZED LINEAR REWARD-INACTION LEARNING AUTOMATA**
B. J. Oommen and Eldon Hansen, January 1984

SCS-TR-37 **GEOMETRIC CONTAINMENT IS NOT REDUCIBLE TO PARETO DOMINANCE**
N. Santoro, J. B. Sidney, S. J. Sidney, and J. Urrutia, January 1984

SCS-TR-38 **AN IMPROVED AGLORITHM FOR BOOLEAN MATRIX MULTIPLICATION**
N. Santoro and J. Urrutia, January 1984

SCS-TR-39 **CONTAINMENT OF ELEMENTARY GEOMETRIC OBJECTS**
J. Sack, N. Santoro and J. Urrutia, February 1984

SCS-TR-40 **SADE: A PROGRAMMING ENVIRONMENT FOR DESIGNING AND TESTING SYSTOLIC ALGORITHMS**
J. P. Corriveau and N. Santoro, February 1984

SCS-TR-41 **INTERSECTION GRAPHS, {B }-ORIENTABLE GRAPHS AND PROPER CIRCULAR ARC GRAPHS**
Jorge Urrutia, February 1984

SCS-TR-42 **MINIMUM DECOMPOSITIONS OF POLYGONAL OBJECTS**
J. Mark Keil and Jorg-R. Sack, March 1984

SCS-TR-43 **AN ALGORITHM FOR MERGING HEAPS**
Jorg-R. Sack and Thomas Strothotte, March 1984

SCS-TR-44 **A DIGITAL HASHING SCHEME FOR DYNAMIC MULTIATTRIBUTE FILES**
E. J. Otoo, March 1984

SCS-TR-45 **SYMMETRIC INDEX MAINTENANCE USING MULTIDIMENSIONAL LINEAR HASHING**
E. J. Otoo, March 1984

SCS-TR-46 **A MAPPING FUNCTION FOR THE DIRECTORY OF A MULTIDIMENSIONSAL EXTENDIBLE HASHING**
E. J. Otoo, March 1984

SCS-TR-47 **TRANSLATING POLYGON IN THE PLANE**
Jorg-R. Sack, March 1984

SCS-TR-48 **CONSTRAINED STRING EDITING**
J. Oommen, May 1984

SCS-TR-49 **O(N) ELECTION ALGORITHMS IN COMPLETE NETWORKS WITH GLOBAL SENSE OF ORIENTATION**
Jorg Sack, Nicola Santoro, Jorge Urrutia, May 1984

SCS-TR-50 **THE DESIGN OF A PROGRAM EDITOR BASED ON CONSTRAINTS**
Christopher A. Carter and Wilf R. LaLonde, May 1984

SCS-TR-51 **DISCRETIZED LINEAR INACTION-PENALTY LEARNING AUTOMATA**
B. J. Oommen and Eldon Hansen, May 1984

SCS-TR-52 **SENSE OF DIRECTION, TOPOLOGICAL AWARENESS AND COMMUNICATION COMPLEXITY**
Nicola Santoro, May 1984

SCS-TR-53 **OPTIMAL LIST ORGANIZING STRATEGY WHICH USES STOCHASTIC MOVE-TO-FRONT OPERATIONS**
B. J. Oommen, June 1984

SCS-TR-54 **RECTINLINEAR COMPUTATIONAL GEOMETRY**
J. Sack, June 1984

SCS-TR-55 **AN EFFICIENT, IMPLICIT DOUBLE-ENDED PRIORITY QUEUE**
M. D. Atkinson, Jorg Sack, Niccla Santoro, T. Strothotte, July 1984

SCS-TR-56 **DYNAMIC MULTIPAGING: A MULTIDIMENSIONAL STRUCTURE FOR FAST ASSOCIATIVE SEARCHING**
E. Otoo, T. H. Merrett

SCS-TR-57 **SPECIALIZATION, GENERALIZATION AND INHERITANCE**
Wilf R. LaLonde, John R. Pugh, August 1984

SCS-TR-58 **COMPUTER ACCESS METHODS FOR EXTENDIBLE ARRAYS OF VARYING DIMENSIONS**
E. Otoo, August 1984.

SCS-TR-59 **AREA-EFFICIENT EMBEDDINGS OF TREES**
J. P. Corriveau, Nicola Santoro, August 1984.

SCS-TR-60 **UNIQUELY COLOURABLE m-DICHROMATIC ORIENTED GRAPHS**
V. Neumann-Lara, N. Santoro, J. Urrutia, August 1984.

SCS-TR-61 **ANALYSIS OF A DISTRIBUTED ALGORITHMS FOR EXTREMA FINDING IN A RING**
D. Rotem, E. Korach and N. Santoro, August 1984.

SCS-TR-62 **ON ZIGZAG PERMUTATIONS AND COMPARISONS OF ADJACENT ELEMENTS**
M. D. Atkinson, October 1984

SCS-TR-63 **SETS OF INTEGERS WITH DISTINCT DIFFERENCES**
M. D. Atkinson, A. Hassenklover, October 1984.

SCS-TR-64 **TEACHING FIFTH GENERATION COMPUTING: THE IMPORTANCE OF SMALL TALK**
Wilf R. LaLonde, Dave A. Thomas, John R. Pugh, October 1984.

SCS-TR-65 **AN EXTREMELY FAST MINIMUM SPANNING CIRCLE ALGORITHM**
B. J. Oommen, October, 1984.

SCS-TR-66 **ON THE FUTILITY OF ARBITRARILY INCREASING MEMORY CAPABILITIES OF STOCHASTIC LEARNING AUTOMATA**
B. J. Oommen, October, 1984.

SCS-TR-67 **HEAPS IN HEAPS**
T. Strothotte, J.-R. Sack, November 1984

SCS-TR-68 **PARTIAL ORDERS AND COMPARISON PROBLEMS**
M. D. Atkinson, November 1984.