Combinatorics, Probability and Computing (1994) 3, 273–283

A Randomised Approximation Algorithm for Counting the Number of Forests in Dense Graphs

J. D. ANNAN†

University College, Oxford
e-mail: annan@vax.ox.ac.uk

Received 18 August 1993; revised 21 January 1994

For Paul Erdős on his 80th birthday

A polynomial-time randomised algorithm for uniformly generating forests in a dense graph is presented. Using this, a fully polynomial randomised approximation scheme (fpras) for counting the number of forests in a dense graph is created.

1. Introduction and definitions

The number of *forests* (acyclic subgraphs) of a graph is an example of a *Tutte invariant*. That is, it can easily be calculated from the Tutte polynomial of the graph.

The Tutte polynomial of a graph $G = (V, E)$, was introduced as a generalisation of the chromatic polynomial in [14]. It is defined as

$$T(G; x, y) = \sum_{A \subseteq E} (x-1)^{r(E)-r(A)} (y-1)^{|A|-r(A)}, \tag{1}$$

where the sum is over all subsets of the edge set of G, and the *rank* of a set of edges, $r(A)$ is defined by

$$r(A) = |V| - k(A), \tag{2}$$

where $k(A)$ is the number of connected components of the graph with vertex set V and edge set A (including isolated vertices).

It can immediately be seen from the above that the number of forests of a graph is equal to the value of the Tutte polynomial at the point $(2, 1)$. The Tutte polynomial contains many other invariants of fundamental importance in fields as diverse as statistical physics, knot theory and graph colourings.

For example, the partition function of the q-state Potts model of statistical physics can be calculated from an evaluation of the Tutte polynomial along the hyperbola

$$H_q = \{(x, y) : (x-1)(y-1) = q\},$$

† Supported by SERC

where $q \geq 2$ is any positive integer. The hyperbola H_2 gives the partition function of the Ising model.

For planar G, the Jones polynomial of the alternating link associated with G is given by evaluation of the Tutte polynomial along the hyperbola $xy = 1$, and the chromatic polynomial is given by evaluation along the y-axis. A host of other interpretations, and further explanation of these interpretations, can be found in [16].

Vertigan and Welsh proved in [15] that calculating any partial evaluation of the Tutte polynomial is **#P**-hard even for planar bipartite graphs, with a few exceptions (for instance, counting the number of spanning trees is in **P**). The point $(2, 1)$ is not one of these exceptions, so we cannot expect to find a polynomial-time algorithm to calculate the number of forests.

However, if the associated existence problem is easy, there is no *a priori* reason why we should not be able to evaluate approximately a **#P**-hard function using randomised algorithms, and much attention has focussed in this direction over recent years.

The following definition [11] is the standard model of an efficient approximation algorithm.

Definition 1. A *fully polynomial randomised approximation scheme (fpras)* for a function f is an algorithm $\mathscr{A}$ that, on input (x, ϵ), with probability greater than $3/4$, produces an output $\mathscr{A}(x, \epsilon)$ satisfying

$$(1+\epsilon)^{-1} \leq \frac{\mathscr{A}(x,\epsilon)}{f(x)} \leq 1+\epsilon,$$

and that runs in time polynomial in $|x|$ and ϵ^{-1}.

Jerrum, Valiant and Vazirani [10] show that, for a large class of problems (ones that are *self-reducible* [12]), approximate counting of solutions and almost uniform generation of solutions are polynomially equivalent. A *uniform generator* is a randomised algorithm that, given an instance of a problem in **NP**, generates uniformly at random one of the 'witnesses' for that instance, and produces some output at least half the time. *Almost uniform generation* is a slightly weaker condition, only requiring that the output distribution is 'close to' uniform (in a well-defined sense).

Using this idea, Jerrum and Sinclair [8] present a fpras for computing the permanent of dense matrices, whereas exact evaluation for these matrices was shown to be **#P**-hard by Broder [2]. Jerrum and Sinclair again [9] present a fpras for the ferromagnetic Ising partition function of a graph, while showing that no fpras exists for the antiferromagnetic Ising partition function unless **NP** = **RP**. The ferromagnetic case corresponds to the branch of the hyperbola H_2 in which x and y are both greater than 1, and the antiferromagnetic case to the part of the other branch of that hyperbola in which $0 \leq y < 1$.

Perhaps less well known are the results of Edwards [5] concerning 3-colourings of dense graphs. We say a simple graph G is *α-dense* (or *has density α*) if every vertex in G has degree at least $\alpha |V(G)|$. Edwards shows that if $\alpha > 1/2$, the existence problem for α-dense graphs

is in **P**, and also that, in this case, the number of 3-colourings can be counted exactly in polynomial time. For $\alpha < 1/2$, the existence problem remains **NP**-complete, so we cannot hope even to approximate the number of 3-colourings. The number of k-colourings of a graph can easily be calculated from the Tutte polynomial evaluated at the point $(1-k, 0)$ and the results for $k = 3$ generalise to any integer $k > 3$, with the threshold between easiness of counting and hardness of approximation being $(k-2)/(k-1)$ in general.

These results suggest a possible frontier between the points in the Tutte plane at which approximation is possible, and those at which it is not. In [16], Welsh makes the following conjecture.

Conjecture 1. *There exists a fpras for computing the Tutte polynomial at each (rational) point* (x, y) *of the positive quadrant* $x \geq 1$, $y \geq 1$.

The number of forests is a point on the boundary of this region, which also includes the ferromagnetic case of the general q-state Potts model.

In this paper, we present an algorithm for approximately counting the number of forests in a dense graph, and, furthermore, generalise this to show the existence of a fpras for the Tutte polynomial of dense graphs at any rational point on the line $x \geq 1$, $y = 1$. A routine but lengthy reduction shows that the number of forests of a dense graph cannot be *exactly* counted in polynomial time, unless **NP** = **RP** [1].

2. Random generation of forests

We use $\mathscr{G}_\alpha$ to denote the set of all simple α-dense graphs, and $n = |V(G)|$ to denote the number of vertices of G. Given a graph $G \in \mathscr{G}_\alpha$ on vertices $u_1, \ldots, u_n$, we form a new graph G^+ by adding a new vertex v to the vertex set of G, and adding an edge connecting each vertex of G to v. We call this the *join* of G and v.

It is well known that there are fast (polynomial-time) algorithms for generating spanning trees of any graph uniformly at random. We are interested in the degree of the vertex v in T_R^+, the random spanning tree of G^+. The following lemma and corollary will be used.

Lemma 1. *The probability that the edge* vu_i *is contained in a randomly chosen spanning tree of* G^+ *is at most* $2/(\alpha n + 2)$ *for* $1 \leq i \leq n$.

Corollary 1. *The probability that the degree of* v *in* T_R^+ *is greater than* $4/\alpha$ *is less than* $1/2$.

Proof of Lemma 1. We use the theory of electrical networks to bound the probability that a particular edge vu_i is in T_R^+. We form an electrical network from G^+ by replacing every edge by a resistor of resistance 1 Ω. The following two propositions are needed.

Proposition 1. *If the resistance of any resistor of a network is increased, the effective resistance between any two points cannot decrease. Similarly, if any resistance is decreased, the effective resistance cannot increase.*

Proposition 2. *The probability that the edge vu_i is contained in the random spanning tree of a graph is equal to the effective resistance between v and u_i in the associated electrical network.*

A proof of the first Proposition is given in [4] as the monotonicity law, but it dates back a long time before this (as Rayleigh's Principle), and the second Proposition, proved in [3], is due in principle to Kirchhoff.

Now we bound the resistance of G^+ between u_i and v. The vertex u_i has at least αn neighbours in G, each of which is connected to v. By Proposition 1, if we remove every resistor except for the ones connecting u_i to its neighbours, and the resistors connecting v to the neighbours of u_i in G, we can only increase the resistance between v and u_i. But we are left with at least αn disjoint paths of length 2 and a path of length 1 (the edge vu_i), giving a total resistance of $2/(\alpha n + 2)\ \Omega$, which, by Proposition 2, completes the proof of Lemma 1. □

Proof of Corollary 1. The expected degree of vertex v in T_R^+ is equal to the sum, over all edges e incident with v, of the probability that that edge is in the random tree T_R^+. By Lemma 1, we have shown that each of these probabilities is less than $2/\alpha n$. As there are only n edges incident with v, the expected degree of v is less than $2/\alpha$.

Letting d denote the degree of v in T_R^+, we can write:

$$2/\alpha > E(d) \geq E(d \mid d > 4/\alpha) \cdot P(d > 4/\alpha) \geq 4/\alpha \cdot P(d > 4/\alpha),$$

which gives the required result. □

3. An algorithm for the uniform generation of forests

We now present an algorithm for the uniform generation of forests.

A spanning tree T of G^+ induces, in the obvious way, a forest F of G, given by $F = T \backslash v$. This mapping is clearly surjective, but not in general injective. The following lemma tells us how many spanning trees there are in G^+ that induce a particular forest F of G.

For any forest F with connected components $F_1, \ldots, F_c$, we say that the *size* of F_i is $|V(F_i)|$.

Lemma 2. *The number of spanning trees of G^+ that induce a given forest of G is equal to the product of the sizes of the connected components of the forest.*

Proof. Consider an arbitrary component of the forest, of size s say. Exactly one of its vertices must be directly connected to v in any tree that induces the forest. As there is one edge from each vertex in G to v, there are exactly s ways in which this can occur, independently of how the other components are connected to v. □

```
(1)    G+ := join of G and v
(2)    for c := 1 to n^{4/α} do begin
(3)        T := SpT(G+)
(4)        F := T\v
(5)        if R⟨T, F⟩ then output F
(6)        end
```

Figure 1 The uniform forest generator, FGEN_α

We now define a binary relation.

Definition 2. For a spanning tree T of G^+ and forest F of G, we say T *is related to* F (denoted $\mathbf{R}\langle T, F\rangle$) if the following two conditions hold:

1 $T\backslash v = F$

2 In T, v is connected to the vertex of lexicographically lowest index in each component of F.

It is easy to see that, for any forest F of G, there is exactly one spanning tree T of G^+ such that $\mathbf{R}\langle T, F\rangle$.

The algorithm for the uniform generation of forests in G is given in Figure 1. We assume that the subroutine $\mathrm{SpT}(H)$ returns a uniformly generated random spanning tree of any graph H. This can certainly be achieved in polynomial time using a randomised Turing machine.

If the algorithm produces more than one forest, we simply take the first one that is output, and discard the others.

Theorem 1. *The following statements are true.*

1 *FGEN_α generates each forest of G equiprobably.*

2 *For any $G \in \mathscr{G}_\alpha$, FGEN_α outputs a forest with probability greater than* $1/2$.

3 *It runs in polynomial time.*

Proof. Clearly, the algorithm runs in polynomial time. To prove the first statement, consider a single repetition of the **do** loop. For every forest F of G, there is exactly one spanning tree T of G^+ such that $\mathbf{R}\langle T, F\rangle$. Each tree in G^+ is picked with equal probability, so each forest in G is output equiprobably.

For the second statement, from Corollary 1 we know that with probability greater than $1/2$, the degree of v in the randomly chosen tree T is no greater than $4/\alpha$. In this case, there are at most $(\alpha n/4)^{4/\alpha}$ spanning trees in G^+ that induce the forest $T\backslash v$. In one repetition of the loop, the probability that T is related to F, and so F is output in line (5), is therefore at least $1/2 \cdot 1/(\alpha n/4)^{4/\alpha} > n^{-4/\alpha}$. Hence the probability that the algorithm fails to produce a forest in every iteration of the loop is bounded above by $(1 - 1/n^{4/\alpha})^{n^{4/\alpha}} < e^{-1} < 1/2$. □

4. Approximating the number of forests

We now combine the above with the idea of self-reducibility, and a modified version of the algorithm of Jerrum, Valiant and Vazirani [10] to construct a polynomial-time algorithm to approximate the number of forests of $G \in \mathcal{G}_\alpha$. We write $f(G)$ for the number of forests of any graph G.

First, we need a preliminary lemma.

Lemma 3. *The number of forests of the complete graph K_n can be calculated in polynomial time.*

Proof. We create a recurrence relation for the Tutte polynomial of the complete graph evaluated at the point (2, 1). For any integers $r, m \geq 1$ we construct the graph $H_{r,m}$, which consists of a clique on m vertices $\{u_1, \ldots, u_m\}$ and another vertex v, which is connected to each of the u_i by r edges in parallel. By considering all possible subsets of the rm edges adjoining v that a forest of $H_{r,m}$ can contain, and writing $F_{r,m}$ for the number of forests of $H_{r,m}$, it is not hard to show that

$$F_{r,m} = \sum_{i=1}^{m-1} \binom{m}{i} r^i F_{i,m-i} + F_{1,m-1} + 1, \quad m \geq 2$$

and

$$F_{r,1} = r + 1.$$

Note that $F_{1,n-1} = f(K_n)$.

It is clear that to evaluate $F_{1,n-1}$, we only need to evaluate the $F_{r,m}$ for which $r+m \leq n-1$. So we must use the recurrence relation $O(n^2)$ times in all, each time summing $O(n)$ terms. This is certainly a polynomial-time calculation, the exact time taken depending on the model of computation used. □

We can express the number of forests of a graph G as follows:

$$f(G) = f(K_n) \cdot \frac{f(K_n \backslash e_1)}{f(K_n)} \cdot \frac{f(K_n \backslash e_1, e_2)}{f(K_n \backslash e_1)} \cdots \frac{f(G)}{f(G \cup e_l)},$$

where the edges $e_1, \ldots, e_l$ that are successively deleted to create the sequence of graphs are exactly the edges in $K_n \backslash G$ (in any order).

For any dense graph $G \in \mathcal{G}_\alpha$ and edge $e \notin G$, we approximate the fraction $f(G)/f(G \cup e)$ by the simple method of generating forests in $G \cup e$ uniformly at random, and counting the proportion of these that are in fact contained in G. So we approximate independently each term in the above product, and if each approximation is accurate enough, their product will approximate the number of forests in G within ratio $1+\epsilon$. The algorithm for approximating the number of forests of a dense graph is given in Figure 2.

Theorem 2. *The algorithm $APPROX_\alpha(G, \epsilon)$ is a fpras for the number of forests of any graph $G \in \mathcal{G}_\alpha$.*

(1) $H := K_n$
(2) $\Pi := f(H)$
(3) $t := 180n^6/\epsilon^2$
(4) **while** $H \neq G$ **do begin**
(5) Let e be any edge in H but not in G
(6) make $3t$ calls to $\mathrm{FGEN}_\alpha(H)$
(7) **if** at least t of the trials yield an output **then**
(8) let $S = \{y_1, \ldots, y_t\}$ be the first t outputs of $\mathrm{FGEN}_\alpha(H)$
(9) **else halt**
(10) $\beta := |\{y \in S : e \notin y\}|/t$
(11) $H := H \backslash e$
(12) $\Pi := \Pi \times \beta$
(13) **endwhile**
(14) **output** Π

Figure 2 The fpras for forests, $\mathrm{APPROX}_\alpha(G, \epsilon)$

Proof. We have to show that two things happen simultaneously with probability at least 3/4:

1 The algorithm produces an output.
2 This output approximates the number of forests of G within the required ratio $1 + \epsilon$.

Clearly the algorithm runs in polynomial time: the main loop is repeated n^2 times at most.

We know that FGEN fails to produce a forest with probability at most 1/2. Consider the $3t$ calls to FGEN in one loop of the algorithm, and let the random variable F be the number of times that no output is given by FGEN. Clearly $E(F) \leq 3t/2$ and $Var(F) \leq 3t/4$, and hence, using Chebyshev's inequality, we have $P(F > 2t) < 3/t$. Since t is certainly bounded below by $180n^2$, the probability that some output is produced is greater than $(1 - 1/60n^2)^{n^2}$, which is easily shown to be greater than 59/60.

Now we claim that on every iteration of the loop, with high probability, β will approximate $f(H\backslash e)/f(H)$ within ratio $1 + \epsilon/2n^2$. Consider the random variable $X = |\{y \in S : e \notin y\}|/t$. Let $\mu = E(X)$. Since X is the average of t independent $(0, 1)$-valued random variables, $Var(X) \leq 1/t$. Using Chebyshev's inequality again, we see that

$$P\left(|X - \mu| \leq \epsilon/6n^2\right) > 1 - 36n^4/\epsilon^2 t \geq 1 - 1/5n^2.$$

Now μ is never less than 1/2: to see this, consider the set of all forests of $G \bigcup e$. These forests can be partitioned into two sets: the forests that include e, and the ones that do not. The forests that do not include e are precisely the forests of G. It is obvious that the number of forests including e does not exceed the number without e: the natural map

from forests that contain e to those that do not (given by deleting e from the edge-set of the forest) is clearly 1-1. This allows us to bound the *relative* error of β, as we now have

$$P\left(\left|\frac{\beta}{\mu} - 1\right| \leq \frac{\epsilon}{3n^2}\right) \geq 1 - \frac{1}{5n^2}.$$

Since $\epsilon/n^2 \leq 1$, it is easy to see that $(1+\epsilon/2n^2)^{-1} \leq (1-\epsilon/3n^2)$. So, with probability at least $1 - 1/5n^2$, β approximates μ within ratio $1+\epsilon/2n^2$.

The probability of β being such an accurate approximation in every iteration is at least $(1 - 1/5n^2)^{n^2}$, which is greater than 4/5. If β is this accurate in each iteration, then the final output Π approximates $f(G)$ within ratio $(1+\epsilon/2n^2)^{n^2}$, which is no worse than $1+\epsilon$. So the probability of the algorithm producing an output *and* the output being sufficiently accurate is at least $(59/60) \cdot (4/5) > 3/4$, as required. □

5. Further results

We now use the ideas of Jaeger *et al.* [7] to generalise our results a little. In this section, we show how the Tutte polynomial can be approximated at any rational point on the line $x \geq 1$, $y = 1$.

In order to do this, we must first explain how multiple edges can be handled.

Our original algorithm will not work, since a graph with multiple edges is not contained within K_n. However, it will be contained within the k-thickening of K_n (the graph obtained by replacing each edge of the complete graph with k edges in parallel), where k is the greatest multiplicity of any edge of the graph. We can simply modify our algorithm to begin with the k-thickening of K_n (which we denote by K_n^k), and again remove edges one at a time until G is reached, each time approximating the ratio of the number of forests in successive graphs so formed. The extra edges will clearly not hinder the running of the algorithm.

We will also need to calculate the number of forests of the k-thickened complete graph. Equation 4.1 in [7] tells us how to do this: it simplifies to

$$T(K_n^k; 2, 1) = (2^k - 1)T\left(K_n; 1 + \frac{1}{k}, 1\right),$$

and, using similar ideas to those used earlier, we can construct the following recurence relation for the Tutte polynomial of the graph $H_{r,m}$ (defined above) evaluated at the point $(x, 1)$:

$$T(H_{r,m}; x, 1) = \sum_{i=1}^{m-1} \binom{m}{i} r^i T(H_{i,m-i}; x, 1) + (x-1)T(H_{1,m-1}; x, 1) + 1, \ m \geq 2 \qquad (3)$$

$$T(H_{r,1}; x, 1) = x + r - 1.$$

Hence we can create a fpras for the number of forests of dense graphs with multiple edges. The running time will depend polynomially on the greatest multiplicity of an edge.

We can now approximate the Tutte polynomial at other points along the line $y = 1$.

(1) $G^{(x-1)+} :=$ $(x-1)$-join of G and v
(2) **for** $c := 1$ to $n^{2x/\alpha}$ **do begin**
(3) $T := \mathrm{SpT}(G^{(x-1)+})$
(4) $F := T \backslash v$
(5) **if** $\mathbf{R}\langle T, F\rangle$ **then output** F
(6) **end**

Figure 3 The forest generator, $\mathrm{FGEN}_\alpha^{x-1}$

Theorem 3. *For any rational constant $x \geq 1$, there exists a fpras for evaluating the Tutte polynomial of dense graphs at the point $(x, 1)$.*

The running time of our algorithm may not be bounded by a polynomial function of x and the other inputs, so x cannot be input as a variable.

Proof. First we prove the theorem for positive integer values of x. It has already been demonstrated for $x = 1$ and $x = 2$. For $x \geq 2$, we take our input graph G and form a new graph by adding a new vertex v and connecting it to every vertex in G with a bundle of $x - 1$ edges in parallel. We call the new graph thus obtained the $(x-1)$-join of G and v, and denote it by $G^{(x-1)+}$. Note that $G^{(1)+}$ is just the same as G^+.

We now describe an algorithm $\mathrm{FGEN}_\alpha^{x-1}$, which outputs a randomly selected forest of G according to a biased distribution. This will enable us to approximate the Tutte polynomial of G evaluated at the point $(x, 1)$ in a similar manner to our approximation for the number of forests.

The new algorithm $\mathrm{FGEN}_\alpha^{x-1}$ is given in Figure 3. We now consider the probability of a particular forest F being output in a single iteration of the loop. If we write $k(F)$ for the number of connected components of a graph, *counting isolated vertices as connected components*, then there are exactly $(x-1)^{k(F)}$ spanning trees in $G^{(x-1)+}$ that are related to a forest F (using the binary relation $\mathbf{R}$ that was previously defined), as, for each component of F, there are $x - 1$ possible edges in $G^{(x-1)+}$ connecting the vertex of smallest index in this component to v. So the probability of a particular forest being generated in a single iteration of the main loop is proportional to $(x-1)^{k(F)}$. It is straightforward to show, in a similar manner to the method used for Corollary 1, that for at least half of the spanning trees of $G^{(x-1)+}$, the degree of the distinguished vertex v is less than $2x/\alpha$. So, as for FGEN_α, with probability greater than $1/2$, this algorithm produces some output. The running time of this algorithm is where the exponential dependence on x arises: this is necessary in order to ensure that some output is produced (with a high enough probability).

We now look again at the equation for the Tutte polynomial of a graph that we gave in the introduction (1):

$$T(G) = \sum_{A \subseteq E} (x-1)^{r(E)-r(A)} (y-1)^{|A|-r(A)}.$$

At the point $(x, 1)$, the summand is non-zero only when $|A| = r(A)$, that is, when A is the edge set of a forest of G. So we can rewrite this expression as

$$T(G) = (x-1)^{r(E)} \sum_{A} (x-1)^{-r(A)}, \tag{4}$$

where the summation is now over the edge sets A of all forests of G. Substituting the definition of rank (2) into this gives

$$T(G; x, 1) = (x-1)^{-1} \sum_{A} (x-1)^{k(A)}. \tag{5}$$

So the Tutte polynomial of G evaluated at $(x, 1)$ is just $1/(x-1)$ times the weighted sum over all the forests of G, with the weight of a forest F equal to $(x-1)^{k(F)}$. This is precisely the weighting given to the forests using the random generation algorithm $\mathrm{FGEN}_{\alpha}^{x-1}$.

Writing $f^{x-1}(G)$ in place of $T(G; x, 1)$, we have

$$f^{x-1}(G) = f^{x-1}(K_n) \cdot \frac{f^{x-1}(K_n \backslash e_1)}{f^{x-1}(K_n)} \cdots \frac{f^{x-1}(G)}{f^{x-1}(G \cup e_l)}, \tag{6}$$

where the edges $e_1, \ldots, e_l$ are precisely the edges contained in $K_n \backslash G$.

The recurrence relation given above (3) can be used to calculate the Tutte polynomial of the complete graph at the point $(x, 1)$ in polynomial time.

We can again approximate each fraction in (6) by generating forests in the larger graph according to the non-uniform distribution of $\mathrm{FGEN}_{\alpha}^{x-1}$ described above, and seeing how many of them are contained in the smaller graph. The algorithm for APPROX_{α} will do this if we change all the references to FGEN_{α} in it to refer to $\mathrm{FGEN}_{\alpha}^{x-1}$ instead. The proof that this new algorithm works is identical to the proof for the previous counting algorithm.

Hence we have a fpras for the Tutte polynomial of dense graphs at the point $(x, 1)$ for any positive integer $x \geq 1$.

We now consider rational $x = 1 + a/b$, where a and b are positive coprime integers.

It has just been shown that the Tutte polynomial of the b-thickening of G can be approximated at the point $(1 + a, 1)$ and Equation 4.1 from [7] again tells us that

$$T(G^b; 1 + a, 1) = bT\left(G; \frac{a+b}{b}, 1\right),$$

which completes the proof. □

6. Conclusions

More support for Welsh's conjecture (1) has recently been provided by Welsh himself [17]. First, he proved that there exists a fpras to evaluate the Tutte polynomial of dense graphs at any point (x, y) in the area defined by

$$\{(x, y) : x \geq 1, \ y > 1, \text{ and } (x-1)(y-1) \leq 1\}.$$

Further work of Welsh with Alan Frieze and Noga Alon has since led to the elimination of the constraint $(x-1)(y-1) \leq 1$ in the above. This result, in conjunction with the

results contained above, shows that Welsh's conjecture is indeed true when restricted to dense graphs.

However, it does not appear to be possible to extend the methods used to include all graphs.

There is no obvious reason why the Tutte polynomial should not be approximable in the entire positive quadrant $(x, y) : x, y \geq 0$. This region includes the point $(2, 0)$ at which point the Tutte polynomial counts the number of acyclic orientations of a graph. We therefore pose the following question:

Does there exist a fpras for the number of acyclic orientations of a (dense) graph?

7. Acknowledgements

This problem was suggested to me by Dominic Welsh. I am also grateful to him for his help and advice in writing this paper.

References

[1] Annan, J. D. (1994) *The complexity of counting problems,* D. Phil. Thesis (in preparation), University of Oxford.

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

[3] Brooks, R. L., Smith, C. A. B., Stone, A. H. and Tutte, W. H. (1940) The dissection of rectangles into squares. *Duke Mathematics Journal* **7** 312–340.

[4] Doyle, P. G. and Snell, J. L. (1984) Random walks and electrical networks. *Carus Mathematical Monographs* **22.**

[5] Edwards, K. (1986) The complexity of colouring problems on dense graphs. *Theoretical Computer Science* **43** 337–343.

[6] Gill, J. (1977) Computational complexity of probabilistic Turing machines. *SIAM Journal of Computing* **6** 675–695.

[7] Jaeger, F., Vertigan, D. L. and Welsh, D. J. A. (1990) On the computational complexity of the Jones and Tutte polynomials. *Math. Proc. Camb. Phil. Soc.* **108** 35–53.

[8] Jerrum, M. R. and Sinclair, A. J. (1989) Approximating the permanent. *SIAM Journal of Computing* **18** (6) 1149–1178.

[9] Jerrum, M. R. and Sinclair, A. J. (1990) Polynomial-time approximation algorithms for the Ising model. (Extended abstract) *Proc. 17th ICALP* 462–475.

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

[11] Karp, R. M. and Luby, M. (1983) Monte-Carlo algorithms for enumeration and reliability problems. *Proc. 24th annual IEEE Symposium on the Foundations of Computer Science* 56–64.

[12] Schnorr, C. P. (1976) Optimal algorithms for self-reducible problems. *Proc. 3rd ICALP* 322–337.

[13] Sinclair, A. J. (1993) *Algorithms for random generation and counting: a Markov chain approach,* Birkhäuser, Boston.

[14] Tutte, W. T. (1947) A ring in graph theory. *Proceedings of the Cambridge Philosophical Society* **43** 26–40.

[15] Vertigan, D. L. and Welsh, D. J. A. (1992) The computational complexity of the Tutte plane: the bipartite case. *Combinatorics, Probability and Computing* **1** 181–187.

[16] Welsh, D. J. A. (1993) Complexity: Knots, Colourings and Counting *London Math. Soc. Lecture Notes* **186,** Cambridge University Press.

[17] Welsh, D. J. A. (1994) Randomised approximation schemes in the Tutte plane. *Combinatorics, Probability and Computing* **3** 137–143.