Printing Requisition / Graphic Services

Dept. No. 85982

Title or Description: CS-80-25 "Four 0(n**2) ..."

Date: May 13, 1980 **Date Required:** May 15, 1980 **Account:** 126-6177-41

Signature: E. Huang **Signing Authority:** J. Tompkins

Department: Computer Science **Room:** M&C 5100B **Phone:** 3402

Delivery: ☒ Mail ☐ Pick-up ☐ Via Stores ☐ Other

1. Please complete unshaded areas on form as applicable. (4-part no carbon required).
2. Distribute copies as follows: White, Canary and Pink—Printing, Arts Library or applicable Copy Centre Goldenrod—**Retain.**
3. On completion of order, pink copy will be returned with printed material. Canary copy will be costed and returned to requisitioner, **Retain as a record of your charges.**
4. Please direct enquiries, quoting requisition number, to Printing/Graphic Services, Extension 3451.

Reproduction Requirements: ☑ Offset ☐ Signs/Repro's ☐ Xerox

Number of Pages: 17 **Number of Copies:** ~~50~~ 50

Type of Paper Stock: ☑ Bond ☐ Book ☐ Cover ☐ Bristol ☐ Supplied

Paper Size: ☑ 8½ x 11 ☐ 8½ x 14 ☐ 11 x 17

Paper Colour: ☑ White ☐ Other

Ink: ☐ Black

Printing: ☑ 1 Side ☐ 2 Sides

Numbering: to

Binding/Finishing Operations: ☑ Collating ☐ Corner Stitching ☐ 3 Ring ☐ Tape ☐ Plastic Ring ☐ Perforating

Folding: 3 STAPLES LEFT SIDE

Finished Size:

Cutting: Finished Size

Special Instructions:
Please staple in covers - 3 staples.. Thanks.

Cost: Time/Materials	Fun.	Prod. Ch.	Prod. Op. No.	Mins.	Total
Signs/Repro's	1				
Camera	2				
Correcting & Masking Negatives	3				
Platemaking	4				
Printing	5				
Binding	6			20	3.20
				20	2.20
Sub. Total Time					
Sub. Total Materials					
Prov. Tax					
Total					

Film Qty Size

Plates Qty Size & Type

Paper Qty Size

Plastic Rings Qty Size

Outside Services

Graphic Services
May 68 482-2

Printing Requisition / Graphic Services 3 4

Dept. No. 53636

Title or Description
Four O(n2) Multiplication Methods for Sparse and Dense Boolean Matrices***

1. Please complete unshaded areas on form as applicable. (4 part no carbon required).
2. Distribute copies as follows: White, Canary and Pink—Printing, Arts Library or applicable Copy Centre Goldenrod—**Retain.**
3. On completion of order, pink copy will be returned with printed material. Canary copy will be costed and returned to requisitioner, **Retain as a record of your charges.**
4. Please direct enquiries, quoting requisition number, to Printing/Graphic Services, Extension 3451.

Date	Date Required	Account
Jan. 8/82	Jan. 11/82	126-4100-01

Signature	Signing Authority
	Freda Bruner

Department	Room	Phone	Delivery
Computer Science	5179	3143	☒ Mail ☐ Pick-up ☐ Via Stores ☐ Other

Reproduction Requirements
☑ Offset ☐ Signs/Repro's ☐ Xerox

Number of Pages 17 **Number of Copies** 1

Type of Paper Stock
☑ Bond ☐ Book ☐ Cover ☐ Bristol ☐ Supplied

Paper Size
☑ 8½ x 11 ☐ 8½ x 14 ☐ 11 x 17

Paper Colour
☑ White ☐ Other

Ink
☐ Black

Printing
☑ 1 Side ☐ 2 Sides

Numbering
to

Binding/Finishing Operations
☑ Collating ☑ Corner Stitching ☐ 3 Ring ☐ Tape ☐ Plastic Ring ☐ Perforating

Folding
Finished Size

Cutting
Finished Size

Special Instructions

Cost: Time/Materials

		Fun.	Prod. Cl.	Un. Prod. No.	Opr. Mins.	Total
Signs/Repro's	1					
Camera	2					
Correcting & Masking Negatives	3					
Platemaking	4					
Printing	5					
Bindery	6					
Sub. Total Time						
Sub. Total Materials						
Prov. Tax						
Total						

Film	Plates	
Qty Size	Qty Size & Type	
Paper	**Plastic Rings**	
Qty Size	Qty Size	
Outside Services		

GRAPHIC SERVICES SEPT.80 482-2

Copyright: I hereby agree to assume all responsibility and liability for any infringement of copyrights and/or patent rights which may arise from the processing of, and reproduction of, any of the materials herein requested. I further agree to indemnify and hold blameless the University of Waterloo from any liability which may arise from said processing or reproducing. I also acknowledge that materials processed on this requisition are for educational use only.

FOUR O(n**2) MULTIPLICATION METHODS FOR SPARSE AND DENSE BOOLEAN MATRICES*

CS-80-25

Nicola Santoro

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

* This work has been partially supported by the Natural Sciences and Engineering Research Council.

Abstract

In this paper, we present four algorithms to evaluate the product of sparse and dense Boolean matrices. The proposed algorithms determine the Boolean product of two n*n matrices in O(n**2) operations whenever one of the matrices is either sparse or dense. Finally, the multiplication methods for sparse Boolean matrices are extended to work with general sparse matrices with the same time complexity.

Key words and phrases: Boolean matrix multiplication, sparse matrices, dense matrices, bipartite graphs.

1. INTRODUCTION

With the notable exception of the "four Russians' algorithm" [2], Boolean multiplication methods have been usually extensions of general matrix multiplication techniques. Namely, asymptotically fast Boolean algorithms [1,3] have been modeled on the well known asymptotically fast algorithms for general matrix multiplication [6,8]. Because of the versatility of Boolean matrices and of their capability to model different problems, to find efficient and practical algorithms (i.e. algorithms that are simpler to implement than, and/or outperform the asymptotically fast methods for special classes of matrices) becames a relevant problem.

In this paper, we present four efficient and practical multiplication methods particularly suited for sparse and dense Boolean matrices. In fact, with the proposed methods we can evaluate the product of two $n \times n$ Boolean matrices in $O(n^{**}2)$ operations whenever one of the two matrices is sparse or dense. In the following sections, we describe the framework and present the algorithms for the multiplication of sparse and dense Boolean matrices. Finally, we show how to modify the multiplication algorithms for sparse Boolean matrices to work with general sparse matrices with the same time complexity. All the proposed algorithms use a data structure similar to the classical structures emploied by the indexing techniques for sparse and dense matrices.

2. THE FRAMEWORK

There is a natural correspondence between Boolean matrices and binary relations on finite sets or, equivalently, finite directed graphs. Thus, we can represent a n×n Boolean matrix B by the sets R(B,i), i=1,...,n, where R(B,i)={j|B(i,j)=1}. The sets R(B,i) are called adjacency lists of B. Analogously we can define the predecessor lists P(B,i), i=1,...,n, where P(B,i)={k|B(k,i)=1}. Let m(B) be the number of non-zero elements in B; then, matrix B is said to be sparse if m(B)=O(n) [7].

Let us remark that many indexing techniques for sparse matrices use storage schemas very similar to the adjacency or the predecessor lists, e.g. the indexing with row and column designators, with row vector indicators and column designation, with row vector and column index vector, etc. [7]. We will not further specify the implementation characteristics of the information structure used by the proposed algorithms; in fact, whatever of the above data representations is effectively used, this alters the complexity of the matrix multiplication only by a constant factor [4]. Finally, let us remark that the adjacency and the predecessor lists, if not available, can be easily constructed from B in time O(n**2).

Before presenting the algorithms, let us introduce some additional terminology that will be used in the following sections. As mentioned before, we can uniquely associate to each

Boolean matrix B a directed graph G(B)=(V(B),E(B)), where there is a vertex in V(B) for each row and one for each column in B, and there is an edge from the vertex associated with row i to the vertex associated with column j if and only if B(i,j)=1. Let h denote the isomorphic mapping between the set of rows and columns and the set of vertices. It is easy to see that G(B) is a bipartite graph and that {VR(B),VC(B)} is the bipartition [10], where VR(B) and VC(B) are the set of the images of the rows and columns of B, respectively. If B(i,j)=1, then h(j) is said to be reachable from h(i) in G(B). Intuitively, R(B,i) represents the set of the vertices reachable from vertex h(i), and P(B,i) represents the set of vertices from which vertex h(i) is reachable.

In this paper we are interested in the evaluation of the product of two n*n Boolean matrices B=B1*B2. Without loss of generality, let us assume VC(B1)=VR(B2). Obviously, VR(B)=VR(B1) and VC(B)=VC(B2).

3. ALGORITHMS FOR SPARSE BOOLEAN MATRICES

When evaluating the Boolean product B=B1*B2, if B1 or B2 is sparse, then we can take advantage of this fact in the computation of the product matrix B. In this section, we consider two cases, depending on whether B1 or B2 is sparse, and present a simple and efficient algorithm for each case. Obviously, if

both B1 and B2 are sparse, then any of the proposed algorithms can be used.

Case 1 (B2 is sparse)

The algorithm AS to evaluate the product B=B1*B2 when B2 is sparse is described in the appendix.

In order to discuss the correctness and the complexity of procedure AS, let us observe the following relation:

(1) $B(i,j)=1 \iff j \in R(B,i) \iff \exists\, k \in R(B1,i)$ such that $j \in R(B2,k) \iff h(j)$ is reachable in G(B2) from at least one vertex h(k) with $k \in R(B1,i)$

where <==> means "if and only if". It is easy to see that, for each $i \in [1,n]$, procedure AS performs the equivalent of a depth first search [9] in G(B2) from all vertices h(k) reachable from h(i); thus, procedure AS satisfies relation (1). Let us now analyze the complexity of the algorithm. For a given $i \in [1,n]$, the algorithm requires $\max\{n,e(i)\}$ operations,

where $e(i)= \sum_{k \in R(B1,i)} |R(B,k)|$.

Since $e(i) \leq m(B2)$ and since this search is done for all $i \in [1,n]$, in the worst case we need $n \cdot m(B2)$ operations. That is,

Property 1 Procedure AS determines the Boolean product of a n×n matrix by a n×n sparse matrix in O(n**2) operations.

Let us remark that a procedure similar to AS has been used to produce a fast expected time algorithm for Boolean matrix multiplication [5].

Case 2 (B1 is sparse) :

Using the predecessor lists P(B1,i) and P(B2,i), we can write a procedure SA, similar to AS, to evaluate B when B1 is sparse; the algorithm is described in the appendix.

Property 2 Procedure SA determines the Boolean product of a n×n sparse matrix by a n×n matrix in O(n**2).

To prove the above property, let us observe that, if vertex h(j) is reachable from vertex h(i) in G(B), then h(i) is reachable from h(j) in $G(B^T)$, where B^T is the transpose of B. Therefore, relation (1) can be re-expressed as follows:

(2) B(i,j)=1 <==> i∈P(B,j) <==> node h(i) is reachable in $G(B^T)$ from h(j) <==> node h(i) is reachable in $G(B^T)$ from at least a node h(k) with k∈P(B2,j) <==> ∃k∈P(B2,j) i∈P(B1,k)

For each j∈[1,n], procedure SA performs the equivalent of a depth first search in $G(B1^T)$ starting from all vertices reachable in $G(B2^T)$ from h(j); thus, it obviously satisfies relation (2). The analysis of the complexity of procedure SA follows the same lines as the analysis of procedure AS.

4. ALGORITHMS FOR DENSE BOOLEAN MATRICES

Given a n*n Boolean matrix B, its complement $\bar{B}$ is the n*n Boolean matrix defined by

$$\bar{B}(i,j)=1 \quad <==> \quad B(i,j)=0$$

We shall say that B is (very) dense if its complement is sparse. Since $\bar{B}$ has $m(\bar{B})=n**2-m(B)$ non-zero entries, where m(B) is the number of non-zero elements in B, then we can say that B is dense if $m(\bar{B})=O(n)$.

Let us consider the product B=B1*B2 when B1 or B2 is dense. In order to reduce the complexity of the multiplication, we must take advantage of both the density of one of the matrices, and the fact that we are dealing with Boolean matrices. Namely we determine the zero entries of the product matrix instead of the non-zero entries, and use the sets $R(\bar{B},i)$ and $P(\bar{B},i)$ instead of R(B,i) and P(B,i). Again, we present two algorithms, depending on B1 or B2 being a dense matrix. Obviously, if both B1 and B2 are dense, then any of the proposed algorithms can be used. The algorithms (procedures AD and DA)

are described in the appendix.

Property 3 Procedure AD determines the Boolean product of a n*n matrix by a dense n*n matrix in O(n**2) operations.

Let us first prove that procedure AD computes exactly the product B. To do this, it is sufficient to observe that relation (1) is equivalent to

(3) $B(i,j)=0 \iff j \notin R(B,i) \iff \neg(\exists k \in R(B1,i) \quad j \in R(B2,k))$
$\iff \forall k \in R(B1,i) \quad j \notin (B2,k) \iff \forall k \in R(B1,i) \quad j \in R(\overline{B2},k)$

That is, we must set B(i,j) to zero if and only if h(j) is not reachable in G(B2) from any of the vertices reachable from h(i) in G(B1). It is easy to see that h(j) is not reachable from any $h(k) \in V(B1)$ if and only if $mark(j)=|R(B1,i)|$. To prove the order complexity, let us observe that given $i \in [1,n]$, we search the graph $G(\overline{B2})$ starting from all the vertices reachable from h(i) in G(B1). Since the whole search will not take more than m(B2) steps and since we repeat this process for i=1,...,n, then the entire procedure will require at most $n*m(\overline{B2})$ steps. Since B2 is dense, then the above property holds.

Let us now consider the case of B1 being dense.

Property 4 Procedure DA determines the Boolean product of a dense Boolean matrix by a Boolean matrix in O(n**2)

The proof follows the lines of the proof of Property 3, and by observing that relation (2) can be re-written as

(4) $B(i,j)=0 \iff i \notin P(B,j) \iff \neg(\exists k \in P(B2,j) \quad i \in P(B1,k))$
$\iff \forall k \in P(B2,j) \quad i \notin P(B1,k) \iff \forall k \in P(B2,j) \quad i \in P(\overline{B1},k)$

5. EXTENSION TO NON-BOOLEAN MATRICES

The algorithms for the multiplication of sparse Boolean matrices, AS and SA, can be easily modified to work with general sparse matrices with the same time complexity. Unfortunately, this property does not hold for the dense Boolean matrix multiplication methods AD and DA. We will now describe how to extend algorithms AS and SA to work with general matrices.

Let us consider the product M=M1*M2, where M1 and M2 are n*n matrices and at least one of them is sparse; i.e. it has O(n) non-zero elements. We can easily extend to non-Boolean matrices the terminology introduced in the previous sections. The adjacency list R(M,i) is the set of couples (index,value) defined as follows: $R(M,i)=\{(index,value) \mid M(i,index)=value \neq 0\}$. That is, $p=(j,x) \in R(M,i)$ iff $M(i,j)=x \neq 0$; in this case we will

use the notation value [p]=x and index [p]=j. Analogously, P(M,j)={(index,value)|M(index,j)=value/0}.

We can still represent M in graph form: G(M)=(V(M),L(M),E(M)) is the directed edge-labelled bipartite graph where V(M)=VR VC is the set of the isomorphic images of the rows and columns of M, L(M) is the set of labels, and E(M) VR VC L(M) is such that there is an edge e=(h(i),h(j),x) from vertex h(i) to vertex h(j) labelled x if and only if M(i,j)=x. The generalization of AS and SA to non Boolean matrices is now straightforward. As usual, we present two algorithms depending on whether M1 or M2 is sparse; the algorithms (procedures GAS and GSA) are described in the appendix.

Let us analyze procedure GAS. When evaluating the entry

$$M(i,j) = \sum_{k \in [1,n]} M1(i,k) * M2(k,j). \tag{5}$$

we do not need to consider in the summation the contribution for those k such that M1(i,k)=0 or M2(k,j)=0. Let Nij={k∈[1,n]|M1(i,k)≠0 and M2(k,j)≠0}; then, relation (5) is equivalent to

$$M(i,j) = \sum_{k \in Nij} M1(i,k) * M2(k,j). \tag{6}$$

The set Nij represents exactly those vertices in VC(M1) that are reachable from h(i) in G(M1) and from which h(j) is reachable in G(M2) (recall VC(M1)=VR(M2)). It is easy to see that

$$\sum_{j \in [1,n]} |Nij| \leqslant m(M2).$$

That is, for a given i∈[1,n], we compute at most m(M2) multiplications and m(M2) additions. Since this is repeated for i=1,...,n, and since M2 is sparse (i.e. m(M2)=O(n)), then O(n**2) operations are needed. The analysis of procedure GSA follows in a similar way. This leads to the following

Property 5 The product of two n×n matrices can be evaluated in O(n**2) whenever one of the two matrices is sparse.

6. CONCLUSIONS

In this paper we have presented four efficient multiplication algorithms for sparse and dense Boolean matrices. The algorithms work in time O(n**2) and use a data structure similar to many classical structures for sparse and dense matrices (in case of dense matrices, we obviously store the zero entries). Extending some of the above algorithms, two O(n**2) multiplication methods for non-Boolean sparse matrices have been presented. Finally, all the proposed algorithms will yield O(n f(n)) time bounds if sparse is defined to mean O(f(n)) non-zero entries.

REFERENCES

[1] Adleman L., Booth K.S., Preparata F.P., Ruzzo W.L. : Improved time and space bounds for Boolean matrix multiplication. Acta Informatica 11, 61-70 (1978).

[2] Arlazarov V.L., Dinic E.A., Kronrod M.A., Faradzev I.A. : On economical construction of the transitive closure of a directed graph. Dokl. Akad. Nausk SSSR 194, 487-488 (1970).

[3] Fisher P.C., Meyer A. : Boolean matrix multiplication and transitive closure. IEEE Conf. Record of 12th SWAT, 129-131 (1971).

[4] Mac Veigh D.T. : Effect of data representation on cost of sparse matrices operations. Acta Informatica 7, 361-394 (1972).

[5] O'Neil P.E., O'Neil E.J. : A fast expected time algorithm for Boolean matrix multiplication and transitive closure. Information and Control 22, 132-138 (1973).

[6] Pan V. : Strassen's algorithm is not optimal. Proc. of 19th Annual Symp. on Foundations of Computer Science 1978.

[7] Pooch V.W., Nieder A. : A survey of indexing techniques for sparse matrices. Computing Surveys 5, 109-133 (1973).

[8] Strassen V. : Gaussian elimination is not optimal. Numer. Math. 13, 354-356 (1969).

[9] Tarjan R.E. : Depth first search and linear graph algorithms. SIAM J. Comput. 1, 146,160 (1972).

[10] Tutte W.T. : Connectivity in graphs, p. 67. Birkenhead: Oxford University Press 1966.

APPENDIX

```
procedure AS   /*  B2 is sparse   */
begin
integer mark(n);
for i=1 until n do
     begin
     for j=1 until n do mark(j):=B(i,j):=0;
     for all k∈R(B1,i) do
          for all j∈R(B2,k) do
               if mark(j)=0 then
                    begin
                    mark(j):=1;
                    B(i,j):=1;
                    end;
     end;
end;
```

```
procedure SA  /*  B1 is sparse  */
begin
integer mark(n);
for j=1 until n do
     begin
     for i=1 until n do mark(i):=B(i,j):=0;
     for all k∈P(B2,j) do
          for all i∈(P(B1,k) do
               if mark(i)=0 then
                    begin
                    mark(i):=1; B(i,j):=1;
                    end;
     end;
end;
```

```
procedure AD     /*  B2 is dense */
begin
integer mark(n);
for i=1 until n do
     begin
     for j=1 until n do
          begin
          mark(j):=0;
          B(i,j):=1;
          end;
     for all k∈R(B1,i) do
          for all j∈R(B2,k) do mark(j):=+ 1;
     for j=1 until n do
          if mark(j)=|R(B1,i)| then B(i,j):=0;
     end;
end
```

```
procedure DA   /*  B1 is dense  */
begin
integer mark(n);
for j=1 until n do
     begin
     for i=1 until n do
          begin
          mark(i):=0; B(i,j):=1;
          end;
     for all k∈P(B2,j) do
          for all i∈P(B1,k) do mark(i):=+1;
     for i=1 until n do
          if mark(i)=|P(B2,j)| then B(i,j)=0;
     end;
end;
```

```
procedure GAS   /*   M2 is sparse  */
begin
for i=1 until n do
     begin
     for j=1 until n do M(i,j):=0;
     for all p∈R(M1,i) do
          for all q∈R(M2, index [p]) do
               M(i, index [q]):=M(i, index [q]) + value  [p]  *
               value [q];
     end;
end;
```

```
procedure GSA   /*  M1 is sparse  */
begin
for j=1 until n do
     begin
     for i=1 until n do M(i,j):=0;
     for all p∈P(M2,j) do
          for all q∈P(M1, index [p]) do
               M( index [q],j):=M( index [q],j)+  value  [p]  *
               value [q];
     end;
end;
```