PROBABILISTIC ALGORITHMS AND STRAIGHT-LINE PROGRAMS FOR SOME RANK DECISION PROBLEMS *

Oscar H. IBARRA, Shlomo MORAN and Louis E. ROSIER
Computer Science Department, 136 Lind Hall, Institute of Technology, University of Minnesota, Minneapolis, MN 55455, U.S.A.

Received 17 February 1981; revised version received 8 June 1981

Probabilistic algorithms, polynomial time, straight-line programs, rank, fast matrix multiplication

1. Introduction

In this paper, we consider the rank decision problem for matrices with polynomial entries. Specifically, the problem is to determine for an arbitrary m × n matrix A with polynomial entries (where the polynomials are represented by arithmetic expressions with arbitrary parenthesization using addition, subtraction, multiplication, and exponentiation to positive integer constants) and a positive integer r whether rank(A) < r. [1] This problem occurs in several applications. (For a discussion on applications see [7,8].) At present, there is no known polynomial time algorithm to solve this problem. In [7] (see also [8]), a special case was shown to be probabilistically decidable in polynomial time. (This means that there is an algorithm which uses a random number generator to decide the question in polynomial time. If the answer is yes, then rank(A) < r with probability of error less than 1/2. If the algorithm answers no, then rank(A) ⩾ r with certainty. Note that by running the algorithm k times, a probability of error less than $1/2^k$ can be obtained.) However, the probabilistic algorithm in [7] works only for m × n matrices A whose polynomial entries over variables $x_1, x_2, ..., x_t$ satisfy the following conditions:

(1) m and n are bounded by t;

(2) the polynomial entries are in standard form (i.e. each polynomial is a sum of terms of the form $cx_1^{i_1}x_2^{i_2} \cdots x_t^{i_t}$, where c is an integer constant) and their degrees (i.e. $i_1 + \cdots + i_t$) are bounded by a polynomial in t.

In this paper, we show that the rank decision problem for the full class of matrices with polynomial entries (i.e., *no* restrictions (1) and (2)) can be solved probabilistically in polynomial time. (This result generalizes to the class of matrices whose entries are functions computable by straight-line programs.) Our proof involves a reduction of the rank decision problem to the zero-equivalence problem for straight-line programs (i.e. deciding if an arbitrary program over the instruction set {z ← 1, z ← x + y, z ← x − y, z ← x * y} outputs 0 for all integer inputs). It is known that the zero-equivalence problem for {z ← 1, z ← x + y, z ← x − y, z ← x * y}-programs is probabilistically decidable in polynomial time [2]. As a first step in the reduction, we show that the rank decision problem for matrices whose entries are real numbers can be solved by a straight-line program whose length (i.e. number of instructions) is polynomial in m, n, and r. As we shall see in section 2, it is possible to write efficient straight-line programs to solve the rank decision problem for matrices with real entries. The programs are of independent interest since they are about the simplest that one can write, and yet their complexities are not that much worse than the best known

* This research was supported in part by NSF Grant MCS78-01736.

[1] rank(A) is the maximal integer k such that A contains a k × k nonsingular submatrix.

0020-0190/81/0000–0000/$02.50 © 1981 North-Holland

algorithms. For example, we show that there is a straight-line program over $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$ which when given an m × n real matrix A, outputs 0 if and only if rank(A) < r. Moreover, the length of the program is $O(mn^{\beta-1} + \sqrt{r}n^{\beta} + rn^2)$, where β is such that multiplication of two n × n real matrices can be implemented on a straight-line program of length $O(n^{\beta})$. (The best upper bound on β known today is due to Coppersmith and Winograd, and is 2.495 ... .)

The straight-line programs are constructed using two well-known techniques, namely, Gaussian elimination and fast matrix multiplication. Of particular interest is a result (in section 2) which shows that no straight-line program which implements Gaussian elimination can be constructed for real matrices. On the other hand, for integer matrices, Gaussian elimination can easily be implemented by a straight-line program of length $O(mn^2)$. When 'forward if' statements are allowed, computing the determinant, rank, etc. (even for real matrices) can be done by programs of length $O(n^{\beta})$ [1]. We should note that in the case of matrix multiplication, the fast algorithms can be coded by straight-line programs.

We conclude this section with some comments concerning variations of the rank decision problem which are not solvable by straight-line programs.

Remarks. Let L be the instruction set $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y, z \leftarrow x/y\}$.

(1) No straight-line program over L can compute the rank of a matrix. This follows from the fact that programs over L compute rational functions of the inputs, but the rank is not a rational function.

(2) Let m, n, and r be positive integers such that $m \geqslant n > r$. No straight-line program over L can determine given an m × n matrix A whether rank(A) ≠ r. For suppose such a program exists. Call it F. Then for m × n matrices of the form

$$A = \left[\begin{array}{cccccc} 1 & & & & & \\ & \ddots & & \mathbf{0} & & \\ & & 1 & & & \\ & & & x & & \\ & & & & 0 & \\ & \mathbf{0} & & & \ddots & \\ & & & & & 0 \end{array}\right] \begin{array}{l} \updownarrow\, r+1 \\ \\ \\ \\ \\ \\ \\ \end{array}$$

F outputs p(x)/q(x) for some polynomials p(x) and q(x). Since F outputs 0 if and only if rank(A) ≠ r, p(x)/q(x) = 0 for all x ≠ 0. This implies that p(x) is identically equal to 0, a contradiction, since for x = 0, the value of p(x)/q(x) must be nonzero. Similarly, no straight-line program can determine whether rank(A) = r.

2. Straight-line programs based on Gaussian elimination and fast matrix multiplication

In this section, we show how two well-known techniques: Gaussian elimination and fast matrix multiplication, can be used to construct efficient straight-line programs to solve the rank decision problem. Our first result concerns a special case: the singularity problem. We show that Gaussian elimination can be used to construct an $O(mn^2)$ straight-line program to decide singularity of integer matrices. By first multiplying the matrix on the left by its transpose (using fast matrix multiplication), the time bound can be reduced to $O(mn^{\beta-1} + n^3)$. If m = n and it is known that the matrix is nonsingular, the determinant can also be computed by the program using only one division instruction.

Theorem 2.1. Let m and n be positive integers with $m \geqslant n$. We can construct a straight-line program F over $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$ with input variables x_{ij} $(1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n)$ and output variables y and z having the following properties:

(a) If an m × n integer matrix $A = (a_{ij})$ is input to F (i.e. x_{ij} is set to a_{ij}), then at the end of F, y = 0 if and only if A is singular (i.e. rank(A) < n).

(b) Length(F) = number of instruction in F = $O(mn^2)$.

(c) If m = n and y ≠ 0, then det(A) = y/z.

Proof. The implementation of the Gaussian elimination technique to diagonalize the matrix is straightforward provided we can guarantee that the pivot element at stage i (for $1 \leqslant i \leqslant n$) is nonzero if A is nonsingular. This can be accomplished by first applying the following operations to row i at the beginning of stage i:

$$X_i \leftarrow (2x_{ii} + 1)X_i + \sum_{k \neq i} x_{ki}X_k\,, \qquad (*)$$

X_i and X_k denote rows i and k, respectively, and the notation $X_i \leftarrow uX_i + vX_k$ is an abbreviation for the n operations $\{x_{ij} \leftarrow ux_{ij} + vx_{kj} \mid 1 \leqslant j \leqslant n\}$. Note that $(*)$ replaces x_{ii} by $(2x_{ii} + 1)x_{ii} + \Sigma_{k \neq i} x_{ki}^2$. Since A is an integer matrix, $(2x_{ii} + 1)x_{ii} \geqslant 0$. Hence, the pivot element (i.e. the value of x_{ii}) after $(*)$ is nonzero if A is nonsingular. Initially, z is set to 1, and each time an instruction of the form $X_i \leftarrow uX_i$ is executed in the program, we set $z \leftarrow uz$. The variable y is set to the product of the diagonal elements of the matrix after stage n.

Remark. The (Gaussian elimination) technique in Theorem 2.1 can be used to obtain an $O(mn^2 r)$ straight-line program to solve the rank decision problem. However, a faster algorithm which works for real matrices is given in Theorem 2.3.

The Gaussian elimination technique of Theorem 2.1 does not generalize to real matrices. In fact, the pivoting step which replaces row A_i by a linear combination of the rows so that at the end of the process the pivot element $a_{ii} \neq 0$ cannot be executed (for real matrices) by a straight-line program, even when it is known that the input matrix is nonsingular.

Theorem 2.2. Let $n \geqslant 2$. There is no straight-line program F over $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y, z \leftarrow x/y, X_1 \leftarrow yX_1, X_1 \leftarrow X_1 + yX_i \mid z$ is a noninput variable, $i \neq 1\}$ [2] which when given any nonsingular $n \times n$ real matrix $A = (a_{ij})$ can transform A into $A' = (a'_{ij})$ such that

(1) A' is nonsingular;

(2) $a'_{11} > 0$.

Proof. Clearly, it is sufficient to prove the theorem for n = 2. For each 4-tuple of real numbers $\alpha = (x, y, z, w)$, let

$$A(\alpha) = \begin{bmatrix} x & y \\ z & w \end{bmatrix}.$$

Then it is easy to show that the existence of a program F satisfying (1) and (2) is equivalent to showing the existence of a matrix

$$R(\alpha) = \begin{bmatrix} R_1(\alpha) & R_2(\alpha) \\ 0 & 1 \end{bmatrix},$$

where R_1 and R_2 are rational functions in the variables x, y, z, w such that for each $\alpha = (x, y, z, w)$, if $A(\alpha)$ is nonsingular, then the following holds for $A'(\alpha) = R(\alpha)A(\alpha)$:

(3) $A'(\alpha)$ is nonsingular (and hence $R(\alpha)$ is nonsingular);

(4) $a'_{11}(\alpha) > 0$, where $a'_{11}(\alpha)$ is the entry of $A'(\alpha)$ in the 1st row, 1st column.

Assume that such an R exists, and let $\alpha_0 = (1, 0, 0, 1)$. Then

$$A(\alpha_0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

and

$$R(\alpha_0)\, A(\alpha_0) = R(\alpha_0) = \begin{bmatrix} R_1(\alpha_0) & R_2(\alpha_0) \\ 0 & 1 \end{bmatrix}.$$

Hence, by (4), $R_1(\alpha_0) > 0$. Now let $\alpha_1 = (1, 1, -1, 1)$, $\alpha_2 = (-1, 1, -1, -1)$, and $\alpha_3 = (-1, 0, 0, -1)$. Then

$$A(\alpha_1) = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}, \qquad A(\alpha_2) = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix},$$

and

$$A(\alpha_3) = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Let C be the continuous curve in R_4 consisting of the straight-line segments connecting α_0 to α_1, α_1 to α_2, and α_2 to α_3. It is easy to check that for each point $\alpha = (x, y, z, w)$ in C, $xw - yz > 0$, and hence $A(\alpha)$ is nonsingular. Therefore, by (3), $R(\alpha)$ is nonsingular. This implies that $R_1(\alpha)$ is defined and $R_1(\alpha) \neq 0$ for all α in C. Since R_1 is continuous and $R_1(\alpha_0) > 0$, $R_1(\alpha) > 0$ for all α in C. In particular, $R_1(\alpha_3) > 0$. On the otherhand,

$$A'(\alpha_3) = R(\alpha_3)\, A(\alpha_3) = -R(\alpha_3) =$$

$$= \begin{bmatrix} -R_1(\alpha_3) & -R_2(\alpha_3) \\ 0 & -1 \end{bmatrix}$$

which implies, by (4), that $R_1(\alpha_3) < 0$, a contradiction.

We shall show, using a different technique, that the rank decision problem can be solved by a straight-line program whose length is $O(mn^{\beta-1} + \sqrt{m}n^{\beta} + m^2)$. Thus, the singularity problem is solvable by an $O(mn^{\beta-1} + n^{\beta+0.5} + n^3)$ straight-line program. (Note that this bound is at least $O(mn^{\beta-1} + n^3)$.)

[2] Recall that $X_1 \leftarrow yX_1$ is an abbreviation for $\{x_{1j} \leftarrow yx_{1j} \mid 1 \leqslant j \leqslant n\}$.

We will need the following facts concerning positive semi-definite matrices [3] (see [4]):

(1) For each matrix P, P^TP is positive semi-definite. (P^T is the transpose of P.)

(2) If A is a positive semi-definite matrix, then all its characteristic roots are real and nonnegative. Moreover, A is similar to a diagonal matrix, D, whose diagonal elements $d_1, ..., d_n$ are the characteristic roots of A.

Theorem 2.3. For any positive integers m, n, and r with $m \geqslant n \geqslant r$, we can construct a $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$-program F which determines for any real $m \times n$ matrix A whether $\text{rank}(P) < r$. Moreover, $\text{length}(F) = O(mn^{\beta-1} + \sqrt{m}n^\beta + m^2)$.

Proof. It follows from (2) above that the rank of a positive semi-definite matrix A is equal to the number of its nonzero characteristic roots. Let $f_A(\lambda) = \lambda^n + c_1\lambda^{n-1} + \cdots + c_{n-1}\lambda + c_n$ be the characteristic polynomial of A. Since all the characteristic roots of A are nonnegative, $c_i \neq 0$ if and only if A has at least i nonzero characteristic roots. (Note that $c_i = (-1)^i\Sigma_{j_1...j_i} d_{j_1} \cdots d_{j_i}$, where $j_1, ..., j_i$ range over all the i permutations of $\{1, 2, ..., n\}$.) Thus, to decide if $\text{rank}(P) < r$, we can first compute $A = P^TP$ in $O(mn^{\beta-1})$ arithmetic operations, and then compute c_r. Then $\text{rank}(P) < r$ if and only if $c_r = 0$.

Now in [3] (p. 560), a fast algorithm to compute the coefficients of the characteristic polynomial $f_A(\lambda) = \det(\lambda I - A) = \lambda^n + c_1\lambda^{n-1} + \cdots + c_{n-1}\lambda + c_n$ of any $n \times n$ matrix A is given. One can easily check that the algorithm in [3] can be used to obtain, for any given $r \leqslant n$, a $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$-program of length $O(\sqrt{m}n^\beta + m^2)$ to compute $n!c_r$. [4]

Corollary 2.1. For any positive integer n, we can construct a $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$-program of length $O(n^{\beta+0.5} + n^3)$ which computes $n!\det(A)$, where A is any $n \times n$ real matrix. (Note that if the construct $z \leftarrow 1/n!$ or $z \leftarrow x/y$ is allowed, then the program can compute the determinant.)

Remarks. An $O(n^\beta)$ straight-line program over $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y, z \leftarrow x/y\}$ which computes the determinant of an $n \times n$ real matrix is described in [5]. However, the program is *not* total in that it does not work for all matrices, even for those which are nonsingular. At present, Corollary 2.1 is our best result for computing the determinant of *any* real matrix. For integer matrices, an $O(n^\beta)$ straight-line program to decide singularity exists. (This was communicated to us by an anonymous referee.) However, it can be shown (using the technique of Theorem 2.2) that the program, which uses recursive applications of fast matrix multiplication, cannot be generalized to real matrices.

3. Matrices with polynomial (rational) expression entries

Definition. A *polynomial expression* (*p.e.*) is any expression which can be obtained using rules (1)–(3) below:

(1) Any integer constant (positive, negative, zero) is a p.e.

(2) Any variable is a p.e.

(3) If α and β are p.e.'s, then so are $-\alpha, \alpha + \beta, \alpha - \beta, \alpha * \beta, \alpha^k$, where k is any positive integer. (Parentheses may be used to avoid ambiguity.)

A p.e. is in *standard form* if it is a sum of terms of the form $cx_1^{i_1} \cdots x_t^{i_t}$, where c is any integer (positive, negative, zero), $t \geqslant 0$, and $i_1, ..., i_t$ are positive integers. The *size* of a p.e. is the length of its representation.

Examples. -5, $x_1 + 2x_2^2x_3 - x_3^{50}$, $(((2x_1^2x_2 - 3x_3 + x_4)^2 - x_2)^{12}x_1^3 - 5)^{50}$ are p.e.'s. The first two are in standard form.

Let A be a matrix whose entries are polynomial expressions. By definition, rank(A) is the maximal integer r such that there is an $r \times r$ submatrix of A whose determinant is a nonzero polynomial. The following proposition is obvious:

Proposition 3.1. Let A be a matrix whose entries are p.e.'s over variables $x_1, ..., x_t$. Then rank(A) over the reals (i.e. $x_1, ..., x_t$ can assume real values) = rank(A) over the rationals = rank(A) over the integers.

[3] A matrix, A, is positive semi-definite if for each column vector, x, $x^TAx \geqslant 0$.

[4] Without the construct $z \leftarrow 1/n!$ or $z \leftarrow x/y$, we can only compute $n!c_r$. (Note that for a given n, $1/n!$ is a fixed rational constant.)

Theorem 3.1. Let A be an m × n matrix whose entries are polynomial expressions over variables $x_1, ..., x_t$. Let r be a positive integer, $r \leq \min\{m, n\}$. We can construct in polynomial time (in the size= length of the representation of A) a $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$-program F with input variables $x_1, ..., x_t$ and output variable y such that F outputs 0 for all integer inputs $x_1, ..., x_t$ if and only if rank(A) < r.

Proof. By Theorem 2.3 and Proposition 3.1, we need only show that any polynomial expression over the integers can be computed by a straight-line program. Clearly, it is sufficient to show that for any positive integer k and any variable x, k and x^k are computable by straight-line programs of length O(log k). This can be done by standard techniques (see, e.g., [3, section 4.6.3]). The idea is to use the binary representation of k to double and add log k times.

We can generalize Theorem 3.1 to matrices whose entries are total rational expressions.

Definition. A *rational expression* (*r.e.*) is any expression which can be obtained using rules (1)–(3) below:

(1) Any rational constant is a r.e.

(2) Any variable is a r.e.

(3) If α and β are r.e.'s, then so are $-\alpha$, $\alpha + \beta$, $\alpha - \beta$, $\alpha * \beta$, α/β, α^k, where k is any positive integer.

A r.e. is *total* (over a given input domain) if the value of the expression is defined for all values of the variables (in the domain). Note that every p.e. is a total r.e.

Theorem 3.2. Let A be an m × n matrix whose entries are total rational expressions over variables $x_1, ..., x_t$. Let r be a positive integer, $r \leq \min\{m, n\}$. We can construct in polynomial time a $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$-program F with input variables $x_1, ..., x_t$ and output variable y such that F outputs 0 for all integer inputs $x_1, ..., x_t$ if and only if rank(A) < r.

Proof. The construction of F is similar to that of Theorem 3.1. However, since F cannot do division, F uses two variables for each original variable to keep track of the values of the numerator and denominator. In particular, there are intermediate variables $x_1', ..., x_t', y'$ which correspond to the denominators of $x_1, ..., x_t, y$. All intermediate variables are initially set to 1. Each arithmetic operation is computed by parts, e.g., $z \leftarrow u + v$ is coded

$$z \leftarrow u * v' + v * u', \qquad z' \leftarrow u' * v'.$$

Similarly, $z \leftarrow u/v$ is coded

$$z \leftarrow u * v', \qquad z' \leftarrow v * u'.$$

Clearly, F can be constructed so that F outputs y = 0 for all integer inputs $x_1, ..., x_t$ if and only if rank(A) < r.

4. Polynomial time probabilistically solvable problems

Theorem 3.2 can be used to show that the rank of any matrix with total r.e. entries is probabilistically computable in polynomial time. This is a corollary to the following theorem.

Theorem 4.1. Determining for an arbitrary m × n matrix A with total r.e. entries and a positive integer $r \leq \min\{m, n\}$ whether rank(A) < r is probabilistically decidable in polynomial time.

Proof. In a recent paper [2], it is shown that the zero-equivalence problem [5] for $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y\}$-programs over the integers (or over the rationals of reals) is probabilistically decidable in polynomial time. The result now from Theorem 3.2.

Corollary 4.1. There is a function $k(\epsilon)$ such that for every $0 < \epsilon \leq 1$ there is an algorithm of time complexity $O((mn)^{k(\epsilon)})$ which computes the rank of any m × n matrix with probability of error $<\epsilon$.

Remark. Clearly, the results above generalize to the class of matrices whose entries are represented by total $\{z \leftarrow 1, z \leftarrow x + y, z \leftarrow x - y, z \leftarrow x * y, z \leftarrow x/y\}$-programs. For discussions on the relationship between r.e.'s and straight-line programs see [2,6].

[5] Deciding if a program outputs 0 for all inputs.

References

[1] O. Ibarra and S. Moran, A generalized fast matrix decomposition algorithm and applications, University of Minnesota, Department of Computer Science, Tech. Rep. No. 80-20 (1980).

[2] O. Ibarra and S. Moran, Probabilistic algorithms for deciding equivalence of straight-line programs, to appear in J. ACM. (Available as University of Minnesota, Computer Science Department, Tech. Rep. 80-12, 1980.)

[3] D. Knuth, The Art of Computer Programming, Vol. 2; Seminumerical Algorithms (Addison-Wesley, Reading, MA, 1969).

[4] B. Noble, Applied Linear Algebra (Prentice Hall, Englewood Cliffs, NJ, 1969).

[5] V. Strassen, Gaussian elimination is not optimal, Numer. Math. 13 (1969) 354–356.

[6] L. Valiant, Completeness classes in algebra, Proc. 11th Annual ACM Symp. on Theory of Computing (1979) 249–261.

[7] Y. Yemini, On some randomly decidable geometrical problems, submitted for publication, 1979.

[8] Y. Yemini, On some theoretical aspects of position-location problems, Proc. 20th Annual IEEE Symp. on Foundations of Computer Science (1979) 1–8.