MATHEMATICS OF COMPUTATION
VOLUME 38, NUMBER 157
JANUARY 1982

A Method for Finding Permanents of 0, 1 Matrices

By Ralph Kallman

Abstract. Certain row operations are used in a method for computing permanents of 0, 1 matrices. Machine execution times for this method are compared with those for the Ryser and Nijenhuis-Wilf algorithms.

1. Terminology. Since the rows of a 0, 1 matrix may be regarded as the characteristic functions of sets, e.g., (1101) for $\{x_1, x_2, x_4\}$, we shall use the standard names and symbols for set operations and relations for the corresponding ones on the rows. Thus, the intersection of two rows (1101), (1110) is (1100) since $\{x_1, x_2, x_4\} \cap \{x_1, x_2, x_3\} = \{x_1, x_2\}$. Similarly $(1101) \cup (1110) = (1111)$, $(1101) \setminus (1110) = (0001)$, $(1100) \subseteq (1110)$, etc. A universal row is defined to be one equal to the union of all rows. The number of 1's in row R is denoted $\# R$; if $\# R = 1$, then R is called a singleton row; if $\# R = 0$, then R is a null or zero row. All matrices are 0, 1. A permutation of matrix M is a selection of a single 1 from each row with no column duplications. Thus, per M is the count of such permutations.

2. The Method. We use the following splitting and row removal lemmas.

LEMMA 1. *Let $M(S, T)$ denote a matrix where all rows remain fixed except the jth and kth rows which are S, T, respectively. Then*

$$\operatorname{per} M(S, T) = \operatorname{per} M(S \cup T, S \cap T) + \operatorname{per} M(S \setminus T, T \setminus S).$$

Proof. Since T is the disjoint union of $T \cap S$ with $T \setminus S$, $\operatorname{per} M(S, T) = \operatorname{per} M(S, T \cap S) + \operatorname{per} M(S, T \setminus S)$. Likewise

$$\operatorname{per} M(S, T \setminus S) = \operatorname{per} M(S \cap T, T \setminus S) + \operatorname{per} M(S \setminus T, T \setminus S).$$

Then after a row interchange,

$$\begin{aligned}&\operatorname{per} M(S, T)\\ &\quad = (\operatorname{per} M(S, T \cap S) + \operatorname{per} M(T \setminus S, T \cap S)) + \operatorname{per} M(S \setminus T, T \setminus S)\\ &\quad = \operatorname{per} M(S \cup T, S \cap T) + \operatorname{per} M(S \setminus T, T \setminus S). \quad \square\end{aligned}$$

Here is an illustration of Lemma 1 which involves the last two rows.

$$\operatorname{per}\begin{bmatrix}1 & 0 & 1 & 1\\ 1 & 1 & 0 & 1\\ 1 & 1 & 1 & 0\end{bmatrix} = \operatorname{per}\begin{bmatrix}1 & 0 & 1 & 1\\ 1 & 1 & 1 & 1\\ 1 & 1 & 0 & 0\end{bmatrix} + \operatorname{per}\begin{bmatrix}1 & 0 & 1 & 1\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}.$$

Henceforth, $M(R_1, \ldots, R_m)$ denotes an $m \times n$ matrix with rows R_i.

Received August 4, 1980; revised February 6, 1981.
1980 *Mathematics Subject Classification*. Primary 05-04, 05A05.
Key words and phrases. Permanent.

LEMMA 2. *Suppose* $M(R_1, \ldots, R_m)$ *has* $R_m \cap R_i = \varnothing$ *or* $R_m \subseteq R_i$ *for all* i *and* S *is a singleton row,* $S \subseteq R_m$. *Then*

$$\operatorname{per} M(R_1, \ldots, R_m) = \# R_m \cdot \operatorname{per} M(R_1 \setminus S, \ldots, R_{m-1} \setminus S).$$

Proof. If R_m is a singleton, the result is clear. If not, let $S_1, S_2, \ldots, S_t$ be disjoint singleton subsets of R_m where $t = \# R_m$. Then

$$\operatorname{per} M(R_1, \ldots, R_{m-1}, S_j) = \operatorname{per} M(R_1 \setminus S_j, \ldots, R_{m-1} \setminus S_j).$$

By our hypotheses, each of $M(R_1 \setminus S_j, \ldots, R_{m-1} \setminus S_j)$ can be obtained from any other by a column interchange; thus they all have the same permanent. Since R_m is the disjoint union of the S_j, $\operatorname{per} M(R_1, \ldots, R_m) = \sum_{j=1}^{t} \operatorname{per} M(R_1, \ldots, R_{m-1}, S_j)$ and the desired result follows. □

Of course, if any row is null, then per $M = 0$.

LEMMA 3. *If* R_m *is a universal row of* $M(R_1, \ldots, R_m)$, *then* $\operatorname{per} M(R_1, \ldots, R_m) = (\# R_m - m + 1) \cdot \operatorname{per} M(R_1, \ldots, R_{m-1})$.

Proof. Select a permutation from $M(R_1, \ldots, R_{m-1})$ first. Then $\# R_m - m + 1$ choices remain for R_m. □

Of course, the universal need not be the last row since row interchanges may be made.

Description of Method. Our method is to expand per $M = a_1$ per $M_1 + a_2$ per $M_2 + \ldots$ so that rows may be removed from summand matrices; when four or fewer rows remain for a term, a direct evaluation is made.

Let $\rho(t)$ denote the condition $R_i \cap R_m = \varnothing$ or $R_m \subseteq R_i$, $i \geqslant t$, for $M(R_1, \ldots, R_m)$. Note that $\rho(1)$ is the hypothesis for Lemma 2 and $\rho(m)$ is always true. We create terms satisfying $\rho(1)$ by an iterative procedure. If $R_m = \varnothing$, a zero evaluation is made immediately, and if $\# R_m = 1$, then $\rho(1)$ holds.

Suppose $t > 1$ and $\rho(t)$ holds. We do the first permissible operation in the following list to create $\rho(t-1)$.

1. If $R_m \subseteq R_{t-1}$ or $R_m \cap R_{t-1} = \varnothing$, then $\rho(t-1)$ already holds; if R_{t-1} is a universal, it is removed by Lemma 3.

2. If $R_{t-1} \subseteq R_m$, then interchanging the rows creates condition $\rho(t-1)$; however, if $\# R_{t-1} = 1$, it is removed by Lemma 1.

3. If $R_{t-1} \setminus R_m \neq \varnothing$ and $R_m \setminus R_{t-1} \neq \varnothing$, then Lemma 1 is applied. Both terms thus created satisfy $\rho(t-1)$. One term is stored and operations continue on the second.

The preceding procedure is repeated until $\rho(1)$ is established for a term; then Lemma 2 is used to remove a row. When a term has four or fewer rows, the permanent is evaluated using the formulas:

$$\operatorname{per} M(R_1) = \# R_1, \tag{1}$$

$$\operatorname{per} M(R_1, R_2) = \# R_1 \# R_2 - \#(R_1 \cap R_2), \tag{2}$$

$$\begin{aligned}\operatorname{per} M(R_1, R_2, R_3) = {} & \# R_1 \# R_2 \# R_3 - \# R_1 \#(R_2 \cap R_3) - \# R_2 \#(R_1 \cap R_3) \\ & - \# R_3 \#(R_1 \cap R_2) + 2\#(R_1 \cap R_2 \cap R_3).\end{aligned} \tag{3}$$

The formula for the $4 \times n$ case is too lengthy to list here; see [1, p. 8].

Proof of (3). Now, $\# R_3 \cdot \operatorname{per} M(R_1, R_2)$ counts the permutations of $M(R_1, R_2, R_3)$ plus additional ways of selecting a 1 from each row such that the 1 from R_3 is column duplicated exactly once from R_1 or R_2. Subtracting the number of these latter ways gives

$$\begin{aligned}\operatorname{per} M(R_1, R_2, R_3) = \# R_3 \cdot \operatorname{per} M(R_1, R_2) &- \operatorname{per} M(R_1 \cap R_3, R_2) \\ &- \operatorname{per} M(R_1, R_2 \cap R_3);\end{aligned}$$

this expression may be further broken down using (1) and (2) and reassembled into formula (3). □

Although these formulas involve subtractions, the numbers are too small to cause cancellation of digits from subtractions (on the DEC-10 machine). As terms are evaluated they are added to a cumulative sum which eventually becomes the answer.

3. Test Examples. In the following examples, K denotes the method of Section 2 above, R denotes H. J. Ryser's inclusion-exclusion formula [3, p. 26], and NW denotes the Nijenhuis-Wilf adaptation of Method R [2, p. 224]. Method NW is available for square matrices only. Neither R nor NW are restricted to 0, 1 matrices. We compare the methods by giving machine execution times which, of course, depend on the speed of the object machine and the computer programs which implement the algorithms.

Example 1. *Derangements*. The matrices are $n \times n$ with zeros on the main diagonal and ones elsewhere. Table I gives machine execution times.

TABLE I

	Time, seconds		
n	K	R	NW
5	.002	.004	.004
10	.017	.18	.15
15	.20	8.1	6.54
20	2.37	341.a	271.a

a. Answer incorrect because of cancellation of digits in subtractions.

Example 2. *Random Matrices*. The pseudo-random number generator was used to create $m \times 15$ matrices. Table II and Table III give machine execution times for probability $p = .75$ and $p = .25$, respectively, of a 1 in a given position. Two examples of each size were created.

TABLE II
Example 2, $p = .75$

m	Time, seconds K	Time, seconds R	Value
5	.003	.48	74983
5	.004	.48	59052
10	.56	5.3	830203830
10	2.9	5.3	338176115
15	477.	8.1	3622020199
15	316.	8.1	7260574617

TABLE III
Example 2, $p = .25$

m	Time, seconds K	Time, seconds R	Value
5	.001	.48	338
5	.002	.48	34
10	.037	5.3	2513
10	.35	5.3	32628
15	.032	8.1	74
15	.001	8.1	0

4. The Computer Programs. The programs are coded in Fortran; the compiler used was FORTRAN 10 (OPT). Execution was on a DEC10. Method K is machine-dependent since it involves a machine language subroutine for counting bits. Copies of programs for K and R are available in mimeographed form [1] and NW is found in [2, p. 224].

5. Summary. Execution times for method K depend on the one's density and the structure of the matrix while times for R and NW are constant for matrices of fixed dimensions. Cancellation of digits due to subtractions cannot occur in method K. A useful application of K would be if it were required to determine whether or not a 0, 1 matrix has permanent 0; the program could be easily modified to include an exit when the first nonzero term is encountered. This paper incorporates the several valuable suggestions of the referee.

Department of Mathematical Sciences
Ball State University
Muncie, Indiana 47306

1. R. KALLMAN, *Computer Programs for Evaluating Permanents of* 0, 1 *Matrices*, Dept. of Math. Techn. Report., Ball State Univ., 1980.

2. A. NIJENHUIS & H. S. WILF, *Combinatorial Algorithms*, 2nd ed., Academic Press, New York, 1978.

3. H. J. RYSER, *Combinatorial Mathematics*, Carus Math. Monograph No. 14, Math. Assoc. Amer., 1963.