Counting Problems Computationally Equivalent to Computing the Determinant

Seinosuke Toda

Technical Report CSIM 91-07

May 1991

Dept. Computer Science and Information Mathematics

University of Electro-Communications

Chofu-shi, Tokyo 182, Japan

(mailing address)

Seinosuke Toda

Department of Computer Science and Information Mathematics

University of Electro-Communications

1–5–1 Chofugaoka, Chofu-shi, Tokyo 182, Japan

Email: toda@cso.cs.uec.ac.jp   Phone: 0424(83)2161 (Ex. 4112)

(title) Counting Problems Computationally Equivalent to

Computing the Determinant

(author) Seinosuke Toda

(affiliation)

Department of Computer Science and Information Mathematics

University of Electro-Communications

Chofu-shi, Tokyo 182, Japan

(abbreviated title) Counting Problems and the Determinant

(mailing address)

Seinosuke Toda

Department of Computer Science and Information Mathematics

University of Electro-Communications

1–5–1 Chofugaoka, Chofu-shi, Tokyo 182, Japan

Email: toda@cso.cs.uec.ac.jp Phone: 0424(83)2161 (Ex. 4112)

Abstract The main purpose of this paper is to exhibit non-algebraic problems that are computationally equivalent to computing the integer determinant. For this purpose, some graph-theoretic counting problems are shown to be equivalent to the integer determinant problem under suitable reducibilities. Those are the problems of counting the number of all paths between two nodes of a given acyclic digraph, the number of all smallest length paths between two nodes of a given undirected graph, the number of rooted spanning trees of a given digraph, and the number of Eulerian paths in a given digraph. It is also observed that the integer determinant problem (and some well-known linear algebraic problems) remains its essential complexity even if we require all entries of a given matrix to be either zero or one.

1 Introduction

It is well understood that the computational complexity of computing the determinant captures the complexity of most elementary computations in linear algebra, such as matrix inversion, the coefficients of the characteristic polynomial, the power series of a given matrix, and iterated matrix multiplication. This property of the determinant tells us a significance of investigating the complexity of computing the determinant. Until Cook [Coo85] initiated such an investigation, most researchers have given their attention to developing efficient algorithms for computing the determinant and ones for computing other algebraic problems while the later algorithms include several reductions of those problems to the determinant. Cook defined the class DET of functions that are NC^1-reducible to the integer determinant and exhibited some linear algebraic problems that are complete for the class.

Nonetheless, as far as the author knows, all the problems known to be computationally equivalent to the integer determinant are algebraic. The purpose of the present paper is to exhibit some non-algebraic problems that are computationally equivalent to the integer determinant. We expect that such non-algebraic problems shall give us a variety of complexity theoretic insights on the nature of the determinant.

Our main observation is that some graph-theoretic counting problems are as computationally hard as the integer determinant. Some relationships are known so far between such counting problems and the integer determinant. For instance, the number of rooted spanning trees of a digraph is given by the determinant of an appropriate integer matrix constructed from the digraph [Tut48], and similarly, the number of Eulerian cycles in a given Eulerian digraph is given by the determinant of an integer matrix constructed from the digraph [BE51] (see [HP73] also). In fact, we will show that the converse relationships also hold. More precisely speaking, we show that the integer determinant is interreducible to the problems of counting the followings: the number of paths between two nodes of a given acyclic digraph, the number of shortest paths between two nodes of an undirected graph or a digraph where by 'a shortest path' we mean a path of the smallest length between the two nodes, the number of rooted spanning trees of a given digraph, and the number of Eulerian cycles of a given Eulerian digraph. A complexity class related to these problems has been investigated by Àlvarez and Jenner [AJ90]. They defined the class #L of functions that give the number of accepting computations of logspace bounded nondeterministic Turing machines in which all computations are of length at most a polynomial in the length of inputs, and showed a complete function for the class under logspace many-one reducibility. Our first counting problem above is one of the most elementary complete function for the class, and in fact, we can observe that the problem straightforwardly represents the computations in the class. Thus we see that #L is computationally equivalent to DET under NC^1-reducibility and hence that all complete functions for #L is also complete for DET. Our results may also be contrasted to the results by Valiant [Val79b]. A similarity and a difference between the determinant and the permanent are translated into those between his #P-complete functions and the problems above. To state the contrasts on the above four problems, all the problems of counting the number of all paths between two nodes of a digraph *not necessarily acyclic*, the number of all *simple* paths *not necessarily shortest* between two nodes of an undirected graph, the number of all rooted trees *not necessarily spanning* of a digraph, and the number of all *Hamiltonian* paths in a digraph are #P-complete.

As another interesting observation, we also show that the integer determinant does not loose its essential complexity even if we require all entries of a given matrix to be either zero or one. This will be obtained in the intermediate process of showing our main results. By this result, we also see that some other algebraic problems mentioned above have the same property.

2 Counting Problems on Graphs

In order to show our results, we make some technical assumptions on graphs and integers. All nodes in a graph with n nodes are indexed by numbers between 1 through n, and hence each node will often be called by its index. We suppose that a digraph (with n nodes) is given by its adjacency matrix, an $n \times n$ (0,1)-matrix whose (i, j)th entry is one iff there is an edge from the ith node to the jth node. The undirected case is similar. All integers are supposed to be represented in 2's-complement form.

Our results are partitioned into three groups depending on the reducibilities that we use. The first group includes some problems of which the integer determinant problem is a p-projection in the sense of Skyum and Valiant [SV85]. The second group includes three counting problems to which the integer determinant problem is equivalent with respect to the polynomial-size and constant-depth truth-table reducibility in the sense of Chandra, Stockmeyer and Vishkin [CSV84]. The third one includes a counting problem to which the integer determinant problem is equivalent with respect to P-uniform NC^1-reducibility.

Theorem 2.1 The following problems are p-equivalent to each other.

INTDET

Given: an $n \times n$ matrix of n-bit integer entries.

Compute: its determinant.

Remark. The problem remains p-equivalent even if all entries are either 0 or 1. We refer to the restricted version as ZERO-ONE DET.

MATPOW$_{\{-1,0,1\}}$

Given: an $n \times n$ integer matrix A of entries either -1, 0, or 1.

Compute: the $(1, n)$th entry of $\sum_{i=1}^{n} A^i$.

Remark. We denote by MATPOW [Coo85] the same problem where all entries of the matrix are allowed to be arbitrary integers (in 2's-complement form of length n).

NthPOWER$_{\{-1,0,1\}}$

Given: an $n \times n$ integer matrix of entries either -1, 0, or 1.

Compute: the $(1, n)$th entry of the nth power of the matrix.

DIRECTED PATH DIFFERENCE

Given: two monotone digraphs G and H with n nodes, where a digraph is called *monotone* if it contains no edge (i, j) such that $i \geq j$ (more intuitively, G is monotone if it has an obvious topological sort).

Compute: $\#\text{PATH}_G(1, n) - \#\text{PATH}_H(1, n)$, where $\#\text{PATH}_G(i, j)$ denotes the number of paths in G from the ith node to the jth node.

Proof. We perform the following reductions among the problems:

$$\text{INTDET} \le_{\text{proj}} \text{MATPOW}_{\{-1,0,1\}} \le_{\text{proj}} \text{NthPOWER}_{\{-1,0,1\}}$$
$$\le_{\text{proj}} \text{DIRECTED PATH DIFFERENCE} \le_{\text{proj}} \text{ZERO-ONE DET},$$

where by $f \le_{\text{proj}} g$ we mean that f is a p-projection of g. Though we will not give the formal description of each reduction, such a formal description can be easily obtained from our construction below.

INTDET $\le_{\text{proj}}$ MATPOW$_{\{-1,0,1\}}$.

As mentioned in [Coo85], we can perform a reduction of INTDET to MATPOW by converting the algorithm developed by Berkowitz [Ber84] for computing the determinant. Though an immediate translation of his algorithm to a reduction of INTDET to MATPOW needs to use matrix powering several times, it is not hard to modify the reduction into one projecting INTDET to MATPOW. The detail is left to the reader. Thus, in this proof, we show that MATPOW is a p-projection of MATPOW$_{\{-1,0,1\}}$.

Given an $n \times n$ matrix A of n-bit integer entries, we first construct a monotone digraph with $n(n+1)$ nodes and with edges weighted by integers such that the $(1,n)$th entry of $\sum_{i=1}^{n} A^i$ is given by the sum of all products of weights on paths from a node s to a node

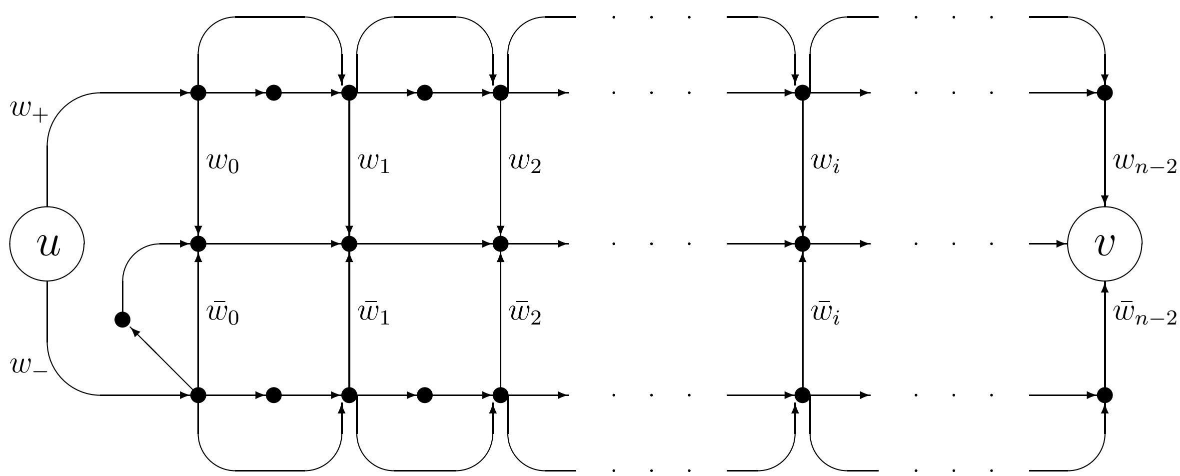

Figure 2.1. The subgraph replacing an edge with weight $w = w_{n-1}w_{n-2}\cdots w_0$ (in 2's complement form), where $w_+=1$ if $w_{n-1}=0$ and $w_+=0$ otherwise, and $w_- = -1$ if $w_{n-1}=1$ and $w_-=0$ otherwise. Note that $w = \sum_{\text{p}}\{$product of all weights on p $\mid$ p is a path from u to v $\}$.

t of the digraph. The construction is easily done according to the inductive definition of matrix powering, and so is left to the reader. We next construct a monotone digraph H from G such that all edges in H are weighted with either -1, 0, or 1 and such that for two specified nodes s_H and t_H of H, the sum of all products of weights on paths in H from s_H to t_H is equal to the sum of all products of weights on paths in G from s to t. An idea for constructing such a digraph is to replace each edge of G by a monotone digraph with two specified nodes such that the number of paths in the digraph between the two nodes equals the absolute value of the weight of the G's edge; for the precise construction, we may attach to the digraph an edge realizing the sign of the weight concerned. We describe such a digraph in Figure 2.1. Let G_e denote a digraph constructed for an edge $e = (u, v)$ of G as in Figure 2.1. Then we easily see that the weight of e is given by the sum of all products of weights on paths in G_e from the node u to the node v. Replacing the edge e by G_e (i.e., deleting the edge from G and instead adding G_e to G) we obtain a desired digraph H. After this construction, we may define a square matrix M whose (i, j)th entry includes the weight of an edge connecting from the ith node to the jth node of H. Then we complete the reduction.

$\mathrm{MATPOW}_{\{-1,0,1\}} \leq_{\mathrm{proj}} \mathrm{NthPOWER}_{\{-1,0,1\}}$.

This is easy and hence is left to the reader.

$\mathrm{NthPOWER}_{\{-1,0,1\}} \leq_{\mathrm{proj}}$ DIRECTED PATH DIFFERENCE.

Let A be an $n \times n$ matrix whose entries each is either -1, 0, or 1. Then we will construct a monotone digraph G such that for three specified nodes s, t_+, and t_-, the $(1, n)$th entry of $\sum_{i=1}^{n} A^i$ is given by $\#\mathrm{PATH}_G(s, t_+) - \#\mathrm{PATH}_G(s, t_-)$. Such a digraph is constructed as follows.

(1) Define G_0 as a digraph of the node set $\{\ [i, l] \mid 1 \leq i \leq n$ and $0 \leq l \leq n\ \}$ such that for all $1 \leq i, j \leq n$ and $0 \leq l < n$, G_0 contains an edge $([i, l], [j, l + 1])$ iff the (i, j)th entry of A is 1.

(2) Make two copies G_+ and G_- of G_0.

(3) For all $0 \leq l < n$ and all $1 \leq i, j \leq n$ such that the (i, j)th entry of A is -1, draw an edge from the node $[i, l]$ of G_+ (resp., G_-) to the node $[j, l + 1]$ of G_- (resp., G_+). G

is the resulting digraph where s is the node $[1,0]$ of G_+, and t_+ (resp., t_-) is the node $[n,n]$ of G_+ (resp., G_-).

Obviously, G is monotone, and we easily see that it satisfies the condition mentioned above.

DIRECTED PATH DIFFERENCE $\leq_{\text{proj}}$ ZERO-ONE DET.

We here use a modification of Valiant's idea in [Val79a]. Let G_1 and G_2 be two monotone digraphs with n nodes. We first construct a digraph H from G_1 and G_2 as follows:

(1) for all edges $e = (u, v)$ in G_1 and in G_2, replace e by two consecutive edges (u, v_e) and (v_e, v) where v_e is a new node,

(2) add to G_1 a new node t and an edge from the nth node of G_1 to t,

(3) connect G_1 and G_2 by introducing a new node s, edges from s to the first nodes of G_1 and G_2, an edge from t to s, and an edge from the nth node of G_2 to s, and finally

(4) attach self-loops to all nodes except s. H is the resulting digraph.

Let m denote the number of nodes in H and let us define an $m \times m$ matrix M as the adjacency matrix of H. It is well known that each cycle cover of H determines exactly one permutation of m elements whose corresponding additive term in $\det(M)$ is not zero, and vice versa. Note that every cycle cover of H consists of one large cycle containing s and some self-loops. This means that for all permutations of m elements, if its corresponding additive term in $\det(M)$ is not zero, then the permutation is a cyclic one. We now partition such all cyclic permutations into ones of an odd length and ones of an even length. The partition also divides all cycle covers of H into two corresponding groups. Then, from the construction of H, we see a one-to-one correspondence between all cycle covers in the first group and all paths in G_1 from the first node to the nth one, and similarly see a one-to-one correspondence between all cycle covers in the second group and all paths in G_2 from the first node to the nth one. Hence, by the definition of the determinant, we have that $\det(M)$ is given by the difference between the number of all permutations in the first group and the number of all permutations in the second group, and it is equal to $\#\text{PATH}_{G_1}(1, n) - \#\text{PATH}_{G_2}(1, n)$. ∎

In the following, we use the polynomial-size constant-depth truth-table reducibility ($\leq_{\text{cd-tt}}$-reducibility for short) due to Chandra, Stockmeyer and Vishkin [CSV84], and by $f \leq_{\text{cd-tt}} g$ we mean that f is $\leq_{\text{cd-tt}}$-reducible to g.

Theorem 2.2 INTDET is $\leq_{\text{cd-tt}}$-reducible to each of the following problems, and vice versa.

DIRECTED PATH

Given: a monotone digraph G with n nodes.

Compute: the number of paths in G from the first node to the nth node.

SHORTEST PATH

Given: an undirected graph G with n nodes.

Compute: the number of shortest paths in G between the first node and the nth node, where we define the *length* of a path as the number of edges on the path.

Remark. The problem remains $\leq_{\text{cd-tt}}$-equivalent when the graph is directed.

ROOTED SPANNING TREE

Given: a digraph G.

Compute: the number of rooted spanning trees of G.

Proof. We perform the following reductions:

$$\text{DIRECTED PATH} \leq_{\text{proj}} \text{SHORTEST PATH} \leq_{\text{cd-tt}} \text{DIRECTED PATH}$$
$$\leq_{\text{cd-tt}} \text{ROOTED SPANNING TREE} \leq_{\text{proj}} \text{MATPOW}.$$

DIRECTED PATH $\leq_{\text{proj}}$ SHORTEST PATH.

Our task in this reduction is to construct from a given monotone digraph G with n nodes an undirected graph H with two specified nodes s and t such that all simple paths in H between s and t are of a same length and the number of those simple paths is equal to the number of paths in G from the first node to the nth node. Such a digraph H can be defined as follows:

(1) for each $0 \leq l \leq n-1$ and each $1 \leq i \leq n$, H contains a node of the form $[i, l]$,

(2) for each $0 \le l < n-1$ and each $1 \le i, j \le n$, H contains an edge $\{[i,l],[j,l+1]\}$ iff G contains the edge (i,j), and H contains an edge $\{[n,l],[n,l+1]\}$.

When we define $s = [1,0]$ and $t = [n, n-1]$, we easily see that H satisfies the condition mentioned above. This completes the resuction, since all simple paths in H between s and t are all of the shortest paths between the two nodes. A p-projection of DIRECTED PATH to the directed case of SHORTEST PATH is easily obtained from an obvious p-projection of the undirected case to the directed case in which every undirected edge is replaced by a pair of bidirectional edges.

SHORTEST PATH $\le_{\text{cd-tt}}$ DIRECTED PATH.

We below perform a reduction of the directed case of SHORTEST PATH to DIRECTED PATH. Given a digraph G with n nodes, we first construct a digraph G_1 from the digraph such that it is monotone and for all $1 \le i < n$, the number of paths of length i in G_1 from the first node to the last node is equal to that of G. Such a construction is very similar to one in the last reduction and hence is omitted. Then we may replace each edge of G_1 by a monotone digraph with two specified nodes such that it has 2^m paths from one node to the other, where 2^m denotes a strict upper bound on the number of possible paths in G_1. Denoting the resulting monotone digraph by H and corresponding the first and the last nodes of G_1 to those nodes of H, we see that the number of paths in H from the first to the last nodes is given in the form $\sum_{i=1}^{n-1} p_i \cdot 2^{i \cdot m}$, where p_i is the number of paths of length i in G from the first node to the last one. From this value, we can easily obtain the number of shortest paths in G from the first node to the last one. It is easy to see that all of the constructions above can be done in polynomial-size and constant-depth.

ROOTED SPANNING TREE $\le_{\text{cd-tt}}$ MATPOW.

In this and the next reductions, we use the following fact shown by Tutte [Tut48] (see [HP73] also).

Fact 1 [Tut48]. Let H be a digraph with n nodes whose edges are weighted by positive integers, and let $a_{i,j}$ denote the weight of the edge from the ith node to the jth node. We define Δ_H by

$$\Delta_H = \begin{bmatrix} \sum_{i\neq 1} a_{i,1} & -a_{1,2} & \cdots & -a_{1,n} \\ -a_{2,1} & \sum_{i\neq 2} a_{i,1} & \cdots & -a_{2,n} \\ -a_{3,1} & -a_{3,2} & \cdots & -a_{3,n} \\ \vdots & \vdots & & \vdots \\ -a_{n,1} & -a_{n,2} & \cdots & \sum_{i\neq n} a_{i,1} \end{bmatrix},$$

and denote by $\Delta_H(i|j)$ a submatrix of Δ_H obtained by deleting the ith row and the jth column. Then, for all $1 \leq i \leq n$, $\det(\Delta_H(i|i))$ is the sum of all products of weights on spanning trees of H rooted at the ith node. **(End-of-Fact 1)**

By an immediate use of the fact, we can reduce ROOTED SPANNING TREE to INT-DET via a suitable reducibility such as NC^1-reducibility, but it cannot be even a $\leq_{\text{cd-tt}}$-reduction since taking each summation in the above matrix involves to take the majority of n Boolean values that cannot be computed in constant-depth and polynomial-size [FSS81]. Thus we need an additional construction in order to establish the $\leq_{\text{proj}}$-reducibility. Given a digraph H as above, we can construct a monotone digraph G, based on Berkowitz's algorithm [Ber84], such that all edges of G are weighted by either entries of Δ_H or -1 and $\det(\Delta_H(1|1))$ is given by the sum of all products of weights on paths in G from the first node to the last node. Moreover, for each entry of Δ_H, we can construct a monotone digraph with weighted edges, as in Figure 2.1, such that for two specified nodes, the sum of all products of weights on paths from one node to the other is equal to the entry. By replacing each edge of G with an entry of Δ_H as its weight by a monotone digraph constructed for the entry as above, we have that the sum of all products of weights on paths in the resulting digraph from the first to the last nodes is equal to $\det(\Delta_H(1|1))$. We can construct a similar digraph for each $\Delta_H(i|i)$. From these digraphs, we can construct an $m \times m$ integer matrix A (for some $m \leq 1$) such that the $(1, m)$th entry of $\sum_{i=1}^{m} A^i$ is equal to the number of all rooted spanning trees of H. All of the above constructions can be realized as a p-projection. We leave the detail to the interested reader.

DIRECTED PATH $\leq_{\text{cd-tt}}$ ROOTED SPANNING TREE.

We use the notations in the last reduction. Let G be a monotone digraph with n nodes given as an instance to DIRECTED PATH. For simplicity, we assume, without loss of generality, that all nodes of G are of indegree at most two. (If a node of G is of indegree more than two, then we may replace all edges coming into the node by a suitable binary tree whose edges are oriented toward the node. Obviously, such a replacement remains the number of paths between any pairs of the original nodes unchanged.) Before stating our construction, we give its outline. Our construction of a desired digraph consists of three stages. In the first stage, we will construct from G a digraph G_1 with edges weighted by either -1 or 2 such that for some positive integers m_1 and m_2 that will be predetermined,

$$\det(M) = 2^{m_1} - 2^{m_2} \cdot \#\mathrm{PATH}_G(1, n),$$

where M denotes the square matrix whose (i, j)th entry contains the weight of an edge from the ith node to the jth node of G_1. In the second stage, We will construct a digraph G_2 from G_1 such that its edges are weighted by positive integers and $\det(M) = \det(\Delta_{G_2}(1|1))$. In the final stage, we will construct a digraph H with no edge weight from G_2 such that the number of all rooted spanning trees of H is equal to the determinant above. These constructions accomplish a required reduction of DIRECTED PATH to ROOTED SPANNING TREE. (Note that G_1 and G_2 are intermediate and technical conventions for proving the correctness of our construction. Thus we do not have to include those constructions in a concise realization of the reduction.)

Now we give the definition of G_1. To define it, we first define monotone digraphs $G_{1,i}$, for $1 \le i \le n-1$, as follows.

(1) $G_{1,i}$ contains nodes of the form $[i; k, l]$ for $1 \le k < n$ and $0 \le l \le i$, and nodes of the form $[i; n, l]$ for $i \le l \le n-1$.

(2) For all $1 \le k_1, k_2 \le n$ and $1 \le l \le i$, it contains an edge $([i; k_1, l], [i; k_2, l+1])$ iff G contains an edge from the k_1th node to the k_2th node, and for all $i \le l < n-1$, it contains an edge $([i; n, l], [i, n, l+1])$.

Then G_1 is constructed from these graphs as follows.

(3) Add a new node s_1 and edges from s_1 to the nodes $[i; 1, 0]$.

(4) Add a new node t_1 and connect all nodes of the form $[i; n, n-1]$ with t_1 by a binary tree satisfying the following conditions:

(a) all leaves are the nodes $[i; n, n-1]$ $(1 \leq i \leq n-1)$,

(b) all edges are oriented toward t_1, and

(c) all paths from the leaves to t_1 are of a same length.

(5) Add a new node s_0, an edge from s_0 to s_1, and an edge from t_1 to s_0, and finally, attach a self-loop to each node.

(6) To end the construction of G_1, we assign the weight two to all self-loops and assign -1 to all other edges.

For the graphs $G_{1,i}$ and G_1, we easily see the following facts.

Fact 2.

(1) For each $1 \leq i \leq n-1$, the number of paths on $G_{1,i}$ from $[i; 1, 0]$ to $[i; n, n-1]$ is equal to the number of paths of length i in G from the first node to the nth node, and hence, we see:

(2) the total number of paths in G from the first node to the nth node is given by the number of cycles in G_1 containing s_0 except its self-loop.

(3) Such all cycles are of a same length, which we below refer to as h. **(End-of-Fact 2)**

Let m denote the number of nodes in G_1 and let M denote an $m \times m$ matrix whose (i, j)th entry contains the weight of the edge of G_1 from the ith node to the jth node. By the same argument as in the reduction of DIRECTED PATH DIFFERENCE to ZERO-ONE DET and by the above facts, we have:

$$\begin{aligned}\det(M) &= 2^m + 2^{m-h} \cdot (-1)^h \cdot \text{parity}(h) \cdot \#\text{PATH}_G(1, n) \\ &= 2^m - 2^{m-h} \cdot \#\text{PATH}_G(1, n),\end{aligned}$$

where parity(h) is $+1$ if h is odd and is -1 otherwise.

We next construct G_2 from G_1. To do that, we first delete all self-loops from G_1 and associate weight one with all remaining edges. Then, we add a new node r to the resulting digraph, draw edges with weight one from r to all nodes of indegree one, and draw edges with weight two from r to all nodes of indegree zero. Note that all nodes in G_1 are of indegree at most two and hence, for every node in G_2 except r, the sum of all weights of edges coming into the node equals two. Then we can describe the matrix Δ_{G_2} in the following form:

$$\Delta_{G_1} = \begin{bmatrix} 0 & -a_{0,1} & \cdots & -a_{0,n} \\ 0 & & & \\ \vdots & & M & \\ 0 & & & \end{bmatrix},$$

where the first row includes the negated weights of the edges incident from r. Note that r is only a node that can be a root of the spanning trees of G_2. Thus, from Fact 1, we have that the sum of all products of weights on rooted spanning trees of G_2 is equal to $\det(\Delta_{G_1}(1|1)) = \det(M)$.

Finally, the digraph H mentioned above is obtained from G_2 by replacing each edge $e = (u, v)$ with weight two by a subgraph defined as follows: the subgraph consists of four nodes u, v, $[e, 1]$, and $[e, 2]$, and contains four edges $(u, [e, 1])$, $(u, [e, 2])$, $([e, 1], v)$, and $([e, 2], v)$. Then we can show that a rooted spanning tree T of H determines a unique rooted spanning tree T' of G_2 and conversely, the spanning tree T' of G_2 determines exactly w rooted spanning trees of H where w is the product of all weights on T'. For the tree T, T' can be determined by the following way: include in T' all edges in T which have weight one in G_2 (note that such all edges have remained in H by the above construction), and include in T' all edges $e = (u, v)$ with weight two if T contains either $([e, 1], v)$ or $([e, 2], v)$. To the tree T' in G_2, we correspond all the rooted spanning trees T of H satisfying the following conditions:

(1) for each edge with weight one in G_2, the edge is in T' iff the edge is in T,

(2) for each edge $e = (u, v)$ with weight two, if T' contains the edge, then T contains $(u, [e, 1])$, $(u, [e, 2])$, and one of $([e, 1], v)$ and $([e, 2], v)$; otherwise, T contains $(u, [e, 1])$ and $(u, [e, 2])$ but neither $([e, 1], v)$ and $([e, 2], v)$.

It is obvious that the number of such trees of H is equal to the product of all weights on T'. Thus we have that the number of rooted spanning trees of H is equal to $\det(M)$ from which we can obtain $\#\mathrm{PATH}_G(1, n)$ in polynomial-size and constant-depth. ∎

In what follows, we use P-uniform NC^1-reducibility. The P-uniformity comes from the use of integer division in our reduction below. It is shown in [BCH86] that the integer division can be done in a P-uniform family of NC^1 circuits, but it is unknown whether we can improve the uniformity bound. Also, we have not been able to find a way to avoid the division in the following result.

Theorem 2.3 The following problem is $\leq_{\rm cd\text{-}tt}$-reducible to INTDET and INTDET is (P-uniform) NC^1-reducible to the problem.

DIRECTED EULERIAN PATH

Given: a digraph G.

Compute: the number of Eulerian paths in G.

Proof. It is well known that G has an Eulerian path if and only if it is connected and the indegree of each node is the same as its outdegree (such a digraph is simply called *Eulerian*). These checks can be done in polynomial-size and constant-depth with oracle gates for MATPOW (note that in such a reduction, we use oracle gates for MATPOW not only for checking the connectivity of the graph but also for checking an equality between an indegree and an outdegree). Thus we below assume that G is Eulerian. Then we will use the following fact [BE51] that follows from Fact 1 mentioned in the last theorem (see [HP73] also).

Fact. [BE51] For an Eulerian digraph H (with n nodes), the number of Eulerian paths in the graph is given by

$$\det(\Delta_H(1|1)) \cdot \prod_{i=1}^{n}(d_i - 1)!,$$

where Δ_H is the same as in Fact 1 and d_i denotes the indegree of the ith node.

(End-of-FACT)

By this fact and by an argument similar to one in the reduction of ROOTED SPANNING TREE to MATPOW, it is easy to perform a $\leq_{\rm cd\text{-}tt}$-reduction of DIRECTED EULERIAN PATH to MATPOW (in fact, when ignoring the check of whether a given digraph is Eulerian, we can evaluate the above formula as a p-projection of MATPOW). We leave the detail to the interested reader.

We next perform a reduction of DIRECTED PATH to DIRECTED EULERIAN PATH, instead of reducing INTDET to the problem. The reduction is very similar to that of DIRECTED PATH to ROOTED SPANNING TREE, and hence we only sketch the reduction. Let G be a monotone digraph with n nodes given as an instance to DIRECTED PATH. As in the previous reduction, we here assume that every node of G is of both indegree and outdegree at most two. By a construction similar to the previous one, we can construct a digraph H with a root node r such that the number of spanning trees of H rooted at r is given by $\det(\Delta_H(1|1))$ and is equal to $2^{m_1} - 2^{m_2} \cdot \#\mathrm{PATH}_G(1, n)$ for some predetermined positive integers m_1 and m_2. Note that every node of H except r is of indegree either 1 or 2 and of outdegree at most two and its indegree is greater than or equal to its outdegree. Then we modify H as follows:

(1) for a node of indegree one and outdegree zero, add an edge from the node to r to H,

(2) for a node of indegree two and outdegree one, add an edge from the node to r to H, and

(3) for a node v of indegree two and outdegree zero, add two new nodes $[v, 1]$ and $[v, 2]$, and add four edges $(v, [v, 1])$, $(v, [v, 2])$, $([v, 1], r)$, $([v, 2], r)$ to H.

We refer to the resulting graph as H_1 and still consider r as its first node. Then it is easy to see from the modification that H_1 is Eulerian and the number of spanning trees of H_1 rooted at r is equal to that of H. Thus we have $\det(\Delta_H(1|1)) = \det(\Delta_{H_1}(1|1))$. The number of Eulerian paths in H_1 is given by $\det(\Delta_{H_1}(1|1)) \cdot (d_r - 1)!$, where d_r denotes the indegree of r in H_1 (note that all other nodes are of indegree at most two). From the result in [BCH86], we can obtain the factorization in P-uniform NC^1 and can also divide the above value by the factorization in P-uniform NC^1. This gives us a desired reduction. ∎

3 Concluding Remarks

In this paper, we exhibited four graph-theoretic counting problems that are computationally equivalent to computing the determinant of integer matrices. We also observed that the integer determinant problem remains its essential complexity even if we are concerned only with (0,1)-matrices as its instances. We have such a situation in some other linear algebraic problems such as matrix powering, iterated matrix product, matrix inversion (when we follow Cook [Coo85] in its definition), and the characteristic polynomial of a given integer matrix.

Our counting problems except SHORTEST PATH are concerned with directed graphs. We have not been able to establish similar relationships for the corresponding undirected cases. As a further question along the line of the present paper, it would be interesting to ask, given an undirected graph with n nodes, whether counting the number of different (not necessarily simple) paths of length at most n between a pair of nodes is computationally equivalent to the integer determinant and whether counting the number of spanning trees of the graph is equivalent to the integer determinant. Note that when we specify an upper bound on path length as a part of an instance to the first problem, there is no essential difference between the modified problem and SHORTEST PATH, that is, we can easily see that the two problems are $\leq_{\text{cd-tt}}$-equivalent to each other, and also note that if we require the simpleness to the paths that we count, the problem is known to be #P-complete [Val79b].

References

[AJ90] Carme Àlvarez and B. Jenner, A Very Hard Log Space Counting Class, *Proc. the 5th IEEE Conference on Structure in Complexity Theory* (1990), IEEE, New York, 154–168.

[Ber84] S.J. Berkowitz, On Computing the Determinant in Small Parallel Time Using a Small Number of Processors, *Inform. Process. Letters*, **18**(1984), 147–150.

[BE51] N. G. de Bruin and T. van Aardenne Ehrenfest, Circuits and Trees in Oriented Graphs, *Simon Stevin* **28**(1951), 203–217.

[BCH86] P. W. Beame, S. A. Cook and H. J. Hoover, Log Depth Circuits for Division and Related Problems, *SIAM J. Computing* **15**(1986), 994–1003.

[Coo85] S. A. Cook, A Taxonomy of Problems with Fast Parallel Algorithms, *Information and Control* **64**(1985), 2–22.

[CSV84] A. K. Chandra, L. Stockmeyer and U. Vishkin, Constant Depth Reducibility, *SIAM J. Computing* **13**(1984), 423–439.

[FSS81] M. Furst, J. Saxe and M. Sipser, Parity, Circuits, and the Polynomial-Time Hierarchy, *Proc. the 22nd IEEE Sympos. on Foundations of Computer Science* (1981), IEEE, 260–270.

[HP73] F. Harary and E. M. Palmer, Graphical Enumeration, Academic Press Inc., New York, 1973.

[SV85] S. Skyum and L. G. Valiant, A Complexity Theory Based on Boolean Algebra, *J. Assoc. Comput. Mach.* **32**(1985), 484–502.

[Tut48] W. T. Tutte, The Dissection of Equilateral Triangles into Equilateral Triangles, *Proc. Cambridge Philos. Soc.* **44**(1948), 463–482.

[Val79a] L.G. Valiant, Completeness Classes in Algebra, *Proc. 11th ACM Sympos. on the Theory of Computing*, 1979, 249–261.

[Val79b] L. G. Valiant, The complexity of reliability and enumeration problems, *SIAM J. Computing* **8** (1979), 410–421.

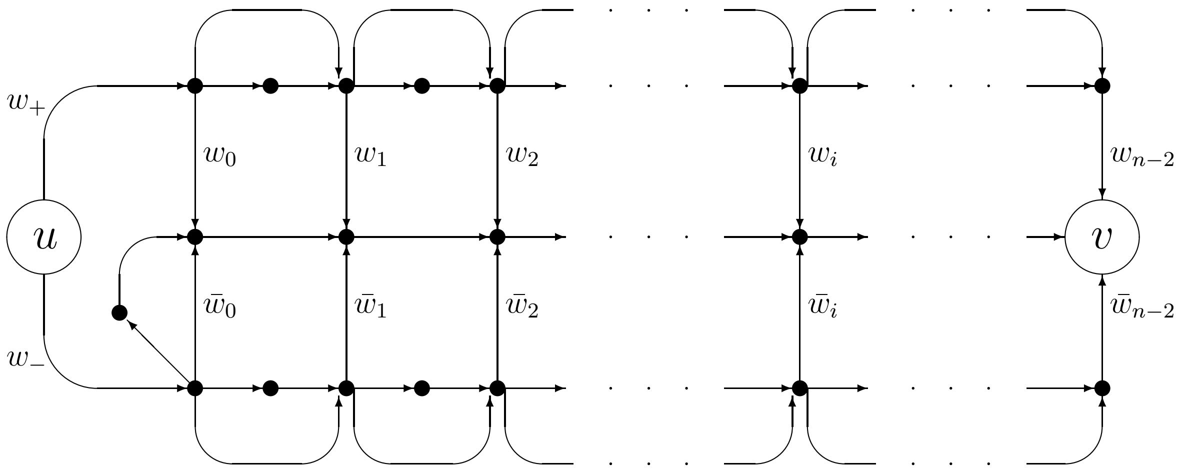

Figure 2.1. The subgraph replacing an edge with weight $w{=}w_{n-1}w_{n-2}\cdots w_0$ (in 2's complement form), where $w_+{=}1$ if $w_{n-1}{=}0$ and $w_+{=}0$ otherwise, and $w_- = -1$ if $w_{n-1}{=}1$ and $w_-{=}0$ otherwise. Note that $w = \sum_{\rm p}\{$product of all weights on p $\mid$ p is a path from u to v $\}$.