The Complexity of Verifying the Characteristic Polynomial and Testing Similarity

Thanh Minh Hoang   Thomas Thierauf

Abt. Theoretische Informatik
Universität Ulm
89069 Ulm, Germany

Abstract

We investigate the computational complexity of some important problems in linear algebra.

1. *The problem of verifying the characteristic polynomial of a matrix is known to be in the complexity class* $\mathbf{C_=L}$ *(*Exact Counting in Logspace*). We show that it is complete for* $\mathbf{C_=L}$ *under logspace many-one reductions.*

2. *The problem of deciding whether two matrices are similar is known to be in the complexity class* $\mathbf{AC}^0(\mathbf{C_=L})$. *We show that it is complete for this class under logspace many-one reductions. We also consider the problems of deciding equivalence and congruence of matrices.*

1 Introduction

Valiant [Val79b, Val79a] initiated the study of the computational complexity of counting problems. He introduced the counting class $\#\mathbf{P}$ that, intuitively, counts the number of solutions of $\mathbf{NP}$-problems. An example for a complete problem for this class is computing the permanent of a matrix.

Since counting is restricted to nonnegative integers, Fenner, Fortnow, and Kurtz [FFK94] extended $\#\mathbf{P}$ to the class $\mathbf{GapP}$, the closure of $\#\mathbf{P}$ under subtraction. It follows that computing the permanent of integer matrices is $\mathbf{GapP}$-complete.

In contrast, computing the determinant of a matrix is logspace many-one complete for $\mathbf{GapL}$ [Dam91, Tod91, Vin91, Val92], the class corresponding to $\mathbf{GapP}$ in the logspace setting. This huge difference in the complexity of the two problems [1] is somewhat surprising since the permanent and the determinant have almost the same cofactor expansion, the only difference comes with the sign.

$\mathbf{GapL}$ turns out to capture the complexity of many other natural problems: computing

- the powers of a matrix,
- iterated matrix multiplication,
- the inverse of a matrix,
- the characteristic polynomial of a matrix.

There are also graph theoretic problems related to counting the number s-t-paths in a graph.

Interesting decision problems can be derived from the above problems. For example, instead of computing the inverse of a matrix, it often suffices to decide whether the inverse *exists*. That is to decide whether the determinant is zero. More general, this motivates the complexity class $\mathbf{C_=L}$ where one has to *verify* the value of a $\mathbf{GapL}$ problem.

Problems that are hard for $\mathbf{GapL}$ usually result in verification problems that are hard for $\mathbf{C_=L}$. The determinant gives a nice example: checking singularity

[1] Note however that there is no proof yet that $\mathbf{GapL} \neq \mathbf{GapP}$.

is complete for $\mathbf{C_{=}L}$. Also, verifying the n-th power of a matrix is complete for $\mathbf{C_{=}L}$.

But there are exceptions! An example is to

- verify the inverse of a matrix:
  given matrices A and B,
  check whether $A^{-1} = B$.

This can be solved by computing the product AB. The product should be the identity matrix. Hence this can be solved in $\mathbf{NC}^1$, a subclass of $\mathbf{C_{=}L}$.

In contrast, if we have to

- verify *one entry* of the inverse:
  given matrix A, an integer a and indices i and j,
  decide whether $(A^{-1})_{i,j} = a$.

This is still complete for $\mathbf{C_{=}L}$. In other words, verifying *one* entry of the inverse is a harder problem than verifying *all* elements. In the latter, we put too much information in the input.

We consider the problem to

- verify the characteristic polynomial of a matrix:
  given a matrix A and the coefficients of a polynomial p,
  check whether $\chi_A = p$.

It follows from a theorem of Berkowitz [Ber84] that this is in $\mathbf{C_{=}L}$, and Santha and Tan [ST98] asked whether it is complete there.

Recall that the determinant is the constant term in the characteristic polynomial of a matrix and that verifying the determinant is complete for $\mathbf{C_{=}L}$. Now, with the different complexities of the above two inverse problems in mind, the question is: is it easier to verify *all* the coefficients of the characteristic polynomial than to verify just *one* of them? We show that this is *not* the case: verifying the characteristic polynomial is complete for $\mathbf{C_{=}L}$.

Furthermore, we consider

- the similarity problem:
  given matrices A and B,
  check whether they are similar, that is, whether there exists a nonsingular transformation matrix P such that $A = P^{-1}BP$.

Santha and Tan [ST98] showed that it is in $\mathbf{AC}^0(\mathbf{C_{=}L})$, the class of sets that are $\mathbf{AC}^0$-reducible to $\mathbf{C_{=}L}$. They ask whether it is complete in this class. Again, we give a positive answer to this question: the similarity problem is complete for $\mathbf{AC}^0(\mathbf{C_{=}L})$ under logspace many-one reductions.

We also consider two related relations on matrices, namely the equivalence relation and the congruence relation. We show that equivalence is complete for $\mathbf{AC}^0(\mathbf{C_{=}L})$ as well. For congruence, we can only show that it is hard for $\mathbf{AC}^0(\mathbf{C_{=}L})$.

The maybe most challenging open problem here is whether $\mathbf{C_{=}L}$ is closed under complement. Many related classes have this property:

- The most popular one is nondeterministic logspace, $\mathbf{NL}$, shown by Immerman [Imm88] and Szelepcsényi [Sze88].

- For symmetric logspace, $\mathbf{SL}$, this was shown by Nisan and Ta-Shma [NTS95].

Also, for probabilistic logspace, $\mathbf{PL}$, it is trivial. For unambiguous logspace, $\mathbf{UL}$, it is open as well. For the latter class, however, Reinhardt and Allender [RA97] showed that the *nonuniform* version of it, $\mathbf{UL}/$*poly*, is closed under complement. This gives rise to the conjecture that $\mathbf{UL}$ is closed under complement too.

One possible way of proving $\mathbf{C_{=}L}$ to be closed under complement is to reduce the singularity problem to the non-singularity problem. That is, given a matrix A, construct a matrix B (in logspace) such that A is singular if and only if B is nonsingular. It is well known that one does not need to consider an *arbitrary* matrix A: one can assume that A is an upper triangular matrix except for the entry in lower left corner. To prove our completeness result for verifying the characteristic polynomial, we manipulate such matrices. We think that it is quite interesting to see such transformations, because this can give some hints on how to come up with a reduction as above to solve the complementation problem for $\mathbf{C_{=}L}$. Therefore the methods we use are interesting in their own right. For more background and interesting results we recommend the paper of Allender, Beals, and Ogihara [ABO99].

2 Preliminaries

For a nondeterministic logspace bounded Turing machine M, we denote the number of accepting paths on input x by $acc_M(x)$, and by $rej_M(x)$ the number of rejecting paths. The difference of these two numbers is $gap_M(x) = acc_M(x) - rej_M(x)$.

The class of logspace computable sets is denoted by $\mathbf{L}$, the corresponding function class by $\mathbf{FL}$. The class of sets computable in nondeterministic logspace is denoted by $\mathbf{NL}$.

For the counting classes, we have $\#\mathbf{L}$, the class of functions $acc_M(x)$ for some nondeterministic logspace bounded Turing machine M, and $\mathbf{GapL}$ based analogously on functions gap_M. Based on counting, we consider the language class $\mathbf{C_=L}$: a set A is in $\mathbf{C_=L}$, if there exists a $f \in \mathbf{GapL}$ such that for all x:

$$x \in A \iff f(x) = 0.$$

For sets A and B, A is *(logspace) many-one reducible to* B, in symbols: $A \leq_m^L B$, if there is a function $f \in \mathbf{FL}$ such that for all x we have $x \in A \iff f(x) \in B$. We also consider other reducibility notions below. When we talk of reductions, we mean logspace many-one reductions.

We note that $\mathbf{C_=L}$ is closed under many-one reductions: $A \leq_m^L B$ and $B \in \mathbf{C_=L}$ then $A \in \mathbf{C_=L}$.

A set A is *(logspace many-one) hard* for a complexity class $\mathcal{C}$, if $L \leq_m^L A$ for every set $L \in \mathcal{C}$. If additionally A is in $\mathcal{C}$, we call A *(logspace many-one) complete* for $\mathcal{C}$.

A is $\mathbf{AC}^0$*-reducible to* B, if there is a logspace uniform circuit family of polynomial size and constant depth that computes A with unbounded fan-in and-, or-gates and oracle gates for B. In particular, we consider the class $\mathbf{AC}^0(\mathbf{C_=L})$ of sets that are $\mathbf{AC}^0$-reducible to a set in $\mathbf{C_=L}$.

Next we define the problems we are looking at. If nothing else is said, our domain for the algebraic problems are the integers. For $n \times n$ matrices over the intergers we assume that the matrix elements have a binary representation of at most n bits.

- POWER
  Input: a $n \times n$-matrix A and m, $(1 \leq m \leq n)$.
  Output: A^m, the m-th power of A.

- POWERELEMENT
  Input: a $n \times n$-matrix A and integers i, j, m with $(1 \leq i, j, m \leq n)$.
  Output: $(A^m)_{i,j}$, the (i, j)-th entry of A^m.

- DETERMINANT
  Input: a $n \times n$-matrix A.
  Output: $\det(A)$, the determinant of A.

- CHARPOLYNOMIAL
  Input: a $n \times n$-matrix A.
  Output: $(c_0, c_1, \ldots, c_{n-1})$, the coefficients of the characteristic polynomial $\chi_A(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_0$ of the matrix A.

These problems are all known to be in $\mathbf{GapL}$. For each of them, we define the *verification problem* as the graph of the corresponding $\mathbf{GapL}$-function. That is, for a function $f(x)$, we denote the graph of f as V-f (for *verify* f),

$$\text{V-}f \quad = \quad \{ (x, y) \mid f(x) = y \}.$$

This yields the verification problems

- V-POWER,
- V-POWERELEMENT,
- V-DETERMINANT, and
- V-CHARPOLYNOMIAL.

The first three problems are known to be complete for $\mathbf{C_=L}$. V-CHARPOLYNOMIAL is known to be in $\mathbf{C_=L}$. We show in Section 3 that it is complete for $\mathbf{C_=L}$ as well.

A special case of V-DETERMINANT is

- SINGULARITY
  Input: a $n \times n$-matrix A.
  Decide whether $\det(A) = 0$.

Also SINGULARITY is complete for $\mathbf{C_=L}$.

Related problems are computing the rank of a matrix, RANK, or deciding whether a system of linear equations is feasible, FSLE for short.

- FSLE
  Input: a matrix A and a vector b.
  Decide whether there is a rational vector x such that $Ax = b$.

FSLE is complete for $\mathbf{AC}^0(\mathbf{C_=L})$ [ABO99].

In Section 4 we consider three problems with complexity related to FSLE. These are some standard equivalence relations on matrices: equivalence, congruence, and similarity of matrices.

- EQUIVALENCE
  Input: two $n \times n$-matrices A and B.
  Decide whether A and B are equivalent. That is, whether there exist two nonsingular matrices P and Q such that $A = PBQ$.

Congruence is a special case of equivalence of two symmetric matrices where we have $Q = P^T$.

- CONGRUENCE
  Input: two symmetric $n \times n$-matrices A and B.
  Decide whether A and B are congruent. That is, whether there exists a nonsingular matrix P such that $A = P^T BP$.

Finally, the similarity problem is a special case of equivalence where we have $Q = P^{-1}$.

- SIMILARITY
  Input: two $n \times n$-matrices A and B.
  Decide whether A and B are similar.That is, whether there exists a nonsingular matrix P such that $A = P^{-1}BP$.

Santha and Tan [ST98] have shown that SIMILARITY in $\mathbf{AC}^0(\mathbf{C}_=\mathbf{L})$. We show in Section 4 that it is complete for $\mathbf{AC}^0(\mathbf{C}_=\mathbf{L})$ under logspace many-one reductions. More specifically, we reduce FSLE to SIMILARITY. This also holds for EQUIVALENCE. For CONGRUENCE, we can show that it is logspace many-one hard for $AC^0(\mathbf{C}_=\mathbf{L})$.

3 Verifying the Characteristic Polynomial

To show that V-CHARPOLYNOMIAL is complete for $\mathbf{C}_=\mathbf{L}$, we give a reduction from V-POWERELEMENT to V-CHARPOLYNOMIAL. This follows from the reduction POWERELEMENT $\leq_m^L$ DETERMINANT which goes back to Toda [Tod91] and Valiant [Val92]. The reduction presented here is taken from [ABO99].

Theorem 3.1 [Tod91, Val92]
POWERELEMENT $\leq_m^L$ DETERMINANT.

Proof. Let A be a $n \times n$ matrix and $1 \leq m \leq n$. We construct a matrix B such that $(A^m)_{1,n} = \det(B)$. That is, w.l.o.g. we fix $i = 1$ and $j = n$ in the definition of POWERELEMENT.

Interpret A as representing a directed bipartite graph on $2n$ nodes. That is, the nodes are arranged in two columns of n nodes each. In both columns, nodes are numbered from 1 to n. If entry $a_{k,l}$ of A is not zero, then there is an edge labeled $a_{k,l}$ from node k in the first column to node l in the second column. Now, take m copies of this graph, put them in a sequence and identify each second column of nodes with the first column of the next graph in the sequence. Call the resulting graph G'.

G' has $m + 1$ columns of nodes. The *weight* of a path in G' is the product of all labels on the edges of the path. The crucial observation now is that the entry at position $(1, n)$ in A^m is the sum of the weights of all paths in G' from node 1 in the first column to node n in the last column. Call these two nodes s and t, respectively.

As an intermediate result this provides a reduction from POWERELEMENT to the weighted path problem on graphs.

Graph G' is further modified: for each edge (k, l) with label $a_{k,l}$, introduce a new node u and replace the edge by two edges, (k, u) with label 1 and (u, l) with label $a_{k,l}$. Now all paths from s to t have *even* length, but still the same weight. Add an edge labeled 1 from t to s. Finally, add self loops labeled 1 to all nodes, except t. Call the resulting graph G.

Let B be the adjacency matrix of G. The determinant of B can be expressed as the sum over all weighted cycle covers of G. However, every cycle cover of G consists of a path from s to t, (due to the extra edge from t to s) and self loops for the remaining nodes. The single nontrivial cycle in each cover has odd length, and thus corresponds to an even permutation. Therefore, $\det(B)$ is precisely the sum over all weighted path from s to t in G'. We conclude that $\det(B) = (A^m)_{1,n}$ as desired. □

We want to use these techniques to show

$$\text{V-POWERELEMENT} \leq_m^L \text{V-CHARPOLYNOMIAL}.$$

The idea for this reduction is to construct a matrix, where the coefficients of the characteristic polynomial of the matrix can be expressed in terms of the value $(A^m)_{1,n}$. We show that the matrix $B - I_N$ has this property, where B is the matrix from the proof above and I_N is the N-dimensional identity matrix.

Let $C = B - I_N$. Matrix C is the adjacency matrix of a graph, call it H. We obtain H from graph G in the above proof as follows: the subtraction of I_N from B corresponds to taking away all the self loops in graph G and adding a self loop with weight -1 to the node t.

We consider the matrix C in more detail. Let the bipartite graph defined by matrix A have n nodes and e edges. Then graph H has $N = m(n+e)+n$ nodes and therefore C (and B) is a $N \times N$ matrix.

Except for the self loop at node t and the edge from t to s, graph H is acyclic. Thus we can put the nodes of H in such an order, that adjacency matrix C is upper triangular for the first $n-1$ rows with zeros along the main diagonal. The last row of C has a one on the first position, a minus one on the last position, and the rest is zero.

We also consider the upper triangle in C. Each column of graph G' was split in our construction into two columns and we got a new node on every edge.The first part we describe by the $n \times e$ matrix F:

$$F = \begin{pmatrix} 1\cdots 1 & 0\cdots 0 & \cdots & 0\cdots 0 \\ 0\cdots 0 & 1\cdots 1 & \cdots & 0\cdots 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0\cdots 0 & 0\cdots 0 & \cdots & 1\cdots 1 \end{pmatrix}$$

The number of ones in the k-th row of F is the number of edges leaving node k in the first column of G'.

From each of the newly introduced nodes there is one edge going out. Hence this second part we can describe by the $e \times n$-matrix S, which has precisely one non-zero entry in each row. The value of the non-zero entry is the weight of the corresponding edge in G'.

Now we can write C as follows:

$$C = \begin{pmatrix} & F & & & & \\ & & S & & & \\ & & & \ddots & & \\ & & & & F & \\ & & & & & S \\ L & & & & & R \end{pmatrix}$$

The empty places in C are all zero. L is the $n \times n$ matrix with a one at position $(n,1)$ and zero elsewhere, and R is the $n \times n$ matrix with -1 at position (n,n) and zero elsewhere.

Let the characteristic polynomial of C have the form:

$$\chi_C(x) = \det(xI_N - C) = x^N + \sum_{i=0}^{N-1} c_i x^i.$$

We give two ways how to compute the coefficients c_i in $\chi_C(x)$:

1. one way is to use elementary linear algebra and bring matrix C into triangular form. Then the characteristic polynomial is the product of the diagonal entries.

2. a short cut is provided by results in combinatorial matrix theory that generalize the argument given in the proof of Theorem 3.1 from the determinant, the constant coefficient of the characteristic polynomial, to all of its coefficients. (see [BR91, Zei85, MV97, MV99]).

We start by giving the combinatorial argument which is much shorter than the elementary argument.

The combinatorial way

From combinatorial matrix theory we know that the coefficient c_i in $\chi_C(x)$ equals the sum of the disjoint weighted cycles that cover $N-i$ nodes in H, with appropriate sign. In the graph H, all edges go from a layer to the next layer. The only exceptions are the edge (t,s) and the self loop (t,t). So any cycle must use precisely one of these, since the cycles should be disjoint. So the only disjoint cycles in H are of these two types:

- covering one node: the cycle (t,t) with weight -1. The sign is -1. So we have $c_{N-1} = 1$

- covering $2m+1$ nodes: each cycle that uses the edge (t,s) then traces out a path from s to t. The sum of all these paths is precisely $(A^m)_{1,n}$. The sign of these cycles is -1. Hence $c_{N-(2m+1)} = -(A^m)_{1,n}$.

All other coefficients must be zero. Therefore we have

$$(A^m)_{1,n} = a \\ \Longleftrightarrow \\ \chi_C(x) = x^N + x^{N-1} - ax^{N-(2m+1)}. \tag{1}$$

The algebraic way

We bring $(xI_N - C)$ into upper triangular form by doing row transformations.

For $x = 0$ it is easy to see that $\det(-C) = 0$. So let $x \neq 0$. We multiply the last row by x and add the first row to it. This yields a zero in the first position of the last row, but also some number of -1's to the right, coming from the first row of matrix F. We iterate the previous step: multiply the last row by x and add all the rows from it such that the x diagonal entry cancels the x entry in the last row. This in turn yields some non-zero entries further to the right in the last row coming from matrix S.

We continue doing this:

- if the first nonzero entry (from the left) in the last row is a integer constant, say α at position j, then multiply the last row by x and subtract α times the j-th row from the last row;
- if the first nonzero entry in the last row has the form αx, then we can directly subtract α times the j-th row.

Each iteration may put some more constants to the right of the current position into the last row, because of the matrices F and S. Since there are $2m$ of them in total, after $2m$ such iterations, all non-zero entries in the last row are in the last n positions.

This is the part of matrix R in the above description of C. The non-zero entries are all integers except for the last one: here we started with entry $x + 1$ in the beginning. We did $2m$ multiplications with x and added some rows from matrix S (the one just above R in C). Thus the entry has the form $(x + 1)x^{2m} + c$ for some constant c. To eliminate the constant entries in the last row, we multiply it one more time with x and subtract some of the last n rows to obtain zeros in the last row except for the last entry at position (N, N), which now has the form $((x + 1)x^{2m} + c)x$. Let $D(x)$ be the resulting upper triangular matrix.

The determinant of a triangular matrix is the product of the diagonal elements. Hence

$$\det(D(x)) = x^{N-1}\left((x+1)x^{2m} + c\right)x.$$

Note however that this is not the same as $\chi_C(x)$: the latter we changed with each multiplication of the last row by x, and we did this $2m + 1$ times. Therefore we get

$$\begin{aligned} \chi_C(x) &= \frac{\det(D(x))}{x^{2m+1}} \\ &= x^{N-(2m+1)}\left((x+1)x^{2m} + c\right). \end{aligned} \tag{2}$$

Note that $\chi_C(0) = 0$, so that this covers the case $x = 0$ as well.

The problem that remains in order to determine $\chi_C(x)$ is the value of the constant c. Note that c may depend on each of the above transformation steps. From equation (2) we get for $x = -1$

$$\chi_C(-1) = (-1)^{N-(2m+1)}c. \tag{3}$$

On the other hand, we can determine $\chi_C(-1)$ directly as

$$\begin{aligned} \chi_C(-1) &= \det((-1)I_N - C) \\ &= \det(-I_N - (B - I_N)) \\ &= \det(-B) \\ &= (-1)^N \det(B). \end{aligned} \tag{4}$$

From the equations (3) and (4) it follows that c must be $-\det(B)$. Hence,

$$(A^m)_{1,n} = a \\ \Longleftrightarrow \\ \det(B) = a \\ \Longleftrightarrow \\ \chi_C(x) = x^N + x^{N-1} - ax^{N-(2m+1)}.$$

In summary, both methods yield explicitly the coefficients of $\chi_C(x)$. Therefore we have the desired reduction from v-PowerElement to v-CharPolynomial. We conclude:

Theorem 3.2
v-CharPolynomial *is complete for* $\mathbf{C_=L}$.

The reductions shown in this section are actually not just logspace many-one reductions, but much stronger *logspace projections*. That is, the output is a projection of input values and, in addition, some constant values (we only needed -1, 0, and 1 as additional constants) that can be computed in logspace (actually in $\mathbf{TC}^0$).

4 Testing Similarity, Equivalence and Congruence

Recall that two matrices A and B are similar, if there exists a nonsingular matrix P such that $A = P^{-1}BP$. Santha and Tan [ST98] showed that there is a $\mathbf{AC}^0$-reduction from SIMILARITY to V-RANK. From this it follows that SIMILARITY is in $\mathbf{AC}^0(\mathbf{C_=L})$

We show on the other hand that the problem *feasible system of linear equations*, FSLE, can be reduced to SIMILARITY. Since FSLE complete for $\mathbf{AC}^0(\mathbf{C_=L})$ [ABO99], this follows for SIMILARITY as well.

Theorem 4.1
SIMILARITY *is complete for* $\mathbf{AC}^0(\mathbf{C_=L})$.

Proof. Let A be a $n \times n$ matrix and $b = (b_1, \ldots, b_n)$ an n vector. We will construct two matrices C and D, such that the system $Ax = b$ has a solution iff C and D are similar.

Define

$$C = \left(\begin{array}{c|c} & b_1 \\ A & \vdots \\ & b_n \\ \hline 0 \cdots 0 & 0 \end{array} \right), \quad D = \left(\begin{array}{c|c} & 0 \\ A & \vdots \\ & 0 \\ \hline 0 \cdots 0 & 0 \end{array} \right)$$

Suppose first that the system $Ax = b$ has *no* solution. Then rank(C) = rank$(D) + 1$, and therefore C and D cannot be similar, because similar matrices must have the same rank. (Note that the transformation matrix in the definition of SIMILARITY is nonsingular.)

If, on the other hand, $Ax = b$ has a solution, say x_0, then we can define a transformation matrix P as follows:

$$P = \left(\begin{array}{c|c} I_n & x_0 \\ & \\ \hline 0 \cdots 0 & -1 \end{array} \right).$$

Now it is easy to check that P has full rank and that $CP = PD$. Therefore C and D are similar. □

Next we investigate the equivalence and congruence problems. Recall that two matrices A and B of the same order n are equivalent, if there exist nonsingular matrices P and Q, such that $A = PBQ$. It is well known that this holds iff A and B have the same rank. Allender, Beals, and Ogihara [ABO99] have shown that the rank of a matrix can be computed in $\mathbf{AC}^0(\mathbf{C_=L})$. Therefore EQUIVALENCE is in $\mathbf{AC}^0(\mathbf{C_=L})$. They also used the fact that a linear system $Ax = b$ has a solution iff $rank(A) = rank(A|b)$. From this we obtain that EQUIVALENCE is complete for $\mathbf{AC}^0(\mathbf{C_=L})$.

Fact 4.2
EQUIVALENCE *is complete for* $\mathbf{AC}^0(\mathbf{C_=L})$.

Proof. Let A be a $n \times n$ matrix and $b = (b_1, \ldots, b_n)$ a vector. Let C and D be the matrices defined in the proof of Theorem 4.1. Then the system $Ax = b$ has a solution iff $rank(A) = rank(A|b)$ iff rank(C) = rank(D) iff C and D are equivalent. □

Recall that two symmetric matrices A and B (over the reals) of the same order n are congruent, if there exists a nonsingular matrix P, such that $A = P^T BP$. It is known that A and B are congruent iff they have the same rank and signature. (The signature of a symmetric matrix is the number of positive eigenvalues minus the number of negative eigenvalues of the matrix.)

We don't have an upper bound on the complexity of CONGRUENCE. In particular, the complexity of computing the signature is an open problem. As a lower bound, we can show that it is hard for $\mathbf{AC}^0(\mathbf{C_=L})$.

Lemma 4.3
CONGRUENCE *is hard for* $\mathbf{AC}^0(\mathbf{C_=L})$.

Proof. We reduce EQUIVALENCE to CONGRUENCE. Let A and B be two $n \times n$ matrices. We will construct two matrices C and D, such that A and B are equivalent iff C and D are congruent.

We define $C = (A^T A)^2$ and $D = (B^T B)^2$. Note that C and D are symmetric and we have

$$rank(A) = rank(A^T A) = rank((A^T A)^2).$$

Therefore A and C have the same rank, and the same holds for B and D. Moreover, the eigenvalues of C

and D are all nonnegative. Therefore, their rank equals their signature. We conclude that A and B have the same rank iff C and D have the same rank *and* signature. This proves the claim. $\Box$

Note that the reductions in Theorem 4.1 and Fact 4.2 are not just logspace reductions but logspace uniform projections. The reduction in Lemma 4.3 can be computed in $\mathbf{TC}^0$.

Open Problems

Two (real) symmetric matrices are congruent iff they have the same rank and signature. To decide, whether two matrices have the same rank, is in $\mathbf{AC}^0(\mathbf{C}_=\mathbf{L})$. The complexity of the analog problem for the signature is open. So we don't know the complexity of the problem CONGRUENCE. We expect that this problem complete for $\mathbf{AC}^0(\mathbf{C}_=\mathbf{L})$ too. A related problem is to *verify* the signature of a symmetric matrix. Can this be done in $\mathbf{C}_=\mathbf{L} \wedge \mathbf{coC}_=\mathbf{L}$, the class of sets that can be written as the intersection of a set in $\mathbf{C}_=\mathbf{L}$ and a set in $\mathbf{coC}_=\mathbf{L}$?

A problem related to V-CHARPOLYNOMIAL is to decide whether a polynomial is the minimal polynomial of a given matrix A. We don't know the complexity of this problem.

A problem related to SIMILARITY is to decide whether a given matrix A is diagonalizable. That is, whether it is similar to diagonal matrix. We don't know the complexity of this problem.

The more important question is whether $\mathbf{C}_=\mathbf{L}$ is closed under complement. An affirmative answer would solve most of the above questions because then all the complexity classes considered here coincide.

Acknowldegments

We are greatful to Eric Allender, V. Arvind, and Manindra Agrawal for very helpful discussions. Also, the remarks we got from the referees were very detailed and useful. In particular, we have to thank the referee who pointed us to the much shorter combinatorial proof of Theorem 3.2.

References

[ABO99] E. Allender, R. Beals, and M. Ogihara. The complexity of matrix rank and feasible systems of linear equations. volume 8, pages 99 –126, 1999.

[Ber84] S. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18:147–150, 1984.

[BR91] R. Brualdi and H. Ryser. *Combinatorial Matrix Theory*, volume 39 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1991.

[Dam91] C. Damm. DET = $L^{(\#L)}$. Technical Report Informatik-Preprint 8, Fachbereich Informatik der Humboldt-Universität zu Berlin, 1991.

[FFK94] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48:116–148, 1994.

[Imm88] N. Immerman. Nondeterministic space is closed under complement. *SIAM Journal on Computing*, 17:935–938, 1988.

[MV97] M. Mahajan and V Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago Journal of Theoretical Computer Science*, 1997(5), 1997.

[MV99] M. Mahajan and V Vinay. Determinant: Old algorithms, new insights. *SIAM Journal on Discrete Mathematics*, 12(4):474–490, 1999.

[NTS95] N. Nisan and A. Ta-Shma. Symmetric logspace is closed under complement. *Chicago Journal of Theoretical Computer Science*, 1995(Article 1), 1995.

[RA97] K. Reinhardt and E. Allender. Making nondeterminism unambigous. In *38th Symposium on Foundation of Computer Science*, pages 244–253. IEEE Computer Society Press, 1997.

[ST98] M. Santha and S. Tan. Verifying the determinant in parallel. *Computational Complexity*, 7:128–151, 1998.

[Sze88] R. Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26(3):279–284, 1988.

[Tod91] S. Toda. Counting problems computationally equivalent to the determinant. Technical Report CSIM 91-07, Dept. of Computer Science and Information Mathematics, University of Electro-Communications, Chofu-shi, Tokyo 182, Japan, 1991.

[Val79a] L. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.

[Val79b] L. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8:410–421, 1979.

[Val92] L. Valiant. Why is boolean complexity theory difficult. In M.S. Paterson, editor, *Boolean Function Complexity*, London Mathematical Society Lecture Notes Series 169. Cambridge University Press, 1992.

[Vin91] V Vinay. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *6th IEEE Conference on Structure in Complexity Theory*, pages 270–284, 1991.

[Zei85] D. Zeilberger. A combinatorial approach to matrix algebra. *Discrete Mathematics*, 56:61–72, 1985.