RANDOMIZED $\tilde{O}(M(|V|))$ ALGORITHMS FOR PROBLEMS IN MATCHING THEORY

JOSEPH CHERIYAN *

Abstract. A randomized (Las Vegas) algorithm is given for finding the Gallai-Edmonds decomposition of a graph. Let n denote the number of vertices, and let $M(n)$ denote the number of arithmetic operations for multiplying two $n \times n$ matrices. The sequential running time (i.e., number of bit operations) is within a poly-logarithmic factor of $M(n)$. The parallel complexity is $O((\log n)^2)$ parallel time using a number of processors within a poly-logarithmic factor of $M(n)$. The same complexity bounds suffice for solving several other problems:

(i) finding a minimum vertex cover in a bipartite graph,
(ii) finding a minimum $X \rightarrow Y$ vertex separator in a directed graph, where X and Y are specified sets of vertices,
(iii) finding the allowed edges (i.e., edges that occur in some maximum matching) of a graph, and
(iv) finding the canonical partition of the vertex set of an elementary graph.

The sequential algorithms for problems (i), (ii), and (iv) are Las Vegas, and the algorithm for problem (iii) is Monte Carlo. The new complexity bounds are significantly better than the best previous ones, e.g., using the best value of $M(n)$ currently known, the new sequential running time is $O(n^{2.38})$ versus the previous best $O(n^{2.5}/(\log n))$ or more.

Key words. randomized algorithms, matching theory, Gallai-Edmonds decomposition, allowed edges, canonical partition, bipartite minimum vertex covers, digraph minimum vertex separators

AMS(MOS) subject classifications. 68R10, 05C85, 05C50, 05C40, 05C70, 90C27

1. Introduction.

A *matching* of an undirected, possibly nonbipartite, graph $G = (V, E)$ is a subset E' of the edges such that no two of the edges in E' have a vertex in common. A *perfect matching* is one with cardinality $|V|/2$. Tutte [T 47] gave a good characterization of graphs that have perfect matchings, i.e., he showed that the perfect matching decision problem (decide whether or not a given graph has a perfect matching) is in NP $\cap$ co-NP. One of Tutte's innovations was to introduce the skew symmetric adjacency matrix $\tilde{B}$ of the graph G, defined as follows: Associate each edge ij of G with a distinct variable x_{ij}. Then $\tilde{B} = \tilde{B}(x_{ij})$ is a $|V| \times |V|$ matrix whose entries are given by

$$\tilde{B}_{ij} = \begin{cases} x_{ij} & \text{if } i > j \text{ and } ij \in E \\ -x_{ij} & \text{if } i < j \text{ and } ij \in E \\ 0 & \text{otherwise.} \end{cases}$$

Tutte observed that G has a perfect matching iff the determinant of $\tilde{B}(x_{ij})$, $\det(\tilde{B}(x_{ij}))$, is not identically zero; here, $\det(\tilde{B}(x_{ij}))$ is a polynomial in the variables x_{ij}. Lovász [Lo 79]

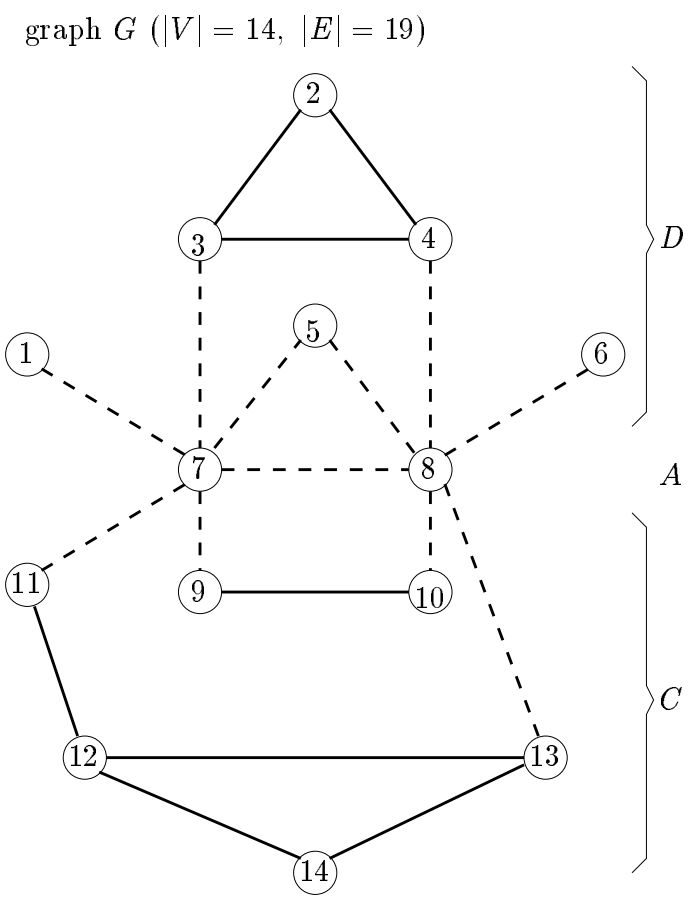
* Department of Combinatorics & Optimization, University of Waterloo, Ontario, Canada N2L 3G1. This research has been supported by NSERC grant no. OGP0138432 (NSERC code OGPIN 007), by a University of Waterloo faculty research grant, and by the Lucille and David Packard Fellowship of Éva Tardos.

used this observation to give an efficient randomized algorithm for the perfect matching decision problem: Choose a prime number $q = |V|^{O(1)}$, and substitute each variable x_{ij} in $\widetilde{B}$ by an independent random number drawn from $\{1, 2, \ldots, q-1\}$. Compute the determinant of the resulting random matrix B over the field of integers modulo q. With high probability (i.e., probability $\geq 1 - 1/\Omega(|V|)$, see Lemma 2.1), $\det(B) \neq 0 \mod q$ iff $\det(\widetilde{B}(x_{ij}))$ is not identically zero iff G has a perfect matching. This algorithm has two especially attractive features: it is simple, solving the decision problem by executing one "matrix operation", and it is efficient, running in sequential time $\widetilde{O}(M(|V|)) = O(|V|^{2.38})$ and in parallel time $O((\log |V|)^2)$ using $\widetilde{O}(M(|V|))$ processors. Here, $M(n)$ denotes the number of arithmetic operations for multiplying two $n \times n$ matrices, and is currently known to be $O(n^{2.376})$, see Coppersmith and Winograd [CW 90]. Throughout, the bounds on the sequential running time or on the number of parallel processors are stated for the arithmetic complexity model (uniform-cost RAM or PRAM), but they apply to the bit complexity model too, because for each arithmetic operation, comparison, or data transfer, each operand has $O(\log |V|)$ bits, hence the number of bit operations is at most $O((\log |V|)^2)$ times the number of arithmetic operations; see the last paragraph of Section 2.

The problem of *finding* a perfect matching of a graph G in time polynomial in $|V(G)|$ remained open till Edmonds [E 65] gave the first algorithm. Edmonds' algorithm solves a problem that is more general: For every graph G, the algorithm finds a matching of maximum cardinality in time $|V(G)|^{O(1)}$. One consequence of the algorithm is a theorem that was discovered independently by Gallai [Ga 64], the so-called Gallai-Edmonds theorem. According to this theorem, for every graph G, the vertex set can be partitioned in a unique way into three sets $A(G), C(G), D(G)$ such that certain properties hold (see Theorem 3.1). The partition gives much useful information, e.g., the cardinality of a maximum matching, the vertices that are incident to every maximum matching, etc. Several algorithms for constructing the partition are known. Edmonds' matching algorithm implicitly constructs the partition. Lovász (see [Kf 86, Section 2]) developed a randomized algorithm for finding the Gallai-Edmonds decomposition that runs in time $\widetilde{O}(|V|\, M(|V|))$; though there are faster algorithms for finding the decomposition, the algorithm of [Kf 86] is interesting for its simplicity. This paper (see Figure 1) presents a simple and efficient randomized algorithm for finding the Gallai-Edmonds decomposition: Lemma 3.3 shows that, with high probability, the partition $A(G) \cup C(G), D(G)$ for a given graph G can be found by computing a basis for the null space of a random skew symmetric adjacency matrix B, i.e., executing one "matrix operation" on B yields this partition. Obtaining the partition $A(G), C(G), D(G)$ from $A(G) \cup C(G), D(G)$ is trivial. The sequential running time is $\widetilde{O}(M(|V|))$ and the parallel time is $O((\log |V|)^2)$ using $\widetilde{O}(M(|V|))$ processors. Our algorithm is closely related to Lovász's algorithm (in [Kf 86]) for the Gallai-Edmonds decomposition; also, the algorithm uses a technique due to Eberly [E 91]. The algorithm, its proof, and running time analysis are all quite simple. Due to the information provided by the Gallai-Edmonds decomposition, our algorithm can be used to find a minimum cardinality vertex cover of a bipartite graph

within the same complexity bounds. The minimum cardinality bipartite vertex cover problem is equivalent to the problem of finding a minimum vertex separator for two given vertex sets X and Y in a directed graph (see Proposition 2.4), hence the directed graph problem can be solved within the same complexity bounds.

An edge of a graph G is called *allowed* if it occurs in at least one maximum cardinality matching. Consider the problem of finding the allowed edges. If G has a perfect matching, then the Gallai-Edmonds decomposition gives *no* information about the allowed edges because the partition $A(G), C(G), D(G)$ is trivial with $A(G) = \emptyset = D(G)$. Rabin and Vazirani




```
≫ for i = 1 : size(ADJ),
      for j = 1 : size(ADJ),
         B(i,j) = ADJ(i,j) * (i - j);
      end
   end
≫ rank(B)
ans = 12
% compute basis for null space of B
≫ N = null(B)
ans =

    0.1508    0.5161
    0.1877    0.2578
   -0.3753   -0.5156
    0.1877    0.2578
    0.2983   -0.5170
   -0.8228    0.2599
    0.0000    0.0000
    0.0000    0.0000
    0.0000    0.0000
    0.0000    0.0000
    0.0000    0.0000
    0.0000    0.0000
    0.0000    0.0000
    0.0000    0.0000
```

FIG. 1. *Finding the Gallai-Edmonds decomposition of an example graph G, using Lemma 3.3. The MATLAB code forms a pseudo-random skew symmetric adjacency matrix B from the adjacency matrix* ADJ *of G, by substituting $(i - j)$ for each nonzero* ADJ_{ij}. *With high probability, a vertex j is noncritical iff row j of the basis N of the null space of B is nonzero. The resulting partition, $A = \{v_7, v_8\}$, $C = \{v_9, \ldots, v_{14}\}$, $D = \{v_1, \ldots, v_6\}$, is shown. Note that each connected component of D has odd cardinality, and each connected component of C has even cardinality. Since* $\text{rank}(B) = 12 = |V| - (\sharp\text{components}(D) - |A|)$, *we have $\nu(G) = 6$. It follows that this partition gives the Gallai-Edmonds decomposition.*

[RV 89], in an elegant study of the random skew symmetric adjacency matrix B, observed that if $\det(B) \neq 0$ and the (i,j) minor of B (i.e., the determinant of the submatrix obtained from B by removing row i and column j) is nonzero, then the edge ij (if present) must be allowed. Moreover, all the (i,j) minors of B can be computed simultaneously by computing the inverse B^{-1}; the (j,i) entry of B^{-1} equals $(-1)^{i+j}/\det(B)$ times the (i,j) minor of B. Combining Rabin and Vazirani's method with our algorithm for the Gallai-Edmonds decomposition gives a randomized algorithm for finding the allowed edges of arbitrary graphs (see Section 3.3); the sequential running time is $\widetilde{O}(M(|V|))$ and the parallel time is $O((\log |V|)^2)$ using $\widetilde{O}(M(|V|))$ processors. We also give a randomized algorithm with the same complexity bounds for finding the canonical partition of an elementary graph, where a graph is called *elementary* if it has a perfect matching, and its allowed edges form a connected spanning subgraph (see Section 3.4).

Both Lovász's algorithm for the perfect matching decision problem and our algorithm for the Gallai-Edmonds decomposition are Monte Carlo, however, using results from matching theory, we show how to make both algorithms Las Vegas while achieving the same sequential and parallel complexity bounds. While analyzing our randomized algorithms, we assume that the random bits drawn by the execution have no effect on the sequential or parallel complexity; this assumption may not be appropriate in other contexts. More precisely, for the execution of a randomized algorithm on a *fixed input*, let us take the sequential running time (or parallel running time, or number of parallel processors) to be the maximum sequential running time (or maximum parallel running time, or maximum number of parallel processors) over all possible choices of the random bits. A randomized algorithm is said to be *Monte Carlo* if for a fixed input, an execution may give incorrect results with small probability. For a randomized algorithm and a problem instance of size n, an event is said to occur with *small* probability if the probability is $\leq 1/\Omega(n)$. A randomized algorithm is said to be *Las Vegas* if, for a fixed input, an execution either returns an output guaranteed to be correct, or reports failure, the latter with small probability. A Las Vegas algorithm may be trivially converted into a Monte Carlo algorithm without changing the complexity. To convert a Monte Carlo algorithm into a Las Vegas algorithm, we need a subroutine for verifying whether the output of the Monte Carlo algorithm is correct. If the complexity of the verifying subroutine is bounded by that of the Monte Carlo algorithm, then the Las Vegas and Monte Carlo algorithms have the same order of complexity. This raises a difficulty for our randomized algorithms: we need to verify the correctness of results for problems in matching theory within a complexity bound that is significantly less than that of the best algorithms known for finding a maximum cardinality matching (see the next paragraph). Fortunately, the partition of $V(G)$ computed by our randomized algorithm for the Gallai-Edmonds decomposition can be verified in sequential time $\tilde{O}(|E| + |V|)$ (or in parallel time $O((\log |V|)^2)$ using $\widetilde{O}(|E| + |V|)$ parallel processors). As a consequence, our algorithms for the Gallai-Edmonds decomposition, a minimum vertex cover of a bipartite graph, and a minimum vertex separator of a directed graph all can be made Las Vegas without affecting

the complexity. If the graph is bipartite, then our algorithm for finding the allowed edges can be made Las Vegas without affecting the complexity, but for nonbipartite graphs, we do not have a sufficiently efficient subroutine for verifying the allowed edges. Given an elementary graph, there is a sequential $\tilde{O}(|E| + |V|)$-time algorithm for verifying whether the partition computed by our Monte Carlo algorithm is a canonical partition, [L 95], so our sequential algorithm for the canonical partition can be made Las Vegas without affecting the complexity.

We briefly discuss the best sequential and parallel complexities known for computing a maximum cardinality (or, a perfect) matching. The fastest known sequential algorithms for finding a maximum matching due to Micali and Vazirani, Blum, and Gabow and Tarjan [MV 80, B 90, GT 91] (also see [V 94]) run in time $O(|E|\sqrt{|V|})$; for dense graphs this is $O(|V|^{2.5})$ time; these algorithms are deterministic, however, they are significantly slower than Lovász's randomized algorithm for the perfect matching decision problem. At present, all efficient (i.e., poly-logarithmic time) parallel algorithms for matching problems use randomization. The best parallel algorithms for finding a maximum matching are the Monte Carlo algorithms of Mulmuley, Vazirani and Vazirani [MVV 87] and Galil and Pan, and Karp, Upfal and Wigderson [GP 88, KUW 86]; the parallel complexities are $O((\log |V|)^2)$ time using $\tilde{O}(|V|\,|E|\,M(|V|))$ processors, and $O((\log |V|)^3)$ time using $\tilde{O}(|V|\,M(|V|))$ processors, respectively. Our parallel complexity bounds are stated for the Exclusive Read Exclusive Write (EREW) PRAM model. Efficient parallel Las Vegas algorithms for matching problems have been designed by Karloff [Kf 86] and Wein [W 91].

It turns out that our algorithm for finding a minimum vertex separator for two given vertex sets X and Y in a directed graph can be developed independently of matching theory, and this is done in Section 4, building on the work by Linial, Lovász and Wigderson, and Cheriyan and Reif [LLW 88, CR 94]. Preliminary versions of the results on computing the Gallai-Edmonds decomposition and directed graph $X \rightarrow Y$ separators have appeared in [C 94] and [C 93], respectively.

Section 2 has notation, definitions, and preliminary results. Section 3 develops the algorithms for problems in matching theory. Section 4 is independent of Section 3, and develops an algorithm for a minimum $X \rightarrow Y$ separator in a directed graph. Finally, Section 5 has conclusions. The appendix contains some proofs.

2. Preliminaries. For the given graph $G = (V, E)$, we use n and m to denote the number of vertices and edges, i.e., $n = |V|$ and $m = |E|$. For a subset X of V, $\overline{X}$ denotes $V - X$. For a matrix A with row and column indices from V and two subsets X, Y of V, $A(X, Y)$ denotes the submatrix of A formed by the X-rows and the Y-columns. The vector with a one in position j and zeros elsewhere is denoted by e_j, and $\left[\begin{smallmatrix} A \\ e_j \end{smallmatrix}\right]$ denotes the $(n+1) \times n$ matrix formed by adding the $(n+1)$th row e_j to an $n \times n$ matrix A.

A few standard definitions from matching theory are needed, see [LP 86]. An *odd (even) component* of a graph is a connected component whose vertex set has odd (even) cardinality.

A *vertex cover* of a graph $G = (V, E)$ is a vertex set $C \subseteq V$ such that each edge is incident with some vertex of C. Given a graph $G = (V, E)$ and a matching E', a vertex is called *matched* if it is incident to an edge of E', and is called *exposed* otherwise. A *near perfect matching* is one with exactly one exposed vertex. For a graph G, $\nu(G)$ denotes the number of edges of a maximum matching. The *deficiency* of G is the number of vertices exposed in a maximum matching, $n - 2\nu(G)$. A vertex x is called *noncritical* if it is exposed in at least one maximum matching, otherwise x is called *critical*. Equivalently, x is noncritical if $\nu(G - x) = \nu(G)$, and is critical if $\nu(G - x) < \nu(G)$. A graph H is called *factor critical* if for each of its vertices x, $H - x$ has a perfect matching.

The following lemma due to Zippel and Schwartz [Z 79, Sc 80], see also [Ko 91, Corollary 40.2], is useful for estimating the failure probability of a whole class of randomized algorithms.

LEMMA 2.1 (ZIPPEL-SCHWARTZ). *If $p(x_1, x_2, \ldots, x_m)$ is a nonzero polynomial of degree d with coefficients in a field and S is a subset of the field, then the probability that p evaluates to zero on a random element $(s_1, s_2, \ldots, s_m) \in S^m$ is at most $d/|S|$.*

Recall from Section 1 the definition of the skew symmetric adjacency matrix $\widetilde{B} = \widetilde{B}(x_{ij})$ of a graph G and Tutte's observation that $\det(\widetilde{B})$ is not identically zero iff G has a perfect matching. Lovász generalized this observation; for a proof, see [RV 89].

PROPOSITION 2.2 (LOVÁSZ). *Let $\widetilde{B} = \widetilde{B}(x_{ij})$ be the skew symmetric adjacency matrix of a graph G. Then* $\mathrm{rank}(\widetilde{B}) = 2\nu(G)$.

A random skew symmetric adjacency matrix B is obtained by substituting the variables x_{ij} in $\widetilde{B}(x_{ij})$ by independent random numbers w_{ij} from a subset $\{1, \ldots, W\}$ of a field. The next result is due to Lovász [Lo 79], and follows from the previous one by applying the Zippel-Schwartz Lemma.

PROPOSITION 2.3 (LOVÁSZ). *Let $B = \widetilde{B}(w_{ij})$ be a random skew symmetric adjacency matrix of a graph G, where the w_{ij} are independent random numbers from $\{1, \ldots, W\}$. Then* $\mathrm{rank}(B) \leq 2\nu(G)$, *and with probability at least* $1 - (n/W)$, $\mathrm{rank}(B) = 2\nu(G)$.

Given a digraph (directed graph) $G = (V, E)$ and a pair of subsets X and Y of the vertices, an $X \rightarrow Y$ *(vertex) separator* is a set of vertices S such that $G - S$ has no path from a vertex in $X - S$ to a vertex in $Y - S$. For a pair of subsets X and Y of the vertices, $p(X, Y)$ denotes the maximum number of vertex disjoint paths from X to Y (any two of these paths have no vertices in common, not even the terminal vertices). Clearly, every $X \rightarrow Y$ separator has cardinality at least $p(X, Y)$. Menger's theorem states that for every pair of subsets X and Y of the vertices, there exists an $X \rightarrow Y$ separator with cardinality $p(X, Y)$. We call an

$X \to Y$ separator *minimum* if its cardinality is minimum, namely, $p(X, Y)$.

Let us call two problems *linear-time equivalent* if there is a linear-time algorithm to transform an instance of the first problem to an instance of the second such that a solution to the second instance can be transformed in linear time to a solution of the first instance, and vice versa. Part (i) of the next proposition is well known. The novel point of part (ii) is that a digraph minimum vertex separator can be obtained in linear time from an arbitrary minimum vertex cover of an appropriately constructed bipartite graph. The appendix has a proof of the proposition.

PROPOSITION 2.4.

(i) The problem of finding a maximum cardinality matching in a bipartite graph is linear-time equivalent to the problem of finding a maximum cardinality set of vertex-disjoint $X \to Y$ paths in a digraph.

(ii) The problem of finding a minimum vertex cover in a bipartite graph is linear-time equivalent to the problem of finding a minimum $X \to Y$ separator in a digraph.

We use the soft-Oh notation to denote the complexity of algorithms. The soft-Oh notation drops poly-logarithmic factors: for functions f and g, f is $\tilde{O}(g)$ iff there are constants $n_0,\ k \geq 0$ such that $f(n) \leq g(n)(\log n)^k$, for all $n \geq n_0$. Note that $\tilde{O}(M(n)) = O(n^{2.38})$, since $M(n)$ is known to be $O(n^{2.376})$ (see Section 1). All computations of the algorithms presented below are over the field $\mathbb{Z}_q$ of integers modulo a prime number q. When choosing random numbers w, we assume that they are drawn from the uniform distribution over $\{1, \ldots, W\}$, where W is an integer and $W < q$. Throughout, we take $q = |V(G)|^{O(1)}$, i.e., q is polynomially bounded in the number of vertices of the graph G. Consider the number of bit operations for multiplying two $|V| \times |V|$ matrices over the field of integers modulo q. Since an integer modulo q can be represented using $O(\log q) = O(\log |V|)$ bits, it follows that the number of bit operations is $\tilde{O}(M(|V|))$.

3. Randomized algorithms for problems in matching theory. This section develops randomized $\tilde{O}(M(|V|))$-time algorithms for the following problems in matching theory: finding a Gallai-Edmonds decomposition, finding a minimum vertex cover in a bipartite graph, finding the allowed edges of a graph, and finding the canonical partition of an elementary graph.

3.1. A randomized algorithm for the Gallai-Edmonds decomposition. Recall that a vertex x is called *noncritical* if it is exposed in at least one maximum matching, otherwise x is called *critical*. We use $D(G)$ to denote the set of noncritical vertices, and $A(G)$ to denote the set of vertices in $V(G) - D(G)$ adjacent to vertices of $D(G)$. The set of remaining vertices, $V(G) - (D(G) \cup A(G))$, is denoted by $C(G)$. For ease of notation, $D(G)$ and $C(G)$ are also used to denote the subgraphs of G induced by the respective vertex sets.

See [LP 86, Theorem 3.2.1] for a proof of the next theorem.

THEOREM 3.1 (GALLAI-EDMONDS THEOREM). *Let G be a graph, and let $D(G)$, $A(G)$ and $C(G)$ be as defined above. Then*

(i) each component of the subgraph induced by $C(G)$ has a perfect matching;

(ii) each component of the subgraph induced by $D(G)$ is factor critical;

(iii) the deficiency of G equals

$$\sharp\text{components}(D(G)) - |A(G)|,$$

where $\sharp$components$(D(G))$ denotes the number of connected components in the subgraph induced by $D(G)$;

(iv) every maximum matching of G contains a perfect matching of each component of $C(G)$, a near perfect matching of each component of $D(G)$, and matches all the vertices of $A(G)$ with vertices in distinct components of $D(G)$.

The key result for our algorithm follows. Recall the notation $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$ from Section 2.

LEMMA 3.2. *Let $B = \widetilde{B}(w_{ij})$ be a random skew symmetric adjacency matrix of a graph G, where the w_{ij} are independent random numbers from $\{1, \ldots, W\}$. Consider any vertex x, and let j be its index in B.*

(i) If x is noncritical, then with probability at least $1 - (2n/W)$, the rank of the matrix $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$ is greater than that of B.

(ii) If x is critical, then with probability at least $1 - (n/W)$, the rank of the matrix $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$ equals that of B.

Proof. Consider the augmented graph G' and its random skew symmetric adjacency matrix B', where G' is obtained from G by adding a new vertex z (with index $n+1$) and the edge xz, and B' is obtained from B by adding a row $r \cdot e_j$ and corresponding column, where r is a random number independent of the entries of B, i.e.,

$$B' = \begin{bmatrix} B & & & & 0 \\ & & & & \vdots \\ & & & & -r \\ & & & & \vdots \\ 0 & \ldots & r & \ldots & 0 \end{bmatrix}.$$

Consider the cardinality of a maximum matching of G'. If there exists a maximum matching of G with x exposed, then $\nu(G')$ is greater than $\nu(G)$ because the new edge xz of G' may be added to the maximum matching of G. However, if x is matched in every maximum matching of G, then $\nu(G')$ equals $\nu(G)$. In other words, x is noncritical in G iff $\nu(G')$ is

greater than $\nu(G)$. Applying Proposition 2.3 to G' and B', we see that if x is noncritical in G, then with probability at least $1-(2n/W)$,

$$\operatorname{rank}(B') = 2\nu(G') = 2\nu(G) + 2 = \operatorname{rank}(B) + 2.$$

Consider the matrix $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$ obtained from B' by removing the last column, and then dividing the last row by r. Part (i) of the lemma follows since $\operatorname{rank}(\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]) \geq \operatorname{rank}(B') - 1$.

For part (ii), we have seen that $\nu(G')$ equals $\nu(G)$ if x is critical. Hence, with probability at least $1-(n/W)$, $\operatorname{rank}(B) = 2\nu(G) = 2\nu(G') \geq \operatorname{rank}(B') \geq \operatorname{rank}(\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]) \geq \operatorname{rank}(B)$. □

ALGORITHM 1. *Monte Carlo Gallai-Edmonds Decomposition*
INPUT: Graph $G = (V, E)$.
OUTPUT: With high probability, the Gallai-Edmonds decomposition of G.
Step 0:
Order the vertices, and number them $1, 2, \ldots, n$.
Fix the number $W = n^{O(1)}$, and choose a prime q, $W < q = n^{O(1)}$.
For each edge ij, choose a random weight $w(ij) \in \{1, 2, \ldots, W\}$.
Construct a random skew symmetric adjacency matrix B of G,
where for each edge ij, $i > j$,
$B_{ij} = w(ij)$ and $B_{ji} = -w(ij)$ ($B_{ij} = 0$ if ij is not an edge).
Step 1:
Compute the rank r of B over the field $\mathbb{Z}_q$.
Step 2:
For each of the vectors e_j, $j = 1, \ldots, n$,
compute the rank r_j of the matrix $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$ over the field $\mathbb{Z}_q$.
Let D be the set of vertices j with $r_j > r$.
Step 3:
Let A be the subset of $V - D$ adjacent to D, and let C be the set
of vertices neither in D nor in A.
With high probability, the Gallai-Edmonds decomposition of G
is given by A, C, D.

FIG. 2.

The algorithm for finding the Gallai-Edmonds decomposition follows straightaway from the previous lemma and Theorem 3.1. Find the set $D(G)$ of noncritical vertices with high probability by comparing the rank of each $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$, $j = 1, \ldots, n$, with the rank of B. The probability that the set $D(G)$ is correctly computed is at least $1-(2n^2/W)$. Knowing $D(G)$, the sets $A(G)$ and $C(G)$ can be found in $\widetilde{O}(n+m)$ time. See Algorithm 1 in Figure 2 for a full description. The working of the algorithm on an example is illustrated in Figure 1. A straightforward implementation of Algorithm 1 runs in $\widetilde{O}(n \cdot M(n)) = O(n \cdot n^{2.38})$ time:

for each $j = 1, \ldots, n$, use the algorithm of [IMH 82] to find the rank of $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$ in $\widetilde{O}(M(n))$ time. We now improve the running time from $\widetilde{O}(n \cdot M(n))$ to $\widetilde{O}(M(n))$. The $\widetilde{O}(M(n))$ bound holds even for the number of bit operations; to see this, recall the remarks at the end of Section 2. To obtain a faster implementation, observe that $\operatorname{rank}(\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right])$ is greater than $\operatorname{rank}(B)$ iff e_j is *not* in the row space of B, i.e., iff e_j is not a linear combination of the row vectors of B. We can "simultaneously" compute the ranks of all the $\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right]$'s by computing a matrix N such that for any row vector v, $v \cdot N = 0$ iff v is a linear combination of the row vectors of B. Once N is computed, we simply find the product of the $n \times n$ identity matrix I_n with N. The nonzero rows of N correspond exactly to the vectors e_j having $\operatorname{rank}(\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right])$ greater than $\operatorname{rank}(B)$. Coming to the computation of N, we take N to be a basis for the null space (i.e., kernel) of B. Note that for any subspace U (e.g., the row space of B) of a finite-dimensional vector space W over a finite field (e.g., the n-dimensional vector space over $\mathbb{Z}_q$), $\dim U + \dim U^- = \dim W$, and so $(U^-)^- = U$ [Lo, Exercise 5.31]. Hence, even for the n-dimensional vector space over $\mathbb{Z}_q$, we can check whether a vector v is in the row space of B by checking whether $v \cdot N$ is zero. It is well known that for any $n \times n$ matrix B, a basis for the null space can be computed in sequential time $\widetilde{O}(M(n))$, see [IMH 82, pp. 53–54], and in randomized parallel time $O((\log n)^2)$ using $\widetilde{O}(M(n))$ processors, see [KP 91, p. 190].

LEMMA 3.3. *Let $B = \widetilde{B}(w_{ij})$, $w_{ij} \in \{1, \ldots, W\}$, be a random skew symmetric adjacency matrix of a graph, and let the $n \times (n - \operatorname{rank}(B))$ matrix N be a basis for the null space of B. Let v be an arbitrary vertex, and let j be its index in B. If v is critical (noncritical), then with probability at least $1 - (2n/W)$, the jth row of N is zero (nonzero).*

Let A, C, D denote the partition of V computed by an execution of the Monte Carlo algorithm. To make the algorithm Las Vegas, we need to verify whether A, C, D is the Gallai-Edmonds decomposition. We first verify whether $\nu(G)$ is computed correctly, and then verify whether the set D equals the set of noncritical vertices, $D(G)$. By Proposition 2.3, $\nu(G)$ is at least $\operatorname{rank}(B)/2$, where B is the random skew symmetric adjacency matrix. Suppose that each component of D is odd and each component of C is even. Then $\nu(G)$ is at most $(|V| - (\sharp\text{components}(D) - |A|))/2$, because every matching E' of G leaves at least $\sharp\text{components}(D) - |A|$ exposed vertices: to see this, observe that for each odd component of $G - A$, either the odd component contains an exposed vertex, or an edge of E' matches a vertex of the odd component to a vertex of A. Our verification subroutine (in the Las Vegas algorithm) determines the odd and even components of $G - A$, and compares $\operatorname{rank}(B)$ with $|V| - (\sharp\text{components}(D) - |A|)$. If equality fails to hold in the comparison, or one of the components of D is even, or one of the components of C is odd, then the Las Vegas algorithm reports failure. Otherwise, $\nu(G)$ is guaranteed to equal $\operatorname{rank}(B)/2$, and moreover, the set A is guaranteed to be a barrier. A set $X \subseteq V(G)$ is called a *barrier* if $|V| - 2\nu(G)$ (i.e., the deficiency of G) equals the difference of the number of odd components of $G - X$

and $|X|$. We claim that if A is a barrier and $\nu(G) = \text{rank}(B)/2$, then the computed partition A, C, D is the Gallai-Edmonds decomposition. To see this, note that if a vertex with index j has $\text{rank}(\left[\begin{smallmatrix} B \\ e_j \end{smallmatrix}\right])$ greater than $\text{rank}(B) = 2\nu(G)$, then the vertex is noncritical; hence, every vertex in the computed set D is noncritical. Also, every noncritical vertex is contained in D by the following theorem (see [LP 86, Theorem 3.3.17]): if $X \subseteq V(G)$ is a barrier, then every noncritical vertex is contained in the union of the odd components of $G - X$. Consequently, $D = D(G)$, and so, by construction, $A = A(G)$ and $C = C(G)$.

THEOREM 3.4. *There is a Las Vegas algorithm with a sequential running time of $\widetilde{O}(M(n))$ for finding the Gallai-Edmonds decomposition and the cardinality of a maximum matching of a graph. A parallel version of the algorithm uses $\widetilde{O}(M(n))$ processors and takes parallel time $O((\log n)^2)$.*

3.2. Finding a minimum vertex cover in a bipartite graph. Due to the information provided by the Gallai-Edmonds decomposition, the above Las Vegas algorithm may be applied to solve other problems in matching theory within the same complexity bounds. In this subsection, we show how the algorithm may be used to find a minimum vertex cover of a bipartite graph. Moreover, by the equivalence of the bipartite minimum vertex cover problem and the digraph minimum $X \rightarrow Y$ separator problem, see Proposition 2.4, we can also find a minimum $X \rightarrow Y$ separator in a digraph. We need a theorem from matching theory, see [LP 86, Theorem 3.2.4].

THEOREM 3.5 (DULMAGE AND MENDELSOHN). *Let $G = (V_1, V_2, E)$ be a bipartite graph, where V_1 and V_2 are the sets of the vertex bipartition. For $i = 1, 2$ let $A_i = A(G) \cap V_i$, $C_i = C(G) \cap V_i$, and $D_i = D(G) \cap V_i$, where $A(G)$, $C(G)$, and $D(G)$ are the three sets of the Gallai-Edmonds decomposition of G. Then $C_1 \cup A_1 \cup A_2$ and $C_2 \cup A_1 \cup A_2$ are minimum vertex covers.*

The above theorem combined with the Las Vegas algorithm for the Gallai-Edmonds decomposition immediately yields an efficient Las Vegas algorithm for a minimum vertex cover of a bipartite graph $G = (V_1, V_2, E)$. The algorithm may be simplified by focusing on just one of the sets V_i, $i = 1, 2$, of the vertex bipartition, and for each vertex in that set computing whether it is critical or not. Also, instead of the skew symmetric adjacency matrix, we use the bipartite adjacency matrix H, which has a row for each vertex in V_1, and a column for each vertex in V_2; an entry H_{ij} is nonzero iff G has the edge ij, $i \in V_1, j \in V_2$. See Algorithm 2 in Figure 3.

THEOREM 3.6. *There is a Las Vegas algorithm with a sequential running time of $\widetilde{O}(M(n))$ for finding a minimum cardinality vertex cover of a bipartite graph. A parallel version of the algorithm uses $\widetilde{O}(M(n))$ processors and takes parallel time $O((\log n)^2)$. The*

ALGORITHM 2. *Monte Carlo Bipartite Minimum Vertex Cover*
INPUT: Bipartite graph $G = (V_1, V_2, E)$.
OUTPUT: With high probability, a minimum vertex cover of G.
Step 0:
Order the vertices of V_1 and V_2, and number them $1, 2, \ldots$
Fix the number $W = n^{O(1)}$, and choose a prime q, $W < q = n^{O(1)}$.
For each edge ij, $i \in V_1, j \in V_2$, choose a random weight $w(ij) \in \{1, 2, \ldots, W\}$.
Construct a random bipartite adjacency matrix H of G,
where for each edge ij, $i \in V_1, j \in V_2$, $H_{ij} = w(ij)$ ($H_{ij} = 0$ if ij is not an edge).
Step 1:
Compute the rank r of H over the field $\mathbb{Z}_q$.
Step 2:
For each of the vectors e_j, $j = 1, \ldots, |V_2|$,
compute the rank r_j of the matrix $\begin{bmatrix} H \\ e_j \end{bmatrix}$ over the field $\mathbb{Z}_q$.
Let $D_2 \subseteq V_2$ be the set of vertices $j \in V_2$ with $r_j > r$.
Step 3:
Let A_1 be the subset of V_1 adjacent to D_2, i.e.,
$A_1 = \{i \in V_1 : ij \in E \text{ and } j \in D_2\}$.
With high probability, a minimum vertex cover of G is given by
$A_1 \cup (V_2 - D_2)$.

FIG. 3.

same complexity bounds apply for finding a minimum cardinality $X \rightarrow Y$ separator of a digraph.

In Section 4, an algorithm for finding minimum $X \rightarrow Y$ separators in digraphs (and for finding bipartite minimum vertex covers) is designed using different methods than those used in this section. Yet, it turns out that the two algorithms for bipartite minimum vertex covers are identical.

3.3. Finding the allowed edges. Recall from Section 1 that an edge of a graph $G = (V, E)$ is called *allowed* if it is contained in at least one maximum matching. The notion of an allowed edge is important in matching theory, see [LP 86, Chapter 5]. We develop a Monte Carlo algorithm for finding the set of allowed edges of an arbitrary graph; the sequential and parallel complexities are the same as those of our algorithm in Theorem 3.4. The best previous sequential or parallel algorithms for finding the set of allowed edges of a graph take at least as much sequential time (or, parallel time and parallel processors) as needed for computing a maximum matching.

Our method for finding the set of allowed edges first constructs the Gallai-Edmonds decomposition $A(G), C(G), D(G)$ using the algorithm of Theorem 3.4. Now, observe that every edge incident to a noncritical vertex v is allowed: consider a maximum matching such that

v is exposed, and switch the matching by adding any edge vw and removing the matched edge incident to w. Secondly, every edge with one end vertex in $A(G)$ and the other in either $A(G)$ or $C(G)$ is *not* allowed, by Theorem 3.1. Finally, we are left with the edges with both end vertices in $C(G)$. Since every component of $C(G)$ has a perfect matching, we apply the following result of Rabin and Vazirani, see [RV 89, Lemma 4], to find (with high probability) the allowed edges of components of $C(G)$.

LEMMA 3.7 (RABIN AND VAZIRANI). *Let G be a graph with a perfect matching, and let B be a random skew symmetric adjacency matrix of G. If $\det(B) \neq 0$, then for each index i, $1 \leq i \leq n$, there is an index j, $1 \leq j \leq n$, such that $B_{ij} \neq 0$ and $(B^{-1})_{ji} \neq 0$; moreover, for each pair i, j satisfying this condition, the corresponding edge $v_i v_j$ is in some perfect matching of G.*

THEOREM 3.8. *With probability at least $1 - (1/n^{\Theta(1)})$, the set of allowed edges of a graph can be computed in sequential time $\tilde{O}(M(n))$, and in parallel time $O((\log n)^2)$ using $\tilde{O}(M(n))$ processors.*

The above algorithm is Monte Carlo, but not Las Vegas: if the algorithm reports that an edge with both end vertices in $C(G)$ is not allowed, then there is a small probability that the edge is actually allowed. In all other cases, the algorithm's output is correct. For the special case of bipartite graphs, we give a Las Vegas algorithm that achieves the same complexity bounds. Focus on a component $H = (V_1, V_2, E)$ of $C(G)$, where V_1 and V_2 are the sets of the vertex bipartition. We construct the connected subgraphs $H_1, \ldots, H_k$ of H formed by the computed set of allowed edges. For each connected subgraph H_i, $1 \leq i \leq k$, $|V(H_i) \cap V_1|$ must equal $|V(H_i) \cap V_2|$, and each edge with both end vertices in H_i must be allowed, see [LP 86, Theorem 4.1.1], otherwise, the algorithm reports failure. Next, we construct a bipartite graph H' by contracting to a distinct single vertex each of the two sets in the vertex bipartition of each of the connected subgraphs $H_1, \ldots, H_k$, i.e., each $V(H_i) \cap V_j$, $1 \leq i \leq k$, $j = 1, 2$, is contracted to a distinct vertex. Also, we replace any parallel edges by single edges. Thus, each H_i, $1 \leq i \leq k$, is contracted to a distinct edge; observe that these "contracted edges" form a perfect matching of the contracted graph H'. If the contracted graph H' has a unique perfect matching, then the computed set of allowed edges of H is correct; otherwise, the algorithm reports failure; see [LP 86, Lemma 4.3.1]. To test for a unique perfect matching in H', we start with the perfect matching consisting of the edges formed by contracting $H_1, \ldots, H_k$, and check whether there exists an alternating cycle with respect to this matching. The claimed complexity bounds suffice for testing for an alternating cycle.

THEOREM 3.9. *There is a Las Vegas algorithm with a sequential running time of $\tilde{O}(M(n))$ for finding the set of allowed edges of a bipartite graph. A parallel version of the algorithm uses $\tilde{O}(M(n))$ processors and takes parallel time $O((\log n)^2)$.*

3.4. Finding the canonical partition of an elementary graph. Recall from Section 1 that a graph $G = (V, E)$ is called elementary if it has a perfect matching and its allowed edges form a connected spanning subgraph. Also, recall that a set X of vertices is called a barrier if the deficiency of G, $|V| - 2\nu(G)$, equals the difference of the number of odd components of $G - X$ and $|X|$. For an elementary graph, the deficiency is zero, so $X \subseteq V$ is a barrier iff $|X|$ equals the number of odd components of $G - X$. If G is elementary, then the (inclusionwise) maximal barriers of G form a partition $S_1, S_2, \ldots, S_k$ of the vertex set $V(G)$; this partition is called the *canonical partition* [LP 86, Section 5.2]. Here, we develop an efficient randomized algorithm for finding the canonical partition of an elementary graph. The Monte Carlo algorithm for the canonical partition was discovered jointly with K. Padayachee. The sequential $\tilde{O}(m + n)$-time algorithm for verifying the canonical partition is due to J. A. La Poutré, [L 95].

The key result underlying our algorithm is this, see [LP 86, Theorem 5.2.2]: Two (distinct) vertices x and y are in the same set S_i of the canonical partition iff $G - \{x, y\}$ has *no* perfect matching. Based on this result and [RV 89, Lemma 3], we find the canonical partition as follows: Assume that the given graph G is elementary. We construct a random skew symmetric adjacency matrix B of G. If $\det(B) = 0$, then we stop and report failure. Otherwise, we compute the inverse of B, B^{-1}. To compute the canonical partition of $V(G)$, we attempt to construct an equivalence relation Ψ on the vertex pairs such that vertices x and y are related iff the (x, y) entry of B^{-1}, $(B^{-1})_{xy}$, is zero. If Ψ is indeed an equivalence relation, then the algorithm outputs the equivalence classes of Ψ as the computed partition (with high probability, this is the canonical partition); otherwise, if Ψ is not an equivalence relation, then we stop and report failure. Verifying that Ψ is an equivalence relation and computing its equivalence classes takes sequential time $\tilde{O}(n^2)$ and parallel time $O((\log n)^2)$ using $\tilde{O}(n^2)$ processors: first, observe that Ψ is reflexive ($(x, x) \in \Psi, \forall x \in V$) and symmetric ($(x, y) \in \Psi$ iff $(y, x) \in \Psi$) since B is skew symmetric and n is even; Ψ is transitive iff each connected component of the graph (V, Ψ) is a clique.

THEOREM 3.10 (WITH K. PADAYACHEE). *With probability at least* $1 - (1/n^{\Theta(1)})$, *the canonical partition of an elementary graph can be computed in sequential time* $\tilde{O}(M(n))$, *and in parallel time* $O((\log n)^2)$ *using* $\tilde{O}(M(n))$ *processors.*

The above algorithm is Monte Carlo, but not Las Vegas. For this paragraph, let $S_1, \ldots, S_k$ denote the partition computed by the Monte Carlo algorithm; the canonical partition may differ from $S_1, \ldots, S_k$. If each set S_i, $1 \leq i \leq k$, is a barrier, then the computed partition is the canonical partition. To see this, observe that for any two vertices x and y in two different sets of the computed partition, the (x, y) entry of B^{-1} is nonzero, hence, by [RV 89, Lemma 3], the graph $G - \{x, y\}$ has a perfect matching. Consequently, by the key result on canonical partitions quoted above, x and y must be in different sets of the canonical partition. Hence, the canonical partition is a refinement of the partition $S_1, \ldots, S_k$, and if

the two partitions differ, then one of the sets S_i is the union of two or more maximal barriers. To obtain a Las Vegas algorithm we do the following: For each set S_i in the computed partition, we determine whether it is a barrier by comparing the number of odd components of $G - S_i$ with $|S_i|$. If every set S_i is a barrier, then we output $S_1, \ldots, S_k$ as the canonical partition, otherwise, we report failure. There is a sufficiently efficient sequential algorithm due to La Poutré, [L 95], for the key computation in verifying the partition computed by our Monte Carlo algorithm; this algorithm uses Sleator and Tarjan's [ST 83] dynamic trees data structure to maintain the connected components of the current subgraph, and works by appropriately deleting and inserting all edges incident to vertices in the set S_i, $1 \leq i \leq k$; also, see La Poutré and Westbrook [LW 94].

THEOREM 3.11 (J. A. LA POUTRÉ). *Given a graph $G = (V, E)$ and a collection of pairwise disjoint vertex sets $S_1, \ldots, S_k$, the number of odd components in $G - S_i$ for all i, $1 \leq i \leq k$, can be determined in (deterministic) sequential time $\widetilde{O}(m + n)$.*

Alternatively, the sequential $\widetilde{O}(m+n)$ time bound can be achieved by a randomized Las Vegas algorithm using dynamic data structures recently developed by Rauch-Henzinger and King, [HK 95].

THEOREM 3.12. *Given an elementary graph as input, there is a sequential Las Vegas algorithm with a running time of $\widetilde{O}(M(n))$ for finding the canonical partition.*

4. An algorithm for digraph minimum $X \rightarrow Y$ separators. Using methods different from the ones employed in the previous section, this section develops a randomized Monte Carlo algorithm for finding a minimum $X \rightarrow Y$ separator of a given digraph $G = (V, E)$, where X and Y are specified sets of vertices. On every instance, the algorithm outputs a correct solution with high probability, but it may output an incorrect result with small probability. The complexity bounds of this Monte Carlo algorithm suffice for verifying that the computed solution is indeed a minimum $X \rightarrow Y$ separator, thus giving a Las Vegas algorithm with the same complexity bounds.

For an $X \rightarrow Y$ separator S with $|S| = p(X, Y)$, let $T(S)$ denote the set of vertices such that $G - S$ has a path from each vertex in $T(S)$ to at least one vertex in $Y - S$ (note that $T(S)$ is empty iff $S = Y$). Informally, $T(S)$ forms the "Y-side" of the separator S.

First, consider the simple case when the digraph has a *unique* minimum $X \rightarrow Y$ separator S, i.e., any other set of vertices whose removal from G leaves no $X \rightarrow Y$ paths has cardinality at least $|S| + 1$. See Figure 4. Then the set $T = T(S)$ has the key property that a vertex v is in T if and only if $p(\{v\} \cup X, Y)$ is greater than $p(X, Y)$. To see this, deduce from Menger's theorem that G must have a separator S' of cardinality $p(\{v\} \cup X, Y)$ whose removal from G leaves no path from $(\{v\} \cup X)$ to Y. Clearly, S' is an $X \rightarrow Y$ separator too, so $p(\{v\} \cup X, Y) \geq p(X, Y)$. If $v \notin T$, take the separator S' to be S, since $G - S$ has no

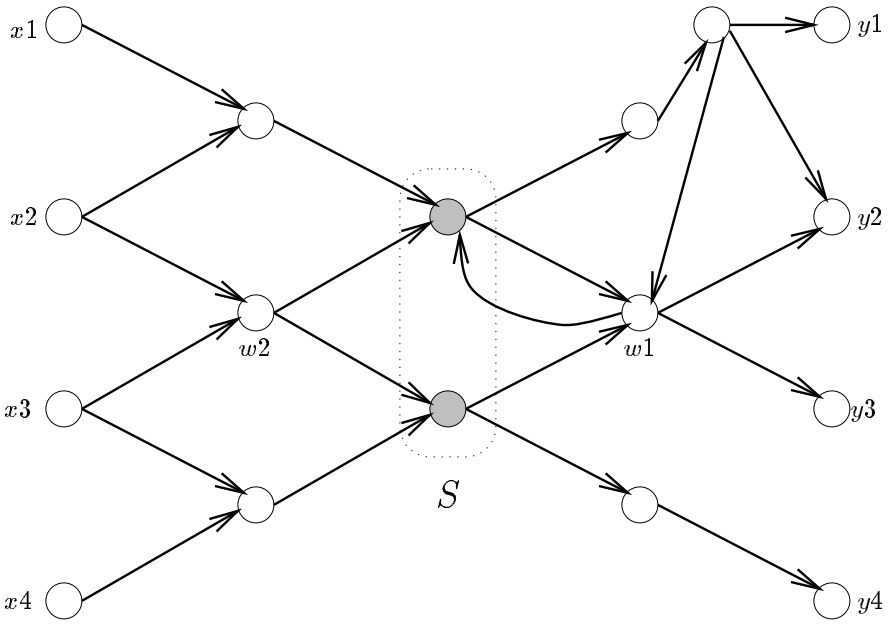


FIG. 4. *A digraph with a unique minimum $X \to Y$ separator S, where $X = \{x_1, x_2, x_3, x_4\}$ and $Y = \{y_1, y_2, y_3, y_4\}$. Vertex w_1 is in $T(S)$ since $p(\{w_1\} \cup X, Y) = 3 > |S|$, but vertex w_2 is not in $T(S)$ since $p(\{w_2\} \cup X, Y) = 2 = |S|$.*

path from a vertex in $\{v\} \cup X$ to a vertex in Y. Otherwise, if the vertex v is in T, then v is a witness to the fact that $S \neq S'$ and hence, by the uniqueness of S, $|S'| > |S|$. Suppose that there is an efficient method of computing $p(X, Y)$ for any specified pair of sets X and Y of the vertices, i.e., suppose that a fast "black box" subroutine for computing $p(X, Y)$ is available. (Such a method is described below.) Then the separator S may be found as follows. For each vertex $v \in V$, check whether $p(\{v\} \cup X, Y)$ is greater than $p(X, Y)$. Then construct the set T of vertices v that satisfy the inequality. The required separator S consists of the predecessors of T together with the Y-vertices not in T, i.e.,

$$S = \{s \in V - T \mid (s, v) \in E \text{ and } v \in T\} \cup (Y - T).$$

In general, a digraph may have many minimum $X \to Y$ separators. Fortunately, one of these separators satisfies the key property of the separator S and the vertex set $T(S)$ used above. This is proved in the next lemma. Though a full proof is given, the first part of the lemma is well known.

LEMMA 4.1. *Let $S^\star$ be an $X \to Y$ separator with cardinality $p(X, Y) = k$ such that $T(S^\star)$ is (inclusionwise) minimal over all $X \to Y$ separators with cardinality k. Then*

(i) $S^\star$ is unique, and

(ii) for each vertex v of G,

$$v \in T(S^\star) \quad \textit{iff} \quad p(\{v\} \cup X, Y) > k.$$

Proof. For a subset A of G's vertices, define $\Delta(A)$ to be the set of vertices

$$\{u \in V - A \mid (u,v) \in E \text{ and } v \in A\} \cup (Y - A),$$

i.e., $\Delta(A)$ consists of the predecessors of A as well as the Y-vertices not in A. Note that if A is the empty set, then $\Delta(A) = Y$. For every $A \subseteq V$, Y is a subset of $A \cup \Delta(A)$, and moreover, if A is a subset of $V - X$, then note that $\Delta(A)$ is an $X \rightarrow Y$ separator (since every path from a vertex in X to a vertex in $A \cup \Delta(A)$ must contain a vertex in $\Delta(A)$). Let $\delta(A)$ denote the cardinality of $\Delta(A)$. The proof hinges on the fact that the function $\delta : 2^V \longrightarrow \mathbb{Z}$ is submodular, i.e., for any two subsets A and B of G's vertices,

$$\delta(A \cap B) + \delta(A \cup B) \quad \leq \quad \delta(A) + \delta(B). \tag{1}$$

To see this, observe that if a vertex u contributes two to the left-hand side (i.e., $u \in \Delta(A \cap B) \cap \Delta(A \cup B)$), then either $u \in Y - (A \cup B)$ or $u \in V - (A \cup B \cup Y)$ and there is an edge uv, $v \in A \cap B$, so u contributes two to the right-hand side; otherwise, if u contributes one to the left-hand side, then u contributes at least one to the right-hand side.

To prove the first part of the lemma, by way of contradiction, assume that there are two minimum $X \rightarrow Y$ separators S_1 and S_2 such that $T_1 = T(S_1)$ is minimal and $T_2 = T(S_2)$ is minimal. Then note that $T_1 \cap T_2$ is both a proper subset of T_1 and a proper subset of T_2 (possibly, $T_1 \cap T_2$ is the empty set). Consider the vertex sets $T_1 \cap T_2$ and $T_1 \cup T_2$. Neither $T_1 \cap T_2$ nor $T_1 \cup T_2$ has any X-vertices since T_1 has no X-vertices and T_2 has no X-vertices. Hence, $\Delta(T_1 \cap T_2)$ is an $X \rightarrow Y$ separator, and $\Delta(T_1 \cup T_2)$ is an $X \rightarrow Y$ separator. Since the minimum cardinality of an $X \rightarrow Y$ separator is $p(X,Y) = k$, it is clear that $\delta(T_1 \cap T_2) \geq k$ and $\delta(T_1 \cup T_2) \geq k$. Now, using the submodularity of δ (equation (1)), it follows that $\delta(T_1 \cap T_2) = k$ and $\delta(T_1 \cup T_2) = k$. Let S' denote $\Delta(T_1 \cap T_2)$. Observe that $T(S')$ is a subset of $T_1 \cap T_2$, because for each vertex $v \notin (T_1 \cap T_2)$, every path from v to a vertex in Y contains a vertex of $\Delta(T_1 \cap T_2)$. This gives the desired contradiction and completes the first part of the lemma, since neither $T_1 = T(S_1)$ nor $T_2 = T(S_2)$ is minimal.

For the second part of the lemma, consider any vertex $v \in T(S^\star)$. The maximum number of vertex disjoint paths from $(\{v\} \cup X)$ to Y is either exactly $k = p(X,Y)$ or greater than k. Suppose that the number is k. Then, by Menger's theorem, there exists a separator S' of cardinality k whose removal from G leaves no path from $(\{v\} \cup X)$ to Y. Clearly, S' is also a minimum $X \rightarrow Y$ separator. Now, consider a minimum $X \rightarrow Y$ separator S such that $T(S)$ is a subset of $T(S')$ and $T(S)$ is (inclusionwise) minimal over all such separators; since S' exists, S must exist. By the first part of the lemma, the separators S and $S^\star$ are the same. This gives the desired contradiction, since $v \in T(S^\star) - T(S)$. We conclude that the maximum number of vertex disjoint paths from $(\{v\} \cup X)$ to Y is greater than k. □

Two results are needed to develop a fast, probabilistic method for computing $p(X,Y)$. The first result is attributed to Ingleton and Piff [IP 73]; for completeness, a proof that

follows [LLW 88, Theorem 3.1] is included in the appendix. Associate a variable $x(i,i)$ with each vertex i, and a variable $x(i,j)$ with each edge (i,j) (all variables are distinct). The *free adjacency matrix* $\widetilde{F} = \widetilde{F}(x(i,j))$ of G is an $n \times n$ matrix whose entries are given by

$$\widetilde{F}_{ij} = \begin{cases} x(i,i) & \text{if } i = j \\ x(i,j) & \text{if } i \neq j \text{ and } (i,j) \in E \\ 0 & \text{otherwise.} \end{cases}$$

THEOREM 4.2 (INGLETON AND PIFF). *Let G be a digraph, and let $\widetilde{F}$ be its free adjacency matrix. Then for any k-vertex set X and any k-vertex set Y,*

$$p(X,Y) = k \quad \textit{iff} \quad \det \widetilde{F}(\overline{Y},\overline{X}) \textit{ is not identically zero.}$$

We also need a matrix identity of Jacobi, see [BR 91, Lemma 9.2.10]:

FACT (JACOBI). *If a matrix F is nonsingular, then a square submatrix $F(\overline{Y},\overline{X})$ is nonsingular iff the complementary submatrix $F^{-1}(X,Y)$ is nonsingular. More precisely,*

$$\det(F^{-1}(X,Y)) = \det(F(\overline{Y},\overline{X}))/\det(F).$$

To apply the above theorem to the algorithm, the variables are substituted by random values. This is motivated by the Zippel-Schwartz lemma, Lemma 2.1.

THEOREM 4.3. *Let $G = (V,E)$ be a digraph, and let $F = \widetilde{F}(w(i,j))$ be obtained from G's free adjacency matrix by randomly and independently assigning each variable $x(i,j)$ a random number $w(i,j)$ from $\{1,\ldots,W\}$. Then with probability at least $1-(n/W)$, F is nonsingular. If F is nonsingular, then for every pair of sets X and Y of the vertices,*

$$p(X,Y) \geq \operatorname{rank}(F^{-1}(X,Y)),$$

and with probability at least $1-(n^2/W)$, $p(X,Y)$ equals $\operatorname{rank}(F^{-1}(X,Y))$.

Proof. View the determinant of the free adjacency matrix $\widetilde{F}$ as a polynomial of degree n in the variables $x(i,j)$, $1 \leq i,j \leq n$, and notice that it is not identically zero because the diagonal term $\Pi_{i=1}^{n} x(i,i)$ is nonzero and no two nonzero terms cancel out. Hence, by Lemma 2.1, F is nonsingular with probability at least $1-(n/W)$.

Consider a maximum-cardinality set of vertex disjoint paths from X to Y. Let A be the set of start vertices of these $X \rightarrow Y$ paths, and let B be the set of end vertices. Obviously, $A \subseteq X$, $B \subseteq Y$, and $|A| = |B| = p(X,Y) = p(A,B)$. Let $\widetilde{H} = \widetilde{H}(x(i,j))$ denote

$(\det \tilde{F})\tilde{F}^{-1}$, i.e., $\widetilde{H}$ is the $n \times n$ matrix whose (k,ℓ) entry is $(-1)^{k+\ell}$ times the (ℓ,k) minor of $\tilde{F}(x(i,j))$; every entry of $\widetilde{H}$ is a polynomial of degree $n-1$ in the variables $x(i,j)$. By Jacobi's identity and Theorem 4.2, $\det \widetilde{H}(A,B)$ is not identically zero, while for every integer $q > p(X,Y)$, the determinant of every $q \times q$ submatrix of $\widetilde{H}(X,Y)$ is identically zero. The second part of the theorem follows by observing that if F is nonsingular, then $F^{-1} = \tilde{F}(w(i,j))^{-1} = \widetilde{H}(w(i,j))/\det \tilde{F}(w(i,j))$; now apply Lemma 2.1 to $\det \widetilde{H}(A,B)$. □

The algorithm can now be sketched. Fix a number $W = n^{O(1)}$, and let q be a prime such that $W < q = n^{O(1)}$. All computations are over the field $\mathbb{Z}_q$ of integers modulo q. The matrix F is constructed, and with high probability it is nonsingular. Inverting F gives the matrix F^{-1}. If $r = \mathrm{rank}(F^{-1}(X,Y))$ equals $|Y|$, then by Theorem 4.3, $p(X,Y)$ equals $|Y|$, therefore Y is a minimum $X \to Y$ separator. Otherwise, consider the unique minimum $X \to Y$ separator $S^\star$ with $T(S^\star)$ minimal. The algorithm attempts to compute the vertex set $T(S^\star)$ by finding the set T of vertices v such that $\mathrm{rank}(F^{-1}(\{v\} \cup X, Y))$ is greater than r. With probability at least $1 - \Theta(n^3)/W$, T equals $T(S^\star)$. Hence, with high probability, the set $S = \Delta(T)$ (i.e., the set of the predecessors of T and the Y-vertices not in T) is the minimum $X \to Y$ separator $S^\star$.

To efficiently compute for each vertex v whether $\mathrm{rank}(F^{-1}(\{v\} \cup X, Y)) > r$, the algorithm needs to check that v's row vector $F^{-1}(\{v\}, Y)$ is not a linear combination of the row vectors of $F^{-1}(X,Y)$. As in Section 3, we "simultaneously" compute the ranks of all the matrices $F^{-1}(\{v\} \cup X, Y)$, $v \in V$, by computing a matrix N such that for any row vector w, $w \cdot N = 0$ iff w is a linear combination of the row vectors of $F^{-1}(X,Y)$. The matrix N is easily obtained by computing a basis for the null space of $F^{-1}(X,Y)$. Once N is computed, we simply find the product of the matrix $F^{-1}(V,Y)$ with N.

To check that the computed set S is indeed a minimum $X \to Y$ separator, observe that the cardinality of every $X \to Y$ separator is at least $p(X,Y) \geq r$. Consequently, if the removal of S from G leaves no path from $X - S$ to $Y - S$, and $|S| = r$, then $|S| = p(X,Y) = r$, and hence S is a minimum $X \to Y$ separator. Also, using Proposition 2.4, this algorithm may be applied to find a minimum vertex cover in a bipartite graph.

Consider the sequential complexity of the above algorithm. Inverting F takes $\tilde{O}(M(n))$ bit operations [AHU 74, Theorem 6.5]. Finding a basis for the null space of $F^{-1}(X,Y)$ takes $\tilde{O}(M(n))$ bit operations [IMH 82, pp. 53–54]. The remaining computations are easy to execute within this bound. Consider the randomized parallel complexity of the algorithm. Inverting F takes parallel time $O((\log n)^2)$ using $\tilde{O}(M(n))$ processors, [KP 91, Theorem 6], and these complexity bounds suffice for finding a basis for the null space of $F^{-1}(X,Y)$ [KP 91, p. 190] (both computations are randomized, and on a given matrix, the computed results may be incorrect with small probability). The remaining steps are easy to implement within these complexity bounds.

ALGORITHM 3. *Monte Carlo Minimum $X \to Y$ Separator*
INPUT: Graph $G = (V, E)$.
OUTPUT: With high probability, a minimum $X \to Y$ separator of G.

Order the vertices, and number them $1, 2, \ldots, n$.
Fix the number $W = n^{O(1)}$, and choose a prime q, $W < q = n^{O(1)}$.
Construct the matrix F by replacing each nonzero entry in the free adjacency matrix of G by an independent random number from $\{1, 2, \ldots, W\}$.

Invert F over the field $\mathbb{Z}_q$ to obtain the matrix F^{-1}.
(If F is singular, the algorithm stops and reports failure.)

Compute the rank r of the submatrix $F^{-1}(X, Y)$ over $\mathbb{Z}_q$.
If $r = |Y|$, then the required separator is $S = Y$. Stop.

Otherwise, compute a basis $\{N_1, \ldots, N_{|Y|-r}\}$ for the null space of the submatrix $F^{-1}(X, Y)$ over $\mathbb{Z}_q$ (each N_i is a vector of dimension $|Y|$).
Compute the matrix $Z = F^{-1}(V, Y) \cdot N$ over $\mathbb{Z}_q$,
where N is the matrix whose ith column is N_i.
Let Z_v denote the row of Z given by $F^{-1}(\{v\}, Y) \cdot N$.
Construct the set of vertices $T = \{v \mid Z_v \text{ is a nonzero vector }\}$.

With high probability, the required separator S consists of the predecessors of T together with the Y-vertices not in T, i.e.,

$$S = \Delta(T) = \{s \in V - T \mid (s, v) \in E \text{ and } v \in T\} \cup (Y - T).$$

Making the algorithm Las Vegas:
If $G - S$ has no path from $X - S$ to $Y - S$, and $|S| = r$,
then guarantee that S is a minimum $X \to Y$ separator, otherwise, report failure.

FIG. 5.

THEOREM 4.4. *Given a digraph G and a pair of sets X, Y of G's vertices, a minimum $X \to Y$ separator can be computed by a Las Vegas algorithm. The sequential running time is $\widetilde{O}(M(n))$. The parallel complexity is $O((\log n)^2)$ time using $\widetilde{O}(M(n))$ processors. The same complexity bounds apply for finding a minimum vertex cover of a bipartite graph.*

5. Conclusions. The most important problem left open is whether a maximum matching can be computed in deterministic or randomized time $O(n^{2.5-\epsilon})$, $\epsilon > 0$. The same problem

specialized to bipartite graphs, or equivalently (by Proposition 2.4), the problem of finding a maximum-cardinality set of vertex disjoint $X \to Y$ paths in a digraph in (randomized) time $O(n^{2.5-\epsilon})$, is also open. Another interesting open problem pertains to graphs with 0–1 weights on the edges: can the maximum weight of a perfect matching, but not necessarily the edge-set of the matching, be computed in (randomized) time $O(n^{2.5-\epsilon})$? Can the algorithm of Theorem 3.8 for finding the allowed edges be made Las Vegas without affecting the complexity bounds? The algorithm for finding a minimum bipartite vertex cover may be derived starting either from the Gallai-Edmonds theorem (Theorem 3.1) or from the theorem on the free adjacency matrix (Theorem 4.2). Do these two theorems have other connections?

6. Appendix: Proofs of Proposition 2.4 and Theorem 4.2.

Proof. (Proposition 2.4) First, we show how to transform instances and recover solutions of the two bipartite graph problems, using the corresponding digraph problems. Let $H = (V_1, V_2, E)$ be a bipartite graph, where V_1 and V_2 are the sets of the vertex bipartition. Construct a digraph $G = (V_1 \cup V_2, F)$ from H by orienting all edges from V_1 to V_2. Every matching of H corresponds to a set of vertex disjoint $V_1 \rightarrow V_2$ paths in G. Hence, a maximum matching of H can be found by computing a maximum cardinality set of vertex disjoint $V_1 \rightarrow V_2$ paths in G. Consider the bipartite graph minimum vertex cover problem. A subset of $V_1 \cup V_2$ is a vertex cover of H iff it is a $V_1 \rightarrow V_2$ separator of the digraph G. Therefore, a minimum vertex cover of H can be found by computing a minimum $V_1 \rightarrow V_2$ separator of G.

Next, consider the transformation of the two digraph problems to the corresponding bipartite graph problems, and the transformation of the solutions. Let $G = (V, F)$ be the given digraph, and let X and Y be specified subsets of V. Without loss of generality, assume that $X \cap Y = \emptyset$; the method here easily extends to the case when $X \cap Y \neq \emptyset$. Let $\overline{n}$ denote $|V(G) - X - Y|$. We construct a bipartite graph $H = (V_1, V_2, E)$ starting from G, X, Y. For each vertex $v \in V(G) - X - Y$, H has a pair of vertices v_1, v_2 with $v_1 \in V_1$ and $v_2 \in V_2$; also, H has the edge $v_1 v_2$; for each vertex $x \in X$, H has a vertex $x_1 \in V_1$; and for each vertex $y \in Y$, H has a vertex $y_2 \in V_2$. For every vertex v of G, let v_1 and v_2 denote the corresponding vertices in V_1 and V_2 (if they exist); let X_1 denote the set of vertices of H that corresponds to X. For each edge (v, w) of G, $v \notin Y$ and $w \notin X$, there is an edge $v_1 w_2$ in H.

A set of vertex disjoint $X \rightarrow Y$ paths of G of maximum cardinality (namely, $p(X, Y)$) gives a matching E' of H with $|E'| = p(X, Y) + \overline{n}$: start with $E' = \{v_1 v_2 \ : \ v \in V(G) - X - Y\}$, and then sequentially for each of the $X \rightarrow Y$ paths of G in the set mentioned above, augment E' using the corresponding alternating path of H. Moreover, we claim that a matching E' of H gives a set of at least $|E'| - \overline{n}$ vertex disjoint $X \rightarrow Y$ paths of G: starting from the vertices in X_1 in H, use the matching E' and the edges $v_1 v_2$, $v \in V(G) - X - Y$, to construct a set of vertex disjoint paths in G; each of these paths ends either at a vertex in Y or at a vertex $v \notin Y$ such that in H the corresponding vertex v_1 is exposed; hence, at most $|V_1| - |E'| = |X_1| + \overline{n} - |E'|$ of these paths in G have their end vertices in $V - Y$; our claim follows since the number of these paths in G is $|X_1|$. Consequently, $\nu(H) = p(X, Y) + \overline{n}$, and every maximum matching of H yields a set of $p(X, Y)$ vertex disjoint $X \rightarrow Y$ paths of G.

Now, consider the problem of finding a minimum $X \rightarrow Y$ separator S of G. We find a minimum vertex cover C of H, and then construct S as follows: S contains a vertex $x \in X$ iff C contains the vertex x_1; S contains a vertex $y \in Y$ iff C contains the vertex y_2; and S contains a vertex $v \in V(G) - X - Y$ iff C contains both the vertices v_1 and v_2. Since $|C| = \nu(H) = p(X, Y) + \overline{n}$, and since either $v_1 \in C$ or $v_2 \in C$ for each vertex $v \in V(G) - X - Y$, we see that $|S| = p(X, Y)$. We claim that S is an $X \rightarrow Y$ separator of G. By way of contradiction, suppose that there is a path P in $G - S$ with start vertex $x \in X$ and end vertex $y \in Y$. Focus on the subgraph $H(P)$ of H formed by the edges that correspond

to the edges of P, together with the edges v_1v_2 of H that correspond to the internal vertices v of P (i.e., $v \in V(P) - \{x, y\}$). Since C is a vertex cover of H, every edge of $H(P)$ must be incident with some vertex of C. Consequently, either $x \in C$ or $y \in C$ or there is an internal vertex v of P such that $v_1 \in C$ and $v_2 \in C$. We have the desired contradiction since S intersects P. □

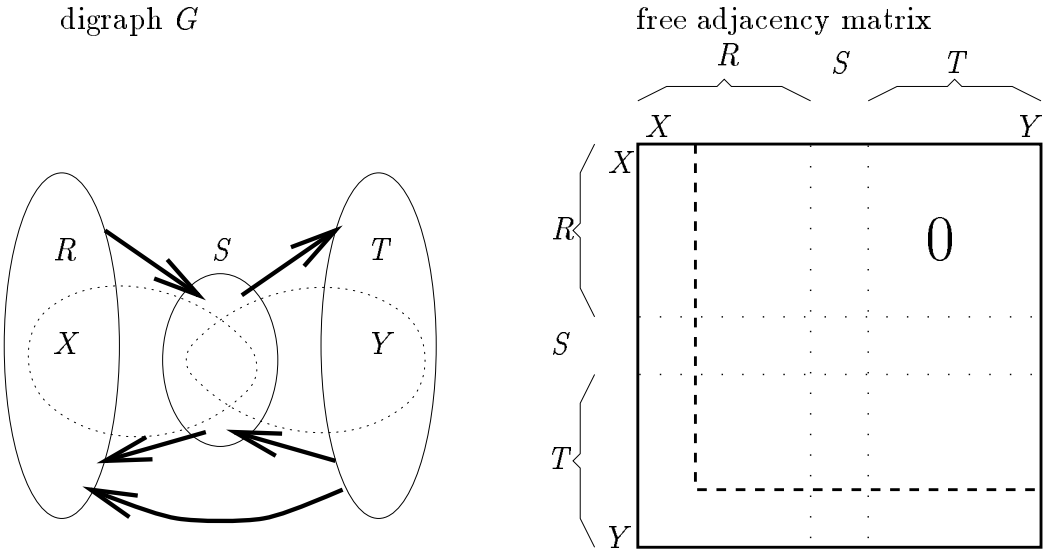


FIG. 6. *An illustration of the proof of Theorem 4.2. The submatrix $\widetilde{F}(\overline{Y}, \overline{X})$ is indicated by dashed lines.*

Proof. (Theorem 4.2) First, consider the case when $p(X, Y)$ is less than $k = |Y|$. Let S be an $X \rightarrow Y$ separator of cardinality $p(X, Y)$, and let T denote the set of vertices that have paths to $Y - S$ in $G - S$. Let R denote $V - (S \cup T)$. Note that X is a subset of $R \cup S$ and Y is a subset of $S \cup T$. Since G has no edges of the form (r, t), $r \in R$, $t \in T$, each entry of the submatrix $\widetilde{F}(R, T)$ is zero. (See Figure 6.) A line denotes either a row or a column of a matrix. Focus on the number of lines needed to cover all the nonzero entries of $\widetilde{F}(\overline{Y}, \overline{X})$, and consider the columns corresponding to the vertex set $(R \cup S) - X$, and the rows corresponding to the vertex set $(S \cup T) - Y$. Each entry of $\widetilde{F}(\overline{Y}, \overline{X})$ that is not covered by these lines is in an R-row and a T-column, hence the entry is zero. Thus the number of lines needed is at most

$$((n - |T|) - k) + ((n - |R|) - k) \leq (n - k - 1),$$

since $n = |R| + |S| + |T| \leq (k-1) + |R| + |T|$. Now, use the fact that for a bipartite graph, the cardinality of every matching is less than or equal to the cardinality of every vertex cover. It follows that there are at most $(n - k - 1)$ nonzero entries in $\widetilde{F}(\overline{Y}, \overline{X})$ with no two of these entries on a line. Hence, $\det \widetilde{F}(\overline{Y}, \overline{X})$ is identically zero since each term in the standard expansion of the determinant is zero.

Next, suppose that $p(X,Y)$ equals $k = |Y|$. Let $P_1, \ldots, P_k$ be a maximum set of vertex disjoint $X \rightarrow Y$ paths. Denote the start vertex of path P_i $(1 \leq i \leq k)$ by x_i, and denote the end vertex by y_i. Let A denote the set of vertices not in these paths. For each vertex $v \in A$, define $\sigma(v)$ to be v, and for each vertex $v \in (V - A - Y)$, define $\sigma(v)$ to be the successor of v in the path P_i containing v. Note that σ is well defined even if trivial paths P_i (having $x_i = y_i$) are present. For each $v \in \overline{Y}$, note that $\sigma(v)$ belongs to $\overline{X}$, and that $\widetilde{F}_{v,\sigma(v)}$ is a nonzero entry of the submatrix $\widetilde{F}(\overline{Y}, \overline{X})$. Moreover, observe that σ is one-one, i.e., $\sigma(v) = \sigma(w)$ iff $v = w$, and therefore no two entries from the set $\{\widetilde{F}_{v,\sigma(v)} \mid v \in \overline{Y}\}$ are on a line. It follows that the product $(\pm 1) \cdot \Pi_{v \in \overline{Y}} \widetilde{F}_{v,\sigma(v)}$ is one of the terms in the standard expansion of $\det \widetilde{F}(\overline{Y}, \overline{X})$. Clearly, the product is nonzero. Hence, the determinant evaluates to ± 1 when the value one is assigned to each entry $\widetilde{F}_{v,\sigma(v)}$, $v \in \overline{Y}$, and the value zero is assigned to the remaining entries of $\widetilde{F}(\overline{Y}, \overline{X})$. Therefore, the determinant is not identically zero. □

Acknowledgments. Section 4 has benefited from discussions with Éva Tardos. The Monte Carlo algorithm for the canonical partition in Section 3.4 was discovered jointly with K. Padayachee and is included with his consent. J. A. La Poutré communicated an almost linear-time algorithm for verifying the canonical partition. The careful comments by the referees are appreciated.

REFERENCES

[AHU 74] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[B 90] N. Blum, *A new approach to maximum matching in general graphs*, Proc. 17th ICALP, Lecture Notes in Computer Science **443**, Springer-Verlag, Berlin, 1990, 586–597.

[BR 91] R. A. Brualdi and H. J. Ryser, *Combinatorial Matrix Theory*, Cambridge University Press, New York, 1991.

[C 93] J. Cheriyan, *Random weighted Laplacians, Lovász minimum digraphs and finding minimum separators*, extended abstract in Proc. 4th ACM-SIAM Symposium on Discrete Algorithms (1993), 31–40.

[C 94] J. Cheriyan, *A Las Vegas $O(n^{2.38})$ algorithm for the cardinality of a maximum matching*, extended abstract in *Proc. 5th ACM-SIAM Symposium on Discrete Algorithms* (1994), 442-451.

[CR 94] J. Cheriyan and J. H. Reif, *Directed s-t numberings, rubber bands, and testing digraph k-vertex connectivity*, Combinatorica **14** (1994), 435–451.

[CW 90] D. Coppersmith and S. Winograd, *Matrix multiplication via arithmetic progressions*, J. Symbolic Comp., **9** (1990), 251–280.

[E 91] W. Eberly, *Efficient parallel independent subsets and matrix factorizations*, Proc. 3rd IEEE Symposium on Parallel and Distributed Processing, (1991), 204–211.

[E 65] J. Edmonds, *Paths, trees and flowers*, Canad. J. Math. **17**, (1965), 449–467.

[GT 91] H. N. Gabow and R. E. Tarjan, *Faster scaling algorithms for general graph-matching problems*, J. Association for Computing Machinery **38** (1991), 815–853.

[GP 88] Z. Galil and V. Pan, *Improved processor bounds for combinatorial problems in RNC*, Combinatorica **8** (1988), 189–200.

[Ga 64] T. Gallai, *Maximale Systeme unabhängiger Kanten*, Magyar Tud. Akad. Mat. Kutató Int. Közl. **9** (1964), 401–413.

[HK 95] M. Rauch Henzinger and V. King, *Randomized dynamic algorithms with polylogarithmic time per operation*, Proc. 27th Annual ACM Symposium on Theory of Computing (1995), 519–527.

[IMH 82] O. H. Ibarra, S. Moran and R. Hui, *A generalization of the fast LUP matrix decomposition algorithm and applications*, J. Algorithms **3** (1982), 45–56.

[IP 73] A. W. Ingleton and M. J. Piff, *Gammoids and transversal matroids*, J. Combinatorial Theory (B) **15** (1973), 51–68.

[KP 91] E. Kaltofen and V. Pan, *Processor efficient parallel solution of linear systems over an abstract field*, Proc. 3rd Annual ACM Symposium on Parallel Algorithms and Architectures (1991), 180–191.

[Kf 86] H. J. Karloff, *A Las Vegas RNC algorithm for maximum matching*, Combinatorica **6** (1986), 387–391.

[KUW 86] R. M. Karp, E. Upfal and A. Wigderson, *Constructing a perfect matching is in Random NC*, Combinatorica **6** (1986), 35–48.

[Ko 91] D. Kozen, *The Design and Analysis of Algorithms*, Springer-Verlag, Berlin, 1991.

[L 95] J. A. La Poutré, Personal communication, (January 1995).

[LW 94] J. A. La Poutré and J. Westbrook, *Dynamic two-connectivity with backtracking*, *Proc. 5th ACM-SIAM Symposium on Discrete Algorithms* (1994), 204–212.

[LLW 88] N. Linial, L. Lovász and A. Wigderson, *Rubber bands, convex embeddings and graph connectivity*, Combinatorica **8** (1988), 91–102.

[Lo 79] L. Lovász, *On determinants, matchings and random algorithms*, In: Fundamentals of Computation Theory, L. Budach, Ed., Akademie-Verlag, Berlin, 1979, 565–574.

[Lo] L. Lovász, *Combinatorial Problems and Exercises* (second ed.), North-Holland, Amsterdam, 1993.

[LP 86] L. Lovász and M. Plummer, *Matching Theory*, Akadémiai Kiadó, Budapest, Hungary, 1986.

[MV 80] S. Micali and V. V. Vazirani, *An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs*, Proc. 21st Annual IEEE Symposium on Foundations of Computer Science (1980), 17–27.

[MVV 87] K. Mulmuley, U. V. Vazirani and V. V. Vazirani, *Matching is as easy as matrix inversion*, Combinatorica **7** (1987), 105–113.

[RV 89] M. O. Rabin and V. V. Vazirani, *Maximum matchings in general graphs through randomization*, J. Algorithms **10** (1989), 557-567.

[Sc 80] J. T. Schwartz, *Fast probabilistic algorithms for verification of polynomial identities*, J. ACM **27** (1980), 701–717.

[ST 83] D. D. Sleator and R. E. Tarjan, *A data structure for dynamic trees*, J. Comput. Syst. Sci. **26** (1983), 362–391.

[T 47] W. T. Tutte, *The factorization of linear graphs*, J. London Math. Soc. **22** (1947), 107–111.

[V 94] V. V. Vazirani, *A theory of alternating paths and blossoms for proving correctness of the $O(\sqrt{V}E)$ general graph matching algorithm*, Combinatorica **14** (1994), 71–109.

[W 91] J. Wein, *Las Vegas RNC algorithms for unary weighted perfect matching and T-join problems*, Information Processing Letters **40** (1991), 161–167.

[Z 79] R. E. Zippel, *Probabilistic algorithms for sparse polynomials.* In Proc. EUROSAM 79, Ng, Ed., Lecture Notes in Computer Science **72**, Springer-Verlag, Berlin, 1979, 216–226.