Algorithmica (1996) 16: 392–401

Algorithmica

A Mildly Exponential Approximation Algorithm for the Permanent

M. Jerrum[1] and U. Vazirani[2]

Abstract. A new approximation algorithm for the permanent of an $n \times n$ 0,1-matrix is presented. The algorithm is shown to have worst-case time complexity $\exp(O(n^{1/2}\log^2 n))$. Asymptotically, this represents a considerable improvement over the best existing algorithm, which has worst-case time complexity $\exp(\Theta(n))$.

Key Words. Approximation algorithms, Combinatorial enumeration, Perfect matchings, Permanent, Rapidly mixing Markov chains.

1. Summary. The *permanent* of an $n \times n$ matrix $A = (a_{ij}\colon\ 0 \le i, j \le n-1)$ is defined by

$$\operatorname{per}(A) = \sum_{\pi} \prod_{i=0}^{n-1} a_{i,\pi(i)}\,,$$

where the sum is over all permutations π of $[n] = \{0, \ldots, n-1\}$. In this paper A is always a 0,1-matrix, in which case the permanent of A has a simple combinatorial interpretation: namely, $\operatorname{per}(A)$ is equal to the number of perfect matchings (1-factors) in the bipartite graph $G = (V, V', E)$, where $V = V' = [n]$, and $(i, j') \in E$ iff $a_{ij'} = 1$. (Primes are used consistently to distinguish the blocks of the bipartition of G.) The permanent has been the object of extensive study, since first appearing in 1812 in the work of Cauchy and Binet. Minc [10] has provided an excellent survey of the results of these investigations.

Despite considerable effort, and in contrast with the syntactically very similar determinant, no efficient procedure for computing the permanent is known. Valiant [13] provided convincing evidence for the inherent intractability of the permanent, by demonstrating that it is complete for the class #P of enumeration problems, and thus as hard as counting the number of satisfying assignments to a CNF formula, or the number of accepting computations of a polynomial-time-bounded nondeterministic Turing machine. In light of this result, we should not expect to find a polynomial-time algorithm for the permanent of a 0,1-matrix. Indeed, the fastest known (exact) algorithms for the permanent have time complexity $\Theta(n2^n)$; see, for example, the method of Ryser [10, p. 122].

[1] Department of Computer Science, University of Edinburgh, The King's Buildings, Edinburgh EH9 3JZ, Scotland. mrj@dcs.ed.ac.uk. Supported by SERC Grant GR/F 90363; work done in part while visiting DIMACS (Center for Discrete Mathematics and Computer Science).

[2] Department of Computer Science, University of California, Berkeley, CA 94720, USA. vazirani@cs.berkeley.edu. Supported by an NSF PYI grant, with matching equipment grant from the AT&T Foundation; work done in part while visiting DIMACS.

Received March, 1992; revised December 16, 1994. Communicated by M. Luby.

Recently, Karmarkar *et al*. [7] showed that a lower time complexity can be achieved, provided we are content to settle for an algorithm that provides only an *approximation* to per(A) within a specified relative error. Their algorithm is an example of a "randomized approximation scheme" [8]. Let f be a function from input strings to the natural numbers. A *randomized approximation scheme* for f is a probabilistic algorithm that takes as input a string x and a real number $0 < \varepsilon < 1$, and produces as output a number Y (a random variable) such that Y approximates $f(x)$ within ratio $1 + \varepsilon$ with high probability.[3] For definiteness we take the phrase "with high probability" to mean with probability at least $\frac{3}{4}$. The success probability may be boosted to $1 - \delta$ for any desired $\delta > 0$ by running the algorithm $O(\log \delta^{-1})$ times and taking the median of the results [6, Lemma 6.1].

The randomized approximation scheme for per(A) proposed by Karmarkar *et al*. has time complexity $\varepsilon^{-2}\,\mathrm{poly}(n)\,2^{n/2}$. It is a classical Monte Carlo algorithm which performs a sequence of trials using an unbiased estimator for the permanent; this estimator is closely modeled on one given earlier by Godsil and Gutman [3]. The worst-case time complexity of this algorithm is the lowest of any existing approximation scheme for the permanent; nevertheless it shares with the exact algorithm of Ryser the general form $\exp(\Theta(n))$.

In this paper we use different techniques to design a randomized approximation scheme for per(A) that has time complexity $\varepsilon^{-2}\exp(O(n^{1/2}\log^2 n))$. The new algorithm is easily sketched at an informal level, though the details need careful working out. The approach is one of "divide-and-conquer," and can best be described using the graph-theoretic formulation introduced earlier. Let G be a bipartite graph, and suppose that we wish to estimate the number $N(G)$ of perfect matchings in G. The progress of the algorithm is controlled by the expansion properties of G. If G has a sufficiently large expansion ratio, then an existing approximation scheme of Jerrum and Sinclair [5] is used to estimate $N(G)$ directly with acceptable speed. Otherwise a set of vertices $A \subset V$ is found that has the property that A is adjacent to rather few vertices in V'. The set A forms a "constriction" in G, which allows an economical decomposition of G into subproblems that may be solved recursively. The full description and analysis of this approximation scheme form the main body of this note.

Given our current (lack of) knowledge, it is entirely possible that there exists a randomized approximation scheme for per(A) that has time complexity $\mathrm{poly}(n, \varepsilon^{-1})$. (Such an approximation scheme is said to be *fully polynomial*.) Thus, although the approximation scheme presented here is asymptotically much faster than any previously proposed, it may still be very far from optimal. The question of whether there exists a fully polynomial (randomized) approximation scheme for per(A) presents an intriguing challenge. Optimists may take heart from the fact that such algorithms are known for some restricted classes of 0,1-matrices [2], [4], [5]; these classes have the property that the proportion of $n \times n$ matrices lying within the class tends to 1 as n tends to infinity.

2. The Algorithm and Its Analysis. Let G be a bipartite graph on vertex set $V + V'$ and edge set E, where $V = V' = [n] = \{0, 1, \ldots, n-1\}$. We use primes to distinguish

[3] For nonnegative real numbers $a, \hat{a}, \varepsilon$, we say that $\hat{a}$ *approximates* a *within ratio* $1 + \varepsilon$ if $a(1+\varepsilon)^{-1} \le \hat{a} \le a(1+\varepsilon)$.

vertices in V' from those in V; thus each vertex v in V has a counterpart v' in V', and each subset U of V has a counterpart $U' = \{u' \in V' \colon u \in U\}$ in V'. Denote by G^* the reversal of G, i.e., the graph with vertex set $V + V'$ and edge set $E^* = \{(u, v') \in V \times V' \colon (v, u') \in E\}$. For each subset $A \subseteq V$, let $\Gamma_G(A)$ denote the set of all vertices in V' that are adjacent to some vertex in A. Suppose α is in the range $0 < \alpha \leq 1$. We say that G is an *α-expander* if $|\Gamma_G(A)| \geq (1+\alpha)|A|$ for every $A \subseteq V$ with $|A| \leq \frac{1}{2}n$. Let $N(G)$ denote the number of perfect matchings in G and $N^-(G)$ the number of *near-perfect* matchings in G, i.e., matchings that leave precisely two vertices uncovered.

Jerrum and Sinclair showed that the total number $N(G)$ of perfect matchings in G may be efficiently estimated, provided the ratio $N^-(G)/N(G)$ of near-perfect to perfect matchings is small. Proposition 1—which simply rephrases Corollary 3.7 of [5] in the terminology and notation of the current note—makes this claim precise.

PROPOSITION 1. *Let $q(n)$ be any fixed polynomial. There is a randomized approximation scheme that meets the following specification*:

(i) *The input is a pair (G, ε), where G is a bipartite graph on $n+n$ vertices satisfying $N^-(G)/N(G) \leq q(n)$, and $0 < \varepsilon < 1$ is a parameter controlling the accuracy of the result.*
(ii) *The output is a random variable that with probability at least $\frac{3}{4}$ approximates $N(G)$ within ratio $1+\varepsilon$.*
(iii) *The execution time is polynomial in n and ε^{-1}.*

The approximation scheme referred to in Proposition 1 works by simulating a Markov chain whose states are matchings in G. The notion of basing an approximation scheme for the permanent on such a Markov chain was first proposed by Broder [1], but two years were to elapse before a correct analysis was completed [11], [5], [12]. The key part to the analysis is showing that the Markov chain in question is "rapidly mixing," i.e., reaches near-equilibrium in polynomially many steps; it is in demonstrating rapid mixing that the upper bound on $N^-(G)/N(G)$ is crucial. By simulating the Markov chain from an arbitrary initial state, it is possible to sample matchings from G according to a known distribution, and from there it is a fairly short step to estimating the total number of perfect matchings.

It is possible to relate the crucial ratio $N^-(G)/N(G)$ to the expansion properties of G.

LEMMA 2. *If G and G^* are α-expanders, then $N^-(G)/N(G) \leq \exp(O(\alpha^{-1}\log^2 n))$.*

We postpone the proof of Lemma 2 to allow an investigation of its consequences.

COROLLARY 3. *There is a randomized approximation scheme* MARKOVSIM *that meets the following specification*:

(i) *The input to* MARKOVSIM *is a triple (G, α, ε), where G is a bipartite graph on $n+n$ vertices, α is a number which is to be interpreted as a guarantee that G and G^* are both α-expanders, and $0 < \varepsilon < 1$ controls the accuracy of the result.*

(ii) *The output is a random variable that with probability at least* $\frac{3}{4}$ *approximates* $N(G)$ *within ratio* $1+\varepsilon$.
(iii) *The execution time of* MARKOVSIM *is* $\varepsilon^{-2}\exp(O(\alpha^{-1}\log^2 n))$.

PROOF. Lemma 2 provides an upper bound ρ on the ratio $N^-(G)/N(G)$ which is of the form $\rho = \exp(O(\alpha^{-1}\log^2 n))$. Now ρ is not in general bounded by any fixed polynomial in n. However, we can still use the approximation scheme promised by Proposition 1 (with $q(n) = 2n$) by padding the input. Let $m = \lceil \rho \rceil$ and let G' be the graph obtained by taking the disjoint union of G with m copies of K_2, the complete graph on two vertices. Then

$$\frac{N^-(G')}{N(G')} = \frac{N^-(G) + m\,N(G)}{N(G)} \leq 2m.$$

However, since $2m$ is less than the number of vertices in G', we are in the situation of Proposition 1 with $q(n) = 2n$. This gives us an approximation scheme MARKOVSIM whose overall time complexity is bounded by $\varepsilon^{-2}\log(\varepsilon^{-1})\,\mathrm{poly}(n, \rho)$. (The chosen name reflects the algorithmic technique at the heart of the approximation scheme.) Observe that there is no point in taking ε^{-1} to be greater than $3n!$, for this already guarantees an *exact* evaluation of $N(G)$; thus, without loss of generality, we may assume $\log \varepsilon^{-1} = O(n \log n)$. Further observe that $\rho \geq N^-(G)/N(G) \geq n$. These two observations allow us to simplify the expression of the time complexity of MARKOVSIM to $\varepsilon^{-2}\,\mathrm{poly}(\rho)$.

The above padding argument is not particularly natural, but is the easiest route to establishing Proposition 1 from the existing published results. A more natural and direct approach would be to reappraise the approximation scheme presented in Figure 2 of [5], and demonstrate that it runs in time polynomial in ε^{-1} and ρ. Only minor changes to the analysis provided in [5] are needed to achieve this. □

PROOF OF LEMMA 2. Let M be any near-perfect matching in G. We demonstrate that M can be transformed to a perfect matching by augmentation along a short alternating path. It follows easily that the number of near-perfect matchings cannot exceed the number of perfect matchings by a large factor.

Define an *alternating path* (for M in G) to be a path that is composed of edges that are alternately elements of M and $E - M$. (The first and last edges of the path may or may not be elements of M.) Let $s \in V$ and $t' \in V'$ be the vertices left uncovered by M. We shall show that s is joined to t' via an alternating path of length at most $4k+1$, where k is the smallest integer such that $(1+\alpha)^k > \frac{1}{2}n$; observe that $k = O(\alpha^{-1}\log n)$. Let A_i, for $0 \leq i \leq k$, be the set of vertices in V that are reachable from s via an alternating path of length at most $2i$. The expansion property of G entails $|A_{i+1}| \geq (1+\alpha)|A_i|$ provided $|A_i| \leq \frac{1}{2}n$. Thus $|A_k| \geq \min\{(1+\alpha)^k, \frac{1}{2}n\} = \frac{1}{2}n$. Similarly, let B'_i, for $0 \leq i \leq k$, be the set of vertices in V' that are reachable from t' via an alternating path of length at most $2i$. The expansion property of G^* entails $|B'_k| \geq \frac{1}{2}n$. Now, by the expansion property, there must be an edge joining A_k to B'_k, and hence an alternating path from s to t' of length at most $4k+1$. (If the edge e joining A_k to B'_k happens to be an element of M, we apparently do not obtain an alternating path; note, however, that the edge e must then occur three times in succession, and can be collapsed to a single occurrence.)

We have demonstrated that every near-perfect matching M may be associated with a perfect matching $\bar{M}$ by augmentation along an alternating path of length $O(\alpha^{-1} \log n)$. This process assigns at most $\exp(O(\alpha^{-1} \log^2 n))$ near-perfect matchings to any given perfect matching. (To recover M from $\bar{M}$ we follow an alternating path of length at most $4k+1$, the selection of which involves a sequence of at most $2k+1$ choices from among at most n possibilities.) Thus $N^-(G)/N(G) = \exp(O(\alpha^{-1} \log^2 n))$, as claimed. □

Suppose the bipartite graph G contains a perfect matching, say M. Reorder the vertices in V' so that M matches each vertex v in V with its corresponding vertex v' in V'. It is clear that $|\Gamma_G(A)| \geq |A|$ for every set $A \subseteq V$; we call A *tight* [9] if equality holds, i.e., if $\Gamma_G(A) = A'$. Note that the collection of all tight sets in V is closed under union and intersection; in particular for each $v \in V$ there is a unique *smallest* tight set containing v, which we denote by $\Delta_G(v)$. It is not hard to check that $\Delta_G(v)$ is the set of all vertices in V that can be reached from v by an even-length alternating path in G which starts with an edge not in M; thus the sets $\Delta_G(v)$ are easy to compute in polynomial time. The procedure TESTEXPANSION presented in Figure 1 uses the idea of tight sets to compute (approximately) the expansion ratio of the input graph G, i.e., the largest α such that G is an α-expander. If the expansion ratio is not too small, Corollary 3 can be applied to estimate $N(G)$ directly. Otherwise, TESTEXPANSION returns a "constriction" in G (a set $A \subset V$ such that $|\Gamma_G(A)|/|A|$ is close to 1) which permits an efficient decomposition into subproblems which can then be solved recursively. The specification of TESTEXPANSION

```
    procedure TESTEXPANSION(G, α);
    begin
      Let M be any perfect matching in G;
      Reorder the vertices in V′ so that
        M matches each vertex v in V with the corresponding vertex v′ in V′;
      for each subset X ⊂ V with |X| < ½αn do begin
        Let Ĝ be the graph obtained from G
          by removing vertices X ∪ X′, and all incident edges;
(1)     if there exists v ∈ V with ¼n < |Δ_Ĝ(v)| ≤ ½n then A := Δ_Ĝ(v)
        else begin
          Let v_0, v_1, ..., v_{n−|X|−1} be an enumeration of vertices v ∈ V − X
            satisfying |Δ_Ĝ(v_0)| ≤ |Δ_Ĝ(v_1)| ≤ ··· ≤ |Δ_Ĝ(v_{n−|X|−1})|;
          i := 0;   A := ∅;
(2)       while |A ∪ Δ_Ĝ(v_i)| ≤ ½n do begin
            A := A ∪ Δ_Ĝ(v_i);   i := i + 1
          end
        end;
(3)     if |X|/|A| < 2α then output A and halt
      end;
(4)   output "G is an α-expander"
    end
```

Fig. 1. Algorithm for testing the expansion factor of a graph.

is made explicit in Lemma 4. The idea underlying the procedure is as follows: if the expansion ratio of G is small, then there must exist a relatively small set $X \subset V$ such that the graph obtained from G by removing X and X' contains a relatively large tight set. The procedure exhaustively searches for such a set X.

LEMMA 4. *On input* (G, α), *where* G *is a bipartite graph and* $\alpha > 0$, *the algorithm* TESTEXPANSION *presented in Figure* 1 either

(i) *correctly identifies* G *as an* α*-expander*, or
(ii) *produces a set* $A \subseteq V$ *such that* $|A| \leq \frac{1}{2}n$ *and* $|\Gamma_G(A)| < (1 + 2\alpha)|A|$.

The execution time of TESTEXPANSION *is* $\exp(O(\alpha n \log n))$.

PROOF. It is clear that the algorithm terminates, and either outputs a set $A \subseteq V$, or an assertion that G is an α-expander. In the first instance we must assure ourselves that the set A satisfies the conditions $|A| \leq \frac{1}{2}n$ and $|\Gamma_G(A)| < (1 + 2\alpha)|A|$, and in the second that the assertion is correct.

Suppose first that line (3) is reached and the condition $|X|/|A| < 2\alpha$ is true. The set A is a union of tight sets, and hence is itself tight (with respect to the graph $\hat{G}$); equivalently, $\Gamma_{\hat{G}}(A) = A'$. Thus $|\Gamma_G(A)| \leq |A' \cup X'| = |A \cup X| < (1 + 2\alpha)|A|$, as required.

Now suppose that the for-loop runs to completion, and line (4) is reached. We must demonstrate that the claim that G is an α-expander is correct. Suppose to the contrary that there exists a set $B \subset V$ such that $|B| \leq \frac{1}{2}n$ and $|\Gamma_G(B)| < (1 + \alpha)|B|$. Note that $\Gamma_G(B)$ can be written as a disjoint union $B' + X'$ with $|X'| < \alpha|B|$. Since $|X| < \frac{1}{2}\alpha n$, the for-loop will eventually consider the set X; observe that B is tight with respect to the graph $\hat{G}$ obtained by removing vertices X, X', and all incident edges. We distinguish two cases and obtain a contradiction in both.

Case 1. There exists a vertex $v \in V$ such that $\frac{1}{4}n < |\Delta_{\hat{G}}(v)| \leq \frac{1}{2}n$. This situation is caught in line (1). Since $|A| > \frac{1}{4}n$ and $|X| < \frac{1}{2}\alpha n$, the for-loop terminates prematurely at line (3), contrary to our assumption.

Case 2. There does not exist a vertex $v \in V$ such that $\frac{1}{4}n < |\Delta_{\hat{G}}(v)| \leq \frac{1}{2}n$. After execution of the while-loop in line (2) we are left with a set $A \subset V$ satisfying $|A| \leq \frac{1}{2}n$ and $|A \cup \Delta_{\hat{G}}(v_i)| > \frac{1}{2}n$. Now either $|\Delta_{\hat{G}}(v_i)| \leq \frac{1}{4}n$ or $|\Delta_{\hat{G}}(v_i)| > \frac{1}{2}n$. If the first inequality holds, $|A| > \frac{1}{4}n$ and we obtain a contradiction as in Case 1. If the second inequality holds, then no v_j with $j \geq i$ can be a member of B (since B is tight with respect to $\hat{G}$) and hence $A \supseteq B$. Thus $|X|/|A| \leq |X|/|B| < \alpha$ and again the for-loop terminates prematurely at line (3). □

We now have all the ingredients for the proposed approximation scheme, which is presented in Figure 2 as function COUNTPM. The analysis of COUNTPM is the subject of Theorem 5.

THEOREM 5. *With appropriate choices for the parameters* t *and* $\hat{\varepsilon}$, *the algorithm* COUNTPM *of Figure* 2 *is a randomised approximation scheme for the number of perfect*

```
      function CountPM(G);
      [[Returns an approximation to the number of perfect matchings in G.]]
      begin
        if n = 1 then return 0 or 1 as appropriate;
        if G has no perfect matchings then return 0;
        α := n^{-1/2};
(1)     Execute TestExpansion(G, α) and TestExpansion(G*, α);
(2)     if G and G* are both confirmed as α-expanders then begin
(3)       Call MarkovSim 2t + 1 times with parameters G, α, ε̂;
          [[The parameters t and ε̂ are appropriately initialized
            before entry to CountPM at the top level.]]
(4)       return the median of the 2t + 1 trials
        end else begin
          Without loss of generality, let A ⊂ V satisfy |A| ≤ ½n and
            |Γ_G(A)| < (1 + 2α)|A|;
          sum := 0;
(5)       for each subset B' ⊆ Γ_G(A) with |B'| = |A| do begin
            Let G_0 be the subgraph of G induced by A and B';
            Let G_1 be the subgraph of G induced by V − A and V' − B';
(6)         sum := sum + CountPM(G_0) × CountPM(G_1)
          end;
          return sum
        end
      end
```

Fig. 2. Recursive algorithm for counting perfect matchings.

matchings in G. The execution time of CountPM *is* $\varepsilon^{-2}\exp(O(n^{1/2}\log^2 n))$.

Proof. We first deal with the correctness of CountPM, deciding on the parameters t and $\hat{\varepsilon}$ as we do so. First imagine an idealized situation in which the procedure MarkovSim *always* returns the *exact* number of matchings in G. Then it is not too difficult to see that CountPM also produces the correct result. The crucial step to check is the division into subproblems effected by the for-loop in line (5). Observe that any perfect matching M in G matches $A \subset V$ with some subset $B' \subseteq \Gamma_G(A)$ of size $|A|$, and that the set of all matchings in G can be partitioned according to the set B' so defined. The for-loop ranges over the blocks of this partition, accumulating the number of matchings in each block as it goes. To see that the body of the loop is correct, it is merely necessary to observe that each perfect matching in the block defined by B' can be decomposed into a disjoint union of a perfect matching in G_0 and a perfect matching in G_1.

Next set $\hat{\varepsilon} = \varepsilon/2n_0$, where n_0 is the value of n at the top level of recursion, and suppose that MarkovSim always produces a result that is within ratio $1+\hat{\varepsilon}$ of the correct value. (This is still an idealized situation, since the specification of MarkovSim allows large deviations with probability $\frac{1}{4}$.) Then a simple induction on n establishes that CountPM returns a result that is within ratio $(1+\hat{\varepsilon})^n$ of $N(G)$ when applied to a

bipartite graph G with $n+n$ vertices. In particular, the result returned from the top-level call to COUNTPM is within ratio $(1+\hat{\varepsilon})^{n_0} = (1+\varepsilon/2n_0)^{n_0} < 1+\varepsilon$, as required.

Finally, suppose that MARKOVSIM is as specified, and produces a result that is within ratio $1+\hat{\varepsilon}$ with probability at least $\frac{3}{4}$. We call the sequence of trials carried out in line (3) an *experiment*, and say that the experiment is a *success* if the median of the $2t+1$ trials approximates $N(G)$ within ratio $1+\hat{\varepsilon}$. We choose t so that the probability that *every* experiment is a success is at least $\frac{3}{4}$. An easy induction on n establishes that the number of experiments is bounded by 8^n. (This is a gross overestimate, as will be apparent once we have completed the analysis of the time complexity of COUNTPM.) By Lemma 6.1 of [6], the failure probability of a single experiment may be reduced to $8^{-(n_0+1)}$ by setting $t = 18(n_0+1) = O(n_0)$. (The point here is that t need not be very large.) With this choice of t, the probability that *every* experiment is a success is at least $\frac{3}{4}$. This completes the validation of COUNTPM.

Let $T(n,\varepsilon)$ be the (worst-case) time complexity of COUNTPM as a function of the size of G and the accuracy parameter ε. (Technically, the time complexity depends on $\hat{\varepsilon}$ and t, and hence should be a function also of n_0; however, the dependence on n_0 is cubic, and can safely be swept under the carpet in the current context.) We show, by induction on n, that $T(n,\varepsilon) \le \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n)$, where c is a suitable constant. From Figure 2,

$$T(n,\varepsilon) \le \max\{T_1(n,\varepsilon), T_2(n,\varepsilon)\}, \tag{1}$$

where $T_1(n,\varepsilon)$ and $T_2(n,\varepsilon)$ are, respectively, the time complexities conditional on whether the "then" or "else" part of the if-statement is selected at line (2). The time complexity T_1 is determined by the calls to TESTEXPANSION in line (1), and the sequence of $2t+1$ trials using MARKOVSIM in line (3). From Corollary 3 and Lemma 4, setting $\alpha = n^{-1/2}$, we have

$$\begin{aligned} T_1(n,\varepsilon) &\le \exp(O(n^{1/2}\log n)) + \varepsilon^{-2}\exp(O(n^{1/2}\log^2 n)) \\ &\le \varepsilon^{-2}\exp(O(n^{1/2}\log^2 n)). \end{aligned} \tag{2}$$

The time complexity T_2 is determined by the call to TESTEXPANSION in line (1) and the recursive calls to COUNTPM in line (6). Let $S(k)$ be an upper bound on the number of subproblems generated (i.e., the number of times the for-loop is executed) when the set A returned by TESTEXPANSION has size k. By Lemma 4,

$$T_2(n,\varepsilon) \le \exp(O(n^{1/2}\log n)) + \max\{S(k)[T(k,\varepsilon)+T(n-k,\varepsilon)]\colon 1 \le k \le \tfrac{1}{2}n\}. \tag{3}$$

We proceed to estimate each of the quantities appearing on the right-hand side of (3). Suppose the set A returned by TESTEXPANSION is of size k; then $|\Gamma_G(A)| = k+d$, where $d < 2\alpha k = 2kn^{-1/2}$, and we may take as our upper bound on the number of subproblems:

$$S(k) = \binom{k+d}{k} \le n^d \le \exp(2kn^{-1/2}\ln n) \tag{4}$$

$$\le \exp(n^{1/2}\ln n). \tag{5}$$

Applying the induction hypothesis we obtain (since $k \le \frac{1}{2}n$)

$$T(k,\varepsilon) \le \varepsilon^{-2}\exp(ck^{1/2}\ln^2 k) \le \varepsilon^{-2}\exp(\tfrac{3}{4}cn^{1/2}\ln^2 n), \tag{6}$$

and, again,

$$\begin{aligned} T(n-k,\varepsilon) &\le \varepsilon^{-2}\exp(c(n-k)^{1/2}\ln^2(n-k)) \\ &\le \varepsilon^{-2}\exp(c(1-\tfrac{1}{2}kn^{-1})n^{1/2}\ln^2 n) \\ &= \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n - \tfrac{1}{2}ckn^{-1/2}\ln^2 n). \end{aligned} \tag{7}$$

Now suppose $c \ge 12$. By combining inequalities (5) and (6), we have

$$S(k)T(k,\varepsilon) \le \varepsilon^{-2}\exp(\tfrac{7}{8}cn^{1/2}\ln^2 n), \tag{8}$$

and by combining (4) and (7),

$$\begin{aligned} S(k)T(n-k,\varepsilon) &\le \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n - \tfrac{1}{4}ckn^{-1/2}\ln^2 n) \\ &\le \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n - 2n^{-1/2}) \\ &\le (1-n^{-1/2})\varepsilon^{-2}\exp(cn^{1/2}\ln^2 n). \end{aligned} \tag{9}$$

It is immediate from (2) that $T_1(n,\varepsilon) \le \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n)$ for c sufficiently large, and it follows easily from (3), (8), and (9) that $T_2(n,\varepsilon) \le \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n)$. Finally, from (1), $T(n,\varepsilon) \le \varepsilon^{-2}\exp(cn^{1/2}\ln^2 n)$, which completes the inductive step. □

3. Extensions and Open Problems. The algorithm presented here rests on a tradeoff: a graph with a high expansion factor is amenable to direct solution using an existing approximation algorithm, while a graph with a low expansion factor admits an efficient recursive decomposition into subproblems. The exponent $\frac{1}{2}$ appearing in the expression for the time complexity arises quite naturally from a consideration of the crossover point for this tradeoff, which is near $\alpha = n^{-1/2}$, and it seems unlikely that a further improvement in asymptotic time complexity is possible without the introduction of some new technique. The ultimate goal must surely be to construct a fully polynomial approximation scheme (or to provide convincing complexity-theoretic evidence that no such scheme exists). However, a reasonable intermediate goal might be to design an approximation algorithm with "quasi-polynomial" time complexity, i.e., with time complexity $O(\exp(\mathrm{poly}(\log n)))$.

Of possible generalizations of the problem considered in this note, the two most natural appear to be:

(1) Estimating the permanent of a general non-negative integer matrix.
(2) Estimating the number of perfect matchings in a general (nonbipartite) graph.

Provided the entries in the matrix are small, the first of these problems can be handled by reduction to the 0,1 case. Suppose A is an $n \times n$ matrix of nonnegative integers whose binary expansions are uniformly bounded in length by k. It is not difficult to show that there exists an $m \times m$ matrix A' with 0,1-entries for which $\mathrm{per}(A') = \mathrm{per}(A)$; here $m = O(nk)$, and the matrix A' is easily constructed in polynomial time. However, if the matrix entries are very large integers (of roughly n bits) it will be seen that this reduction provides no gain in efficiency over exact computation using Ryser's algorithm. The question of whether Ryser's method can be beaten for general nonnegative matrices remains open.

It seems plausible that the second generalization (to nonbipartite graphs) can be handled by methods similar to those employed in this note. The stumbling block is Lemma 2: even if the lemma itself can be extended to nonbipartite graphs G, the proof presented here cannot.

References

[1] A. Z. Broder, How hard is it to marry at random? (On the approximation of the permanent), *Proceedings of the* 18*th ACM Symposium on Theory of Computing*, 1986, pp. 50–58. Erratum, *Proceedings of the* 20*th ACM Symposium on Theory of Computing*, 1988, p. 551.

[2] A. Frieze and M. Jerrum, An analysis of a Monte Carlo algorithm for estimating the permanent, *Combinatorica*, **15** (1995), 67–83.

[3] C. D. Godsil and I. Gutman, On the matching polynomial of a graph, *Algebraic Methods in Graph Theory, I* (L. Lovász and V. T. Sós, editors), Colloquia Mathematica Societatis János Bolyai, Vol. 25, North-Holland, Amsterdam, 1981, pp. 241–249.

[4] M. Jerrum, An analysis of a Monte Carlo algorithm for estimating the permanent, *Proceedings of the* 3*rd Conference on Integer Programming and Combinatorial Optimization*, CORE, Louvain-la-Neuve, Belgium, April 1993, pp. 171–182.

[5] M. Jerrum and A. Sinclair, Approximating the permanent, *SIAM Journal on Computing*, **18** (1989), 1149–1178.

[6] M. R. Jerrum, L. G. Valiant, and V. V. Vazirani, Random generation of combinatorial structures from a uniform distribution, *Theoretical Computer Science*, **43** (1986), 169–188.

[7] N. Karmarkar, R. Karp, R. Lipton, L. Lovász, and M. Luby, A Monte-Carlo algorithm for estimating the permanent. *SIAM Journal on Computing*, **22** (1993), 284–293.

[8] R. M. Karp and M. Luby, Monte Carlo algorithms for enumeration and reliability problems, *Proceedings of the* 24*th IEEE Symposium on Foundations of Computer Science*, 1983, pp. 56–64.

[9] L. Lovász and M. D. Plummer, *Matching Theory*, North-Holland, Amsterdam, 1986.

[10] H. Minc, *Permanents*, Addison-Wesley, Reading, MA, 1978.

[11] A. Sinclair, *Randomised Algorithms for Counting and Generating Combinatorial Structures*, Ph.D. Thesis, University of Edinburgh, June 1988.

[12] A. Sinclair, *Algorithms for Random Generation and Counting*: *a Markov Chain Approach*, Birkhäuser, Boston, 1993.

[13] L. G. Valiant, The complexity of computing the permanent, *Theoretical Computer Science*, **8** (1979), 189–201.