Approximately Counting Perfect Matchings in General Graphs

Martin Fürer * Shiva Prasad Kasiviswanathan
Computer Science and Engineering Pennsylvania State University
University Park, PA 16802. {furer, kasivisw}@cse.psu.edu

Abstract

So far only one approximation algorithm for the number of perfect matchings in general graphs is known. This algorithm of Chien [2] is based on determinants. We present a much simpler algorithm together with some of its variants. One of them has an excellent performance for random graphs, another one might be a candidate for a good worst case performance. We also present an experimental analysis of one of our algorithms.

1 Introduction

In this paper we describe a new method for counting the number of perfect matchings $M(G)$ for any graph $G = (V, E)$ with $|V| =$ even. Counting perfect matchings for the bipartite graph with bipartite adjacency matrix A, is equivalent to computing the permanent of that matrix. Computing permanents was shown by Valiant [16] to be #P-complete. Counting the number of perfect matchings for general graphs is #P-complete as it is already #P-complete for the bipartite case. Therefore, one can only hope for efficient approximation algorithms.

A number of approaches have been designed to approximately count the number of perfect matchings for bipartite graphs. Recently, an fpras for computing the permanent of an arbitrary matrix with non-negative entries was proposed by Jerrum, Sinclair, Vigoda [8]. It is based on the Markov chain Monte-Carlo approach. However, their proof relies crucially on the bipartiteness of the underlying graph and their algorithm doesn't generalize to arbitrary graphs. Only one approximation algorithm (Chien [2]) is known for counting the number of perfect matchings in non-bipartite graphs. We propose a very simple randomized algorithm and present several potentially more efficient versions. They are reminiscent of similar algorithms pioneered by Knuth [10] for estimating the size of trees, and later extended by Purdom [12]. Some of the same ideas have been used by Rasmussen [13] for approximating permanents of (0-1) matrices.

*Research supported in part by NSF Grant CCR-0209099

2 Definitions

Let F be some function from Σ^* to the natural numbers. A fully polynomial randomized approximation scheme (a.k.a. fpras) for F is a randomized algorithm that accepts an input $x \in \sum^\star$ together with an accuracy parameter $\epsilon \in (0, 1]$, outputs a number X (a random variable depending on the coin tosses of the algorithm) such that,

$$\Pr[(1-\epsilon)F(x) \leq X \leq (1+\epsilon)F(x)] \geq \frac{3}{4}$$

and runs in time polynomial in $|x|$ and ϵ^{-1}. The probability of $\frac{3}{4}$ can be boosted to $1-\delta$ for any $0 < \delta < 1/4$ by outputting the median of $\mathrm{O}(\log \delta^{-1})$ independent trials [9]. Suppose we would like to estimate F and have a randomized algorithm running in time polynomial in $|x|$ whose output is a random variable X such that $\mathrm{E}[X] = F(x)$ and $\mathrm{E}[X^2]$ is finite. Then we can repeat this experiment as many times as we wish, and the outcomes of the successive trials are independent and identically distributed. Let X_i be the outcome of the i^{th} trial. A straightforward application of Chebychev's inequality shows that, if we conduct $\Theta(\frac{\mathrm{E}[X^2]}{\mathrm{E}[X]^2}\epsilon^{-2})$ trials and take the median, we have a randomized approximation scheme for F. Together, the complexity of performing the stochastic experiment, and the ratio of $\mathrm{E}[X^2]/\mathrm{E}[X]^2$ (a.k.a. critical ratio) determine the efficiency of the algorithm.

3 Estimators

We start this section with a simple estimator and build on it as we go to get better but more complex estimators. We also show how all these estimators fit into a generalized scheme for approximating $M(G)$ and show how this scheme leads to unbiased estimations of $M(G)$. We round off by suggesting a good estimator for $M(G)$ when the graph G is random.

The idea behind SIMPLE (Figure 1) is to repeatedly pick one vertex deterministically and match it uniformly at random with one of its neighbors. Remove both the vertex and the matched vertex and all edges incident on them to get $G_{\ell k}$, recurse on the remaining $G_{\ell k}$. This

SIMPLE (Random Approximator for Counting Perfect Matchings)

Input: Graph $G = (V, E)$ with some fixed vertex ordering
Output: Estimate X_G for the number of Perfect Matchings

if $n = 0$ **then** $X_G = 1$
else
choose vertex ℓ (lowest numbered vertex remaining)
$W = \{j : (\ell, j) \in E\}$
if $W = \emptyset$ **then** $X_G = 0$
else
choose k u.a.r. from W
compute $X_{G_{\ell k}}$ and use it to compute $X_G = |W| X_{G_{\ell k}}$

Figure 1: SIMPLE Estimator

approach may stop early, because at some point the current vertex might have no neighbors.

An obvious modification to this estimator, which could lead to an improved performance, is to introduce a systematic bias so that the choice of a neighbor is not uniform at random. One could assign different probabilities to the neighbors of ℓ and pick a neighbor k with its probability. Knuth [10] analyzed a variant of such an estimator under some tight conditions on probabilities.

We use Knuth's notation for trees. Nodes are finite sequences $(x_1, \ldots, x_k)$ satisfying property $P_k(x_1, \ldots, x_k)$. The root is the empty sequence. If $P_{k+1}(x_1, \ldots, x_k, x_{k+1})$ holds, then also $P_k(x_1, \ldots, x_k)$ holds and $(x_1, \ldots, x_k, x_{k+1})$ is a child of $\vec{x} = (x_1, \ldots, x_k)$. Nodes have arbitrary given costs. We want to estimate the cost of a tree defined as the sum of the costs of its vertices. Knuth's recursive procedure starts at the root. When it is in a leaf $(x_1, \ldots, x_k)$ then it returns with $C = c(x_1, \ldots, x_k)$. When it is in another node $\vec{x} = (x_1, \ldots, x_k)$, it selects a child $(x_1, \ldots, x_k, x_{k+1}(j))$ with probability $\Pr[\vec{x}, j]$ and makes a recursive call to it. When it returns with a cost estimate C it divides it by $\Pr[\vec{x}, j]$, adds $c(\vec{x})$ to it and returns.

THEOREM 3.1. ***Knuth [10]*** *Let tree* $T = \{(x_1, x_2, \ldots, x_n) \,|\, n \geq 0 \text{ and } P_n(x_1, x_2, \ldots, x_n) \text{ holds}\}$ *and* $cost(T)$ *be the function to be estimated. Let* $T(x_1, x_2, \ldots, x_k)$ *be the subtree rooted at* $(x_1, \ldots, x_k)$ *and let* $(x_1, \ldots, x_k, x_{k+1}(j))$ *be the* j^{th} *child of* $(x_1, \ldots, x_k)$. *If the probabilities* $\Pr[\vec{x}, j]$ *(probability of going from* k *to its* j^{th} *successor) satisfy*

$$\frac{cost(T((x_1, \ldots, x_k, x_{k+1}(j))))}{\Pr[\vec{x}, j]} \leq \alpha \frac{cost(T((x_1, \ldots, x_k, x_{k+1}(i))))}{\Pr[\vec{x}, i]}$$

for all $i, j, \vec{x}$ *and some constant* $\alpha \geq 1$, *then the variance of the output* C *computed by Knuth's algorithm is at most*

$$\left(\left(\frac{\alpha^2 + 2\alpha + 1}{4\alpha}\right)^n - 1\right) cost(T)^2$$

The above theorem is applicable only under very restrictive conditions. We note that we needn't always require that the probabilities be good approximations to the relative subtree costs. The major harm to the variance is done by choosing probabilities too low, higher probabilities tend to have lesser influence on the variance. To illustrate this fact let us consider a variant of the graph example introduced by Jerrum, Sinclair & Vigoda [8] (Figure 2) which has exponentially many perfect matchings. If we use a procedure to eliminate useless edges at every step (can be done in polynomial time with Edmond's algorithm [3]) we see that the perfect matching consisting of all horizontal edges is chosen with probability $\Theta(1)$ (i.e. huge probability) and we still can bound variance by a small constant. Taking this fact into account we modify the above theorem to make it more widely applicable.

THEOREM 3.2. *Let* d *be the number of children of* $\vec{x} = (x_1, \ldots, x_k)$. *If the probabilities* $\Pr[\vec{x}, j]$ *satisfy*

$$\frac{cost(T((x_1, \ldots, x_k, x_{k+1}(j))))}{cost(T(x_1, \ldots, x_k))} \leq \alpha \Pr[\vec{x}, j]$$

for all j, *and some constant* $\alpha \geq 1$, *then the variance of* C *as computed by Knuth's algorithm is at most*

$$(\alpha^n - 1) cost(T)^2$$

Proof. The idea is similar to that of Knuth. We use T_i to denote $T((x_1, \ldots, x_k, x_{k+1}(i))$ and $\Pr[j]$ to denote

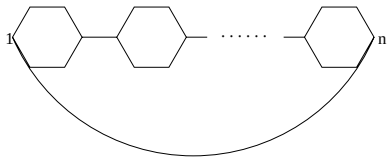

Figure 2: Chain of Hexagons

$\Pr[\vec{x}, j]$. We note that

$$\sum_{1\le i<j\le d} \Pr[i]\Pr[j]\left(\frac{cost(T_i)}{\Pr[i]} - \frac{cost(T_j)}{\Pr[j]}\right)^2$$
$$= \sum_{1\le j\le d} \frac{cost(T_j)^2}{\Pr[j]} - \left(\sum_{1\le j\le d} cost(T_j)\right)^2$$

From our assumption on probabilities we get

$$\sum_{1\le j\le d} \frac{cost(T)^2}{\Pr[j]} \le \alpha \left(\sum_{1\le j\le d} cost(T_j)\right)^2$$

Now let C denote the random variable at node $(x_1, \ldots, x_k)$ and let C_j denote the random variable at $(x_1, \ldots, x_k(j))$. From Knuth we know that the variance of the estimator C is

$$\mathrm{Var}[C] = \sum_{1\le j\le d} \frac{\mathrm{Var}[C_j]}{\Pr[j]} + \sum_{1\le i<j\le d} \Pr[i]\Pr[j]\left(\frac{cost(T_i)}{\Pr[i]} - \frac{cost(T_j)}{\Pr[j]}\right)^2$$
$$\le \sum_{1\le j\le d} \frac{\mathrm{Var}[C_j]}{\Pr[j]} + (\alpha - 1)cost(T)^2$$

By using induction we complete the proof. □

The *cost* function corresponds to $M(G)$ in our case. One way to assign probabilities is to bias the sampling in favor of low degree during the picking of a neighbor k. We also pick as ℓ, the vertex with lowest degree. We call this estimator GREEDY (Figure 3). The optimal choice of the probabilities is proportional to the unknown number of perfect matchings containing the selected partial matching. In a random bipartite graph, the expected number of such perfect matchings is indirectly proportional to the number of additional neighbors of the matched vertex k. This motivates our choices of probabilities. There is no reason to believe that the above choice of probabilities couldn't be beneficial for general graphs. However with all this intelligence, the worst case of GREEDY turned out to be difficult to analyze, so we turned to experimental simulation to test its performance (Section 5). Those results helped us to come up with the following conjecture.

Conjecture 3.1. *GREEDY has a good worst case performance i.e, the critical ratio is small even though not polynomial.*

Another enhancement would be handling vertices with degree 2 in a special way. For vertices with degree 2, one could condense the graph as shown in Figure 5 and recurse over the remaining graph. All such algorithms are unbiased estimators. We show that for some general scheme GEN(Figure 4).

Theorem 3.3. *Let $G = (V, E)$ be a graph, and let X_G be the estimator as given by GEN. Then*

$$\mathrm{E}(X_G) = M(G)$$

Proof. It is sufficient to show that every fixed perfect matching has an expected contribution of 1 towards X_G. This implies, the expected value of X_G is the number of perfect matchings. The proof is by induction on the number of recursive calls. The case $n = 0$ is trivial. Each induction step involves two parts:
Part 1: Assume we alter the graph G in Step 1 to get G'. By induction hypothesis we know that every perfect matching has an expected contribution of 1 towards $X_{G'}$. Since the number of perfect matchings doesn't change from G' to G ($X_G = X_{G'}$), the same contribution is also assigned to G.
Part 2: Let G denote the graph with ℓ as the vertex chosen in Step 2. Assume that $W = \{k : (\ell, k) \in E\} \neq \emptyset$. For all $k \in W$, we know by induction hypothesis that every perfect matching in $G_{\ell k}$ has an expected contribution of 1 to $X_{G_{\ell k}}$. Let $\Pr[\ell, k]$ denote the probability that edge (ℓ, k) is chosen. Now, the definition of X_G is

$$\forall k \in W,\ X_G = X_{G_{\ell k}}/\Pr[\ell, k] \text{ with probability } \Pr[\ell, k]$$

Thus, every edge (ℓ, k) contributes a factor of $1/\Pr[\ell, k]$ with probability $\Pr[\ell, k]$ to X_G. Hence, any fixed perfect matching in G which contains the edge (ℓ, k) also has an expected contribution of 1 to X_G. □

A closer look at random graphs tells us that this estimator often makes most of its mistakes towards the

GREEDY (Greedy Random Approximator for Counting Perfect Matchings)

Input: Graph $G = (V, E)$ with some fixed vertex ordering, $|V| = n = 2h$
Output: Estimate X_G for the number of Perfect Matchings

if $n = 0$ **then** $X_G = 1$
else
    eliminate edges not contributing towards any perfect matching
    choose vertex ℓ (a vertex with the lowest degree)
    $W = \{j : (\ell, j) \in E\}$
    if $W = \emptyset$ **then** $X_G = 0$
    else
        choose k from W with probability $p(k)$ inversely proportional to $deg(k) - 1$
        compute $X_{G_{\ell k}}$ and use it to compute $X_G = \frac{1}{p(k)}(X_{G_{\ell k}})$

Figure 3: GREEDY Estimator

GEN (Generalized Random Approximator for Counting Perfect Matchings)

Input: Graph $G = (V, E)$ with some fixed vertex ordering
Output: Estimate X_G for the number of Perfect Matchings

1. Use any function to alter G without changing the number of perfect matchings.
2. Select any vertex ℓ from the graph deterministically (any criterion).
3. Select a probability distribution on the neighbors of ℓ such that each neighbor k of ℓ has a positive probability $p(k)$.
4. Choose a neighbor k of ℓ with probability $p(k)$.
5. Remove both the vertices and all edges incident on them to get $G_{\ell k}$.
6. Recursively compute the estimator for remaining graph $G_{\ell k}$.
7. Output the estimator $X_G = X_{G_{\ell k}}/p(k)$

Figure 4: Generalized Estimator for counting $M(G)$

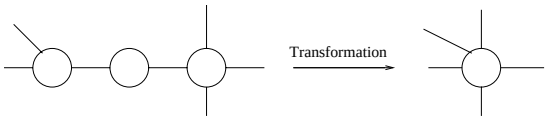

Figure 5: Transformation for Vertex with Degree 2

REP (Repeated Random Approximator for Counting Perfect Matchings)

Input: Graph $G = (V, E)$ with some fixed vertex ordering, $|V| = n = 2h$
Output: Estimate X_G for the number of Perfect Matchings

if $h = 0$ **then** $X_G = 1$
else
    choose vertex ℓ (lowest numbered vertex remaining)
    $W = \{j : (\ell, j) \in E\}$
    if $W = \emptyset$ **then** $X_G = 0$
    else
        if h is a branching point **then** $K =$ branching factor
        else $K = 1$
        for $i = 1$ **to** K **do**
            choose $k(i)$ u.a.r. from W
            compute $X_{G_{\ell k(i)}}$
        $X_G = |W|(\frac{1}{K}\sum_{i=1}^{K} X_{G_{\ell k(i)}})$

Figure 6: REP Estimator

end when the graph becomes small. This motivates us to increase the precision as the size of the graph decreases. At every level designated as branching point, we do multiple recursive calls (branching factor) on the subgraph. For simplicity we assume that both branching points and branching factor are precomputed before the algorithm starts. We call this estimator REP (Figure 6). The resulting computation has a tree structure. We use computation trees with branching points at heights $h = 2^j (j \geq 1)$ and a branching factor of 3 as shown in Figure 7. Also it can easily be verified that such an estimator is unbiased. The idea is similar to one used by Karger and Stein [15] to obtain a faster Min-Cut algorithm. Also, the same idea has already been analyzed for the bipartite case by the authors [5].

4 Performance Bounds

In this section we look into the performance of the REP estimator. We bound its worst case performance and analyze its behavior on random graphs. We show that this estimator performs well on random graphs, and improves the previous bound of $O(n\mathcal{M}(n)\omega)$, where $\mathcal{M}(n)$ is the time required to perform $n \times n$ matrix multiplications and $\omega = \omega(n)$ is any function satisfying $\omega(n) \to \infty$ as $n \to \infty$. Even though we conjecture that adding some intelligent vertex selection mechanism as in GREEDY may actually result in better performance of REP in the worst case, it also makes the problem difficult to analyze.

4.1 Worst Case Performance of SIMPLE/REP

Let $(n)!!$ (called semi-factorial [6]) denote $n(n-2)(n-4)\ldots 1 = (2m)!/(2^m)m!$ when $n = 2m-1$ is odd.

THEOREM 4.1. *Let $G = (V, E)$ be a graph with $|V| = n = 2h$, and let X_G be the estimator defined by REP. Then*

$$\mathrm{E}[X_G^2] \leq M(G)(n-1)!!$$

Proof. Assume that $W = \{j : (\ell, j) \in E\} \neq \emptyset$. Let $\hat{X}_G(i)$ (for $i = 1, \ldots, K$) be the auxiliary estimator defined by the ith branch, and let $\hat{X}_G = \hat{X}_G(1)$.

$$\begin{aligned}
\mathrm{E}[X_G^2] &= \mathrm{E}[(\frac{1}{K}\sum_{i=1}^{K} \hat{X}_G(i))^2] \\
&= \frac{1}{K^2}(\sum_{i=1}^{K} \mathrm{E}[\hat{X}_G^2(i)] + \sum_{i=1}^{K}\sum_{j=1, j\neq i}^{K} \mathrm{E}[\hat{X}_G(i)]E[\hat{X}_G(j)]) \\
&= \frac{1}{K^2}(K\mathrm{E}[\hat{X}_G^2] + K(K-1)\mathrm{E}[\hat{X}_G]^2) \\
&= \frac{1}{K}\sum_{j\in W} \mathrm{E}[\hat{X}_G^2 | k(1) = j]\Pr[k(1) = j] + (1 - 1/K)M(G)^2 \\
&\leq \frac{|W|}{K}\sum_{j\in W} \mathrm{E}[(X_{G_{\ell j}})^2] + (1 - 1/K)M(G)(n-1)!! \\
&\leq \frac{1}{K}\sum_{j\in W} M(G_{\ell j})|W|(n-3)!! + (1 - 1/K)M(G)(n-1)!! \\
&\leq M(G)(n-1)!!
\end{aligned}$$

where the last step follows from the fact that $|W| \leq (2n-1)$ and $\sum_{j\in W} M(G_{\ell j}) = M(G)$. □

Thus, the bound on the critical ratio is

$$\frac{\mathrm{E}[X_G^2]}{\mathrm{E}[X_G]^2} \leq \frac{(n-1)!!}{M(G)}$$

Algorithm	$\mathrm{E}[X_G^2]$	$\mathrm{E}[X_G]^2$	Critical ratio
Chien [2]	$6(2^{n/6}) + (2^{n/2})$	$(2^{n/6}+1)^2$	Exponential ($\Theta(2^{n/6})$)
SIMPLE + Useless Edge Removal	$3 + (3/2)2^{n/3}$	$(2^{n/6}+1)^2$	Constant ($< 3/2$)

Table 1: Performance of SIMPLE on Figure 2

The analysis shows a devastatingly bad worst case bound on the performance of the REP(or SIMPLE) estimator, but one might expect that combining these estimators with one or more of the previous ideas could lead to a better critical ratio. To illustrate this possibility, let us consider the performance of the estimators on the graph from Figure 2. Table 1 summarizes the result.

4.2 Random Graphs The two most frequently used models of random graphs are $\mathcal{G}(n,p)$ and $\mathcal{G}(n,m)$ (see [1] for an extensive treatment). We use $\mathcal{G}(n)$ to represent the set of all graphs with vertices $\{1,\ldots,n\}$ and $\mathcal{G}(n,m)$ to represent the subset of those graphs $\mathcal{G}(n)$ with exactly m edges, both with uniform distribution. We use E_σ to represent the mean over the coin-tosses of the estimator, $\mathrm{E}_\mathcal{G}$ to represent the mean over $\mathcal{G}(n)$, and $\mathrm{E}_{\mathcal{G}(n,m)}$ to represent the mean over $\mathcal{G}(n,m)$. As the output (i.e., the random variable X) depends on both, the input graph and the coin tosses of the algorithm, we can use expressions like $\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X]]$. Here, $\mathrm{E}_\sigma[X]$ is a random variable defined on the set of graphs. We investigate the performance of the REP estimator for the most commonly used model $\mathcal{G}(n) = \mathcal{G}(n,1/2)$.

Our idea of REP is the following. As we work with smaller matrices the time decreases drastically. Without increasing the asymptotic complexity, we can branch at powers of s by any number less than s^2. It turns out that for $s = 2$, branching by 3 is actually just sufficient to keep the critical ratio bounded. The situation becomes simple, if we investigate

$$R(X) = \frac{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X^2]]}{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X]]^2}$$

which we might call the "critical ratio of averages." It is closely related to the critical ratio of almost all matrices. For random bipartite graphs, in a single run, the "critical ratio of averages" doubles when the dimension is increased from n to $2n$. Doing K parallel runs decreases the variance of X by a factor of K. Thus, the "critical ratio of averages" changes from $R(X)$ to $(R(X)-1)/K+1$. Therefore, with branching by 3 at powers of 2 the "critical ratio of averages" grows from ≤ 2 to ≤ 4 and is reduced again to ≤ 2 (also in the non-bipartite case).

THEOREM 4.2. *The running time of the algorithm* REP *with branching factor of 3 and branching points of* 2^i *for* $(i \ge 1)$ *is* $\mathrm{O}(n^2)$.

Proof. Let $2^{i-1} < n \le 2^i$. Then the running time: Between top and 1^{st} branching level is $< n^2 \le 2^{2i}$. Between 1^{st} and 2^{nd} branching level is $< 3(2^{i-1})^2$. Between 2^{nd} and 3^{rd} branching level is $< 9(2^{i-2})^2$. As this forms a geometric series, the total running time is $\mathrm{O}(n^2)$. □

Our result rests on the following weak version of a result of Janson [7], which we state here.

THEOREM 4.3. ***(Janson [7], Chien [2])*** *Let* $G \in \mathcal{G}(n,m)$ *where* $\frac{m^2}{n^3} \to \infty$. *Let* $p = m/\binom{n}{2}$, *then*

$$\mathrm{E}[M(G)] = (n-1)!!\, p^{n/2} \exp\left(-\frac{1-p}{4p} + \mathrm{O}\left((1-p)\frac{n^3}{m^2}\right)\right)$$

and

$$\frac{\mathrm{E}[M(G)^2]}{\mathrm{E}[M(G)]^2} = 1 + \mathrm{O}\left(\frac{n^3}{m^2}\right)$$

LEMMA 4.1. *Let* $\omega = \omega(n)$ *be any function satisfying* $\omega \to \infty$ *as* $n \to \infty$. *Then for almost all graphs* G, *with* X_G *being any unbiased estimator of* $M(G)$,

$$\frac{\mathrm{E}_\sigma[X_G^2]}{\mathrm{E}_\sigma[X_G]^2} \le \omega \frac{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X_G^2]]}{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X_G]]^2}$$

Proof. This proof for the case when the graph is bipartite has already been done by Rasmussen [13] and is based on a result from Frieze and Jerrum [4]. The proof directly follow from these results along with the previous theorem. □

Let $\widehat{\mathrm{REP}}$ be an auxiliary random Approximator. Its only difference from REP is that K=1 at the start, i.e., there is no branching in the root of the computation tree even if $n = 2h = 2^j$. The random variables $\hat{X}_h$ and X_h are the outputs of $\widehat{\mathrm{REP}}$ and REP respectively when the input G_{2h} is a random graph from $\mathcal{G}(2h)$. To model the quality of $\widehat{\mathrm{REP}}$ and REP, we introduce two terms $\hat{R}(h)$ and $R(h)$. $\hat{R}(h)$ models the "critical ratio of averages" of the auxiliary approximator $\widehat{\mathrm{REP}}$, while $R(h)$ models the "critical ratio of averages" of REP until height h.

$$\hat{R}(h) = \frac{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[\hat{X}_h{}^2]]}{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[\hat{X}_h]]^2} \quad \text{and} \quad R(h) = \frac{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X_h^2]]}{\mathrm{E}_\mathcal{G}[\mathrm{E}_\sigma[X_h]]^2}$$

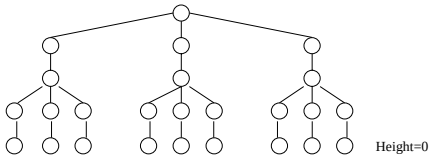

Figure 7: Run of the REP

The proofs are organized as follows: We establish the recursive relationship between $R(h)$ and $\hat{R}(h)$ in Theorems 4.4 and 4.5. With Theorems 4.6 and 4.7, we establish the claimed performance bounds. The following theorem shows how $\hat{R}(h)$ varies as a function of $R(2^{\lfloor \lg(h-1) \rfloor})$, i.e. R at the previous branching point.

THEOREM 4.4. *Let G_{2h} denote a random graph from $\mathcal{G}(2h)$, and let $R(h)$ and $\hat{R}(h)$ be the functions defined above. Then*

$$\hat{R}(h) \leq \begin{cases} 2 & \text{for } h = 1 \\ \dfrac{2h}{2^{\lfloor \lg(h-1) \rfloor + 1} + 1} R(2^{\lfloor \lg(h-1) \rfloor}) & \text{for } h > 1 \end{cases}$$

Proof. Let M_i denote a binomial variable with parameters i and $p = \frac{1}{2}$. Let M be the degree of the first selected vertex ℓ. Thus $M = M_{2h-1}$.

$$\begin{aligned} & \mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[\hat{X}_h^2]] \\ &= \sum_{m=0}^{2h-1} \mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[\hat{X}_h^2] \mid M = m] \Pr[M = m] \\ &= \sum_{m=0}^{2h-1} \mathrm{E}_{\mathcal{G}}[m^2 \mathrm{E}_{\sigma}[(X_{h-1})^2]] \Pr[M = m] \\ &= \mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[(X_{h-1})^2]] \, \mathrm{E}(M_{2h-1}^2) \\ &= \mathrm{E}[M_{2h-1}^2] \mathrm{E}[M_{2h-3}^2] \ldots \mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[(X_{2^{\lfloor \lg(h-1) \rfloor}})^2]] \end{aligned}$$

The denominator is

$$\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[\hat{X}_h]]^2 = \left(\frac{(2h-1)!!}{2^h} \right)^2 = \prod_{i=1}^{h} \mathrm{E}[M_{2i-1}]^2$$

$$\begin{aligned} & \hat{R}(h) \\ &= \frac{\mathrm{E}[M_{2h-1}^2]}{\mathrm{E}[M_{2h-1}]^2} \frac{\mathrm{E}[M_{2h-3}^2]}{\mathrm{E}[M_{2h-3}]^2} \cdots \frac{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[(X_{2^{\lfloor \lg(h-1) \rfloor}})^2]]}{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[X_{2^{\lfloor \lg(h-1) \rfloor}}]]^2} \\ &= \frac{(2h-1)(2h)/4}{(2h-1)^2/4} \frac{(2h-3)(2h-2)/4}{(2h-3)^2/4} \cdots \frac{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[(X_{2^{\lfloor \lg(h-1) \rfloor}})^2]]}{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[X_{2^{\lfloor \lg(h-1) \rfloor}}]]^2} \\ &\leq \frac{2h}{2^{\lfloor \lg(h-1) \rfloor + 1} + 1} \frac{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[(X_{2^{\lfloor \lg(h-1) \rfloor}})^2]]}{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_{\sigma}[X_{2^{\lfloor \lg(h-1) \rfloor}}]]^2} \\ &= \frac{2h}{2^{\lfloor \lg(h-1) \rfloor + 1} + 1} R(2^{\lfloor \lg(h-1) \rfloor}) \end{aligned}$$

□

Before venturing into showing the dependence of $R(h)$ on $\hat{R}(h)$ we establish a few important technical lemmas. The following lemma shows a bound of a higher moment of a binomial distribution. A lot of similar results have appeared in literature (see [11] for more details).

LEMMA 4.2. *For $n \geq 0$ we have*

$$\sum_{j=0}^{\binom{n}{2}} \binom{\binom{n}{2}}{j} j^n = \mathrm{O}\left(\left(\frac{n^2 - n}{4} \right)^n 2^{\binom{n}{2}} \right)$$

Proof. The term $\binom{\binom{n}{2}}{j} j^n$ has its maximum value around $j = \binom{n}{2}/2 + n$. The idea is to split the summation into three parts, from 0 to $\frac{1}{2}\binom{n}{2}$, from $\frac{1}{2}\binom{n}{2} + 1$ to $\frac{1}{2}\binom{n}{2} + n$, and from $\frac{1}{2}\binom{n}{2} + n + 1$ to $\binom{n}{2}$.

Each of these three parts can be upper bounded by a constant multiple of

$$\left(\frac{n^2 - n}{4} \right)^n 2^{\binom{n}{2}}$$

□

We now prove that the number of perfect matchings in a random graph is fairly tightly clustered. A similar result in the case of bipartite random graphs has been shown for $\mathcal{G}(n, m)$ [4] and has later been extended to the $\mathcal{G}(n, p)$ model [5].

LEMMA 4.3. *Let G_n be a random graph from $\mathcal{G}(n)$. Then for some constant c independent of n*

$$\frac{\mathrm{E}[(\mathrm{M}(G_n))^2]}{\mathrm{E}[\mathrm{M}(G_n)]^2} \leq c$$

Proof. Conditioning the numerator on the number of edges m.

$$\begin{aligned} & \mathrm{E}[(M(G_n))^2] \\ &= \Pr\left[m < \frac{1}{4}\binom{n}{2} \right] \mathrm{E}\left[(M(G_n))^2 \mid m < \frac{1}{4}\binom{n}{2} \right] + \\ &\quad \Pr\left[m \geq \frac{1}{4}\binom{n}{2} \right] \mathrm{E}\left[(M(G_n))^2 \mid m \geq \frac{1}{4}\binom{n}{2} \right] \end{aligned}$$

By Chernoff's bound, we have

$$\Pr\left[m < \frac{1}{4}\binom{n}{2}\right] < \exp\left(-\frac{1}{16}\binom{n}{2}\right)$$

So for the numerator we have

$$\begin{aligned}
&\mathrm{E}[M(G_n)^2] \\
&< 2\Pr\left[m \geq \frac{1}{4}\binom{n}{2}\right] \mathrm{E}\left[(M(G_n))^2 \mid m \geq \frac{1}{4}\binom{n}{2}\right] \\
&\leq 2\,\mathrm{E}\left[(M(G_n))^2 \mid m \geq \frac{1}{4}\binom{n}{2}\right] \\
&= 2\sum_{j=\frac{1}{4}\binom{n}{2}}^{\binom{n}{2}} \mathrm{E}[(M(G_n))^2 \mid m = j]\,\Pr[m = j]
\end{aligned}$$

Substituting for the probability of having j 1's and using Theorem 4.3 for the values of $\mathrm{E}[(M(G_n))^2]$ and $\mathrm{E}[M(G_n)]^2$. Let p_j denote $j/\binom{n}{2} = 2j/(n^2-n)$, we obtain

$$\begin{aligned}
&\frac{\mathrm{E}[(M(G_n))^2]}{\mathrm{E}[M(G_n)]^2} \\
&\leq \frac{2^{n+1}}{((n-1)!!)^2}\sum_{j=\frac{n^2}{8}}^{\binom{n}{2}}((n-1)!!)^2\left(\frac{2j}{n^2-n}\right)^n \cdot \\
&\quad \exp\left(\underbrace{\frac{2(p_j-1)}{4p_j} + \mathrm{O}\left(\frac{(1-p_j)n^3}{m^2}\right)}_{\leq c'}\right)\binom{\binom{n}{2}}{j}2^{-\binom{n}{2}} \\
&< \frac{2^{2n+1}\exp(c')}{(n^2-n)^n}\sum_{j=0}^{\binom{n}{2}} j^n\binom{\binom{n}{2}}{j}2^{-\binom{n}{2}}
\end{aligned}$$

However from Lemma 4.2, we know that $\sum_{j=0}^{\binom{n}{2}}\binom{\binom{n}{2}}{j}j^n = \mathrm{O}\left(\left(\frac{n^2-n}{4}\right)^n 2^{\binom{n}{2}}\right)$. Substituting this result we finish the proof of the lemma. □

We are now prepared to establish the dependence of $R(h)$ on $\hat{R}(h)$. As mentioned earlier $R(h)$ and $\hat{R}(h)$ differ only at the branching points.

THEOREM 4.5. *Let $R(h)$ and $\hat{R}(h)$ be the functions defined above with c an upper bound on $\frac{\mathrm{E}_{\mathcal{G}}[M(G)^2]}{\mathrm{E}_{\mathcal{G}}[M(G)]^2}$. Then*

$$R(h) \leq \begin{cases} \frac{\hat{R}(h)}{K} + \frac{(K-1)c}{K} & \text{if } h = \text{ branching point} \\ \hat{R}(h) & \text{otherwise} \end{cases}.$$

Proof. At all levels other than the branching levels, we have $K = 1$ implying $R(h)$=$\hat{R}(h)$. However, at the branching levels we have:

$$\begin{aligned}
R(h) &= \frac{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[X_h^2]]}{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[X_h]]^2} \\
&= \frac{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[(\frac{1}{K}\sum_{i=1}^K \hat{X}_h(i))^2]]}{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[\frac{1}{K}\sum_{i=1}^K \hat{X}_h(i)]]^2}
\end{aligned}$$

Furthermore since the outcomes of the successive trials $\hat{X}_h(i)$ are independent and identically distributed

$$\mathrm{E}_\sigma[(\frac{1}{K}\sum_{i=1}^K \hat{X}_h(i))^2] = \frac{\mathrm{E}_\sigma[\hat{X}_h^2] + (K-1)\mathrm{E}_\sigma[\hat{X}_h]^2}{K}$$

Using this for the numerator and noting that the denominator is just $\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[X_h]]^2$ we get

$$\begin{aligned}
R(h) &= \frac{\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[\hat{X}_h^2]]}{K\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[\hat{X}_h]]^2} + \frac{(K-1)\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[\hat{X}_h]^2]}{K\mathrm{E}_{\mathcal{G}}[\mathrm{E}_\sigma[\hat{X}_h]]^2} \\
&= \frac{\hat{R}(h)}{K} + \frac{(K-1)\mathrm{E}_{\mathcal{G}}[\mathrm{M}(G_h)^2]}{K\mathrm{E}_{\mathcal{G}}[\mathrm{M}(G_h)]^2}
\end{aligned}$$

□

In the following theorem we show that both $R(h)$ and $\hat{R}(h)$ are bound by a constant implying that the critical ratio is $\mathrm{O}(\omega)$ from Lemma 4.1.

THEOREM 4.6. *Let $\hat{R}(h)$ and $R(h)$ be the functions defined above. Then for REP with a branching factor of 3 and branching points at powers of 2, there exists a $c \geq 1$ such that*

$$\hat{R}(1) \leq 2c, \quad \hat{R}(h) \leq \frac{4\,c\,h}{2^{\lfloor \lg(h-1)\rfloor+1}+1} \quad \text{for } h > 1$$

and

$$R(h) \leq \begin{cases} 2c & \text{if } h \text{ is a branching point or } h = 1 \\ \frac{4\,c\,h}{2^{\lfloor \lg(h-1)\rfloor+1}+1} & \text{otherwise} \end{cases}$$

Proof. We use induction on h. We know that $c \geq 1$. For $h = 1$, $\hat{R}(1) = 2 \leq 2c$ and $R(1) = 2 \leq 2c$. Assuming the statement is true for h, we prove it for $h+1$. From Theorem 4.4 we get

$$\hat{R}(h+1) \leq \frac{2h+2}{2^{\lfloor \lg h\rfloor+1}+1}R(2^{\lfloor \lg h\rfloor}) \leq \frac{4c(h+1)}{2^{\lfloor \lg h\rfloor+1}+1}$$

(where $R(2^{\lfloor \lg h\rfloor}) \leq 2c$ is by induction hypothesis). For $R(h+1)$, there are two cases:
Case 1: $h+1$ is a branching point. From Theorem 4.5, we get $R(h+1) \leq 2c$.
Case 2: $h+1$ is not a branching point. From Theorem 4.5 we get $R(h+1) = \hat{R}(h+1)$. □

THEOREM 4.7. *Let $\omega = \omega(n)$ be any function satisfying $\omega \to \infty$ as $n \to \infty$. Then for almost all graphs G, we have,*

$$\frac{\mathrm{E}_\sigma[X_G^2]}{\mathrm{E}_\sigma[X_G]^2} \leq \mathrm{O}(\omega)$$

Proof. The factor of $\frac{2h}{2^{\lfloor \lg(h-1) \rfloor + 1} + 1} \leq 2$. Hence, both $R(h)$ and $\hat{R}(h)$ are O(1) (Theorem 4.6). Using Lemma 4.1 we bound the critical factor by $\mathrm{O}(\omega)$. □

Each call to REP with a branching factor of 3 and branching points at powers of 2 can be performed using $\mathrm{O}(n^2)$ operations. Furthermore, to obtain an fpras for random graphs, it is sufficient to repeat REP $\Omega(\omega)$ times. Thus, we obtain a total running time of $\mathrm{O}(n^2\omega)$ for almost all graphs. This is the fastest known algorithm for approximating $M(G)$ for random graphs. To do better than $\mathrm{O}(n^2)$, one could think of an estimator that inspects only a fraction of edges for a given vertex. However, on the flip side, such a sublinear estimator would have a much higher variance.

We can also use REP to deal with the more general random graph model $\mathcal{G}(n, p)$. We branch by a factor of 2 (i.e, $K = 2$) and investigate the choice of Branching points according to principles established at the start of the section. These parameters for REP are summarized in Table 2.

5 Experimental Analysis for GREEDY

We investigate the performance of GREEDY against Chien's estimator on some instances of commonly used graphs where counting the number of perfect matchings is interesting, and also on some random graphs. We restricted our test cases to inputs (mostly bipartite) where it is possible to accurately count the number of perfect matchings. Not surprisingly, GREEDY not only runs faster, but also produces more accurate results every time. Random graphs were generated as in the DIMACS implementation challenge. The final test case was a complete graph with a perfect matching removed (i.e., deranged matchings of n people with partners (of either sex) other than their spouse [14]). The results (in scientific notation) are summarized in Table 3 and Figure 8. They are based on 1000 runs of both Estimators.

6 Concluding Remarks

Table 4 summarizes the main properties of our algorithms for non-bipartite matchings. The worst case performance of GREEDY is an open problem. We conjecture that GREEDY is always good and provide experimental results to support our claim. We have shown that REP provides the fastest estimator for counting perfect matchings in random graphs and also envisage such a scheme to be part of a general framework which can be used to solve similar combinatorial problems.

References

[1] B. Bollobás, *Random graphs*, Academic Press, London, England, 1985.

[2] Steve Chien, *A determinant-based algorithm for counting perfect matchings in a general graph*, Proceedings of the fifteenth annual ACM-SIAM Symposium On Discrete Algorithms (2004), 728–735.

[3] J. Edmonds, *Paths, trees, and flowers*, Canadian Journal of Mathematics **17** (1965), 449–467.

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

[5] Martin Fürer and Shiva Prasad Kasiviswanathan, *An almost linear time approximation algorithm for permanent of a random (0-1) matrix*, In Proceedings of FSTTCS 2004, Spriger Verlag, LNCS, vol 3328, 2004, 263–275.

[6] S. Janson, T. Luczak, and A. Ruciński, *Random graphs*, Wiley Interscience, 2000.

[7] Svante Janson, *The numbers of spanning trees, Hamilton cycles and perfect matchings in a random graph*, Combinatorics, Probability and Computing (2001), 97–126.

[8] M. Jerrum, A. Sinclair, and E. Vigoda, *A polynomial time approximation algorithm for the permanent of a matrix with non-negative entries*, Journal of the ACM **51** (2004), no. 4.

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

[10] Donald E. Knuth, *Estimating the efficiency of backtrack programs*, Mathematics of Computation **29** (1974), 121–136.

[11] Victor De la Pena and Evarist Giné, *Decoupling, from dependence to independence*, Springer Verlag, New York, 1999.

[12] Paul W. Purdom, *Tree size by partial backtracking*, SIAM Journal on Computing **7** (1978), no. 4, 481–491.

[13] L. Rasmussen, *Approximating the permanent:A simple approach*, Random Structures and Algorithms **5** (1994), 349–361.

[14] N. J. A Sloane, *The on-line encyclopedia of integer sequences*, published electronically at http://www.research.att.com/njas/sequences/, 1996-2004.

[15] Clifford Stein and David R. Karger, *A new approach to the minimum cut problem*, Journal of the ACM **43** (1996), no. 4, 601–640.

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

Probability	*B.P. Selector s*	*Single Run*	*Critical Ratio*	*Total Running Time*
p $> \frac{1}{5}$	$\sqrt{2} < s < 2^{\frac{1}{(p^{-1}-1)}}$	$O(n^2)$	$O(\omega(n))$	$O(n^2\omega(n))$
p $= \frac{1}{5}$	$\sqrt{2}$	$O(n^2 \lg n)$	$O(\lg(n)\omega(n))$	$O(n^2 \lg^2(n)\omega(n))$
p $< \frac{1}{5}$	$2^{\frac{1}{(p^{-1}-1)}} < s < \sqrt{2}$	$O(n^{\frac{1}{\lg s}})$	$O(n^{\frac{1}{p}-1-\frac{1}{\lg s}}\omega(n))$	$O(n^{\frac{1}{p}-1}\omega(n))$

Table 2: Performance of REP for diffferent probabilities

Graph Type	*Correct Result*	*Estimate of GREEDY*	*Estimate of Chien's*
6×6 Square Grid Graph	6.728e+03	6.533e+03	7.310e+03
8×8 Square Grid Graph	1.298e+07	1.270e+07	1.738e+07
10×10 Square Grid Graph	2.586e+11	2.659e+11	1.009e+11
20×3 Rectangular Grid Graph	4.134e+05	4.136e+05	4.222e+05
5D-Hypercube	5.891e+05	5.845e+05	6.659e+05
20+20 Random Bipartite Graph with 100 Edges	6.95e+05	7.191e+05	6.557e+05
20+20 Random Bipartite Graph with 200 Edges	1.871e+12	1.862e+12	1.606e+12
20+20 Random Bipartite Graph with 300 Edges	5.967e+15	5.736e+15	7.693e+15
Complete graph(n=100) with 1 Matching Removed	1.644e+78	1.644e+78	1.530e+78

Table 3: Performance of GREEDY on various graphs

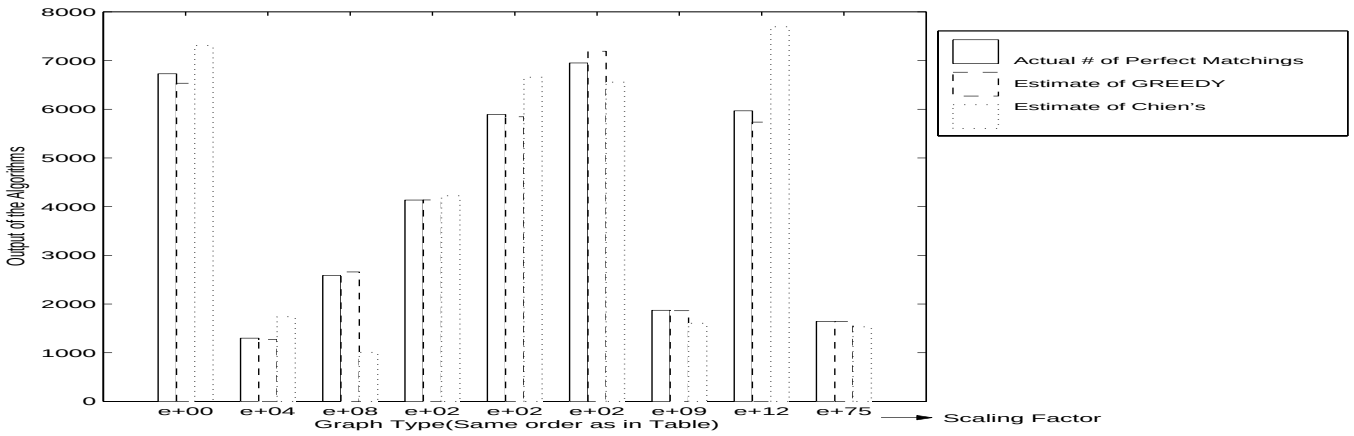

Figure 8: Performance of the Estimators

Algorithm	*Positive Points*	*Negative Points*
SIMPLE	Can't be easier	High worst case bound
GREEDY	Looks good on many graphs	Difficult to analyze
REP	Fastest Estimator for random graphs	High worst case bound

Table 4: Properties of different estimators presented