ELSEVIER

Discrete Applied Mathematics 60 (1995) 119–129

DISCRETE APPLIED MATHEMATICS

A new approach to solving three combinatorial enumeration problems on planar graphs

Charles J. Colbourn[a], J. Scott Provan[b,*], Dirk Vertigan[c,1]

[a]*Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ont., Canada*
[b]*Department of Operations Research, University of North Carolina, CB # 3180, Chapel Hill, NC 27599, USA*
[c]*Mathematical Institute, Oxford University, Oxford, UK*

Received 20 October 1992; revised 17 October 1994

Abstract

The purpose of this paper is to show how the technique of *delta-wye graph reduction* provides an alternative method for solving three enumerative function evaluation problems on planar graphs. In particular, it is shown how to compute the number of spanning trees and perfect matchings, and how to evaluate energy in the Ising "spin glass" model of statistical mechanics. These alternative algorithms require $O(n^2)$ arithmetic operations on an n-vertex planar graph, and are relatively easy to implement.

1. Planar graphs and delta-wye transformations

A natural method to develop graph-theoretic algorithms is to develop simplifying transformations that, when applied repeatedly to a graph, reduce it in polynomially many steps to a trivial graph such as a single edge. We examine algorithms of this type here. We start with n-vertex graph $G = (V, E)$, possibly with loops and parallel edges, and consider the following six basic transformations:

- *Loop deletion*: A loop e (edge whose two endvertices are identical) can be deleted.
- *Pendant edge deletion*: A pendant edge e (edge incident with a vertex of degree 1) can be deleted along with the degree 1 vertex.
- *Parallel reduction*: Two edges e and f in parallel (each having the same two endvertices) can be replaced by a single edge e' having the same two endvertices.
- *Series reduction*: Two edges $e = (x, y)$ and $f = (y, z)$ for which y is a degree 2 vertex can be replaced by a single edge $e' = (x, z)$, with y being deleted.

*Corresponding author.
[1]Currently at the Department of Mathematics, Louisiana State University, Baton Rouge, LA 70803-4918, USA.

SSDI 0166-218X(95)E0111-1

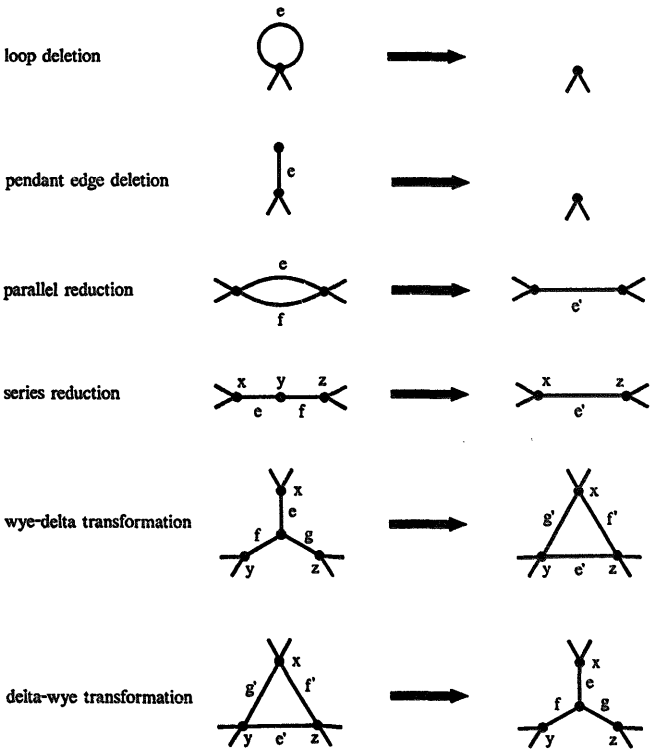

Fig. 1. The six basic transformations.

- *Wye-delta transformation*: If w is a degree 3 vertex (a *wye*) adjacent to three vertices x, y, z by edges e, f, g, respectively, vertex w and edges e, f, g can be deleted and replaced by edges $e' = (y, z)$, $f' = (x, z)$ and $g' = (x, y)$.
- *Delta-wye transformation*: If $e' = (y, z)$, $f' = (x, z)$ and $g' = (x, y)$ are edges of G (a *delta* or *triangle*), they can be deleted and replaced by adding a new vertex w adjacent to x, y, z.

The transformations are given in Fig. 1.

G is called a *$\Delta Y\Delta$-reducible graph* if it can be reduced to a single edge by repeated application of the six transformations above, in any order. At present, no efficiently

recognizable characterization of $\Delta Y\Delta$-reducible graphs is known, although such characterizations are available for several classes of graphs reducible by using certain specified subsets of these transformations [8, 22]. Lehman [17] conjectured that all planar graphs are $\Delta Y\Delta$-reducible; this was first proved by Epifanov [9]; simpler proofs and associated algorithms have been given by Truemper [21] and by Feo and Provan [10]. Feo and Provan [10], in particular, describe an easily implementable algorithm for reducing any planar graph using $O(n^2)$ transformations with constant time per transformation. (For a more detailed discussion of an implementation of this algorithm that avoids loops, pendant edge reductions, and "degenerate" delta-wye and wye-delta transformations, see [5].)

This paper looks at the application of the graph transformations of the type given above to solve combinatorial problems on planar graphs. Our strategy is to devise ways of associating information with each edge of G – and to update the information as each transformation is applied – so as to preserve the relevant solution values at each stage of the reduction. We will refer to any such strategy as a *delta-wye reduction technique*. Delta-wye reduction techniques have been examined by Akers [1], who applied them to shortest path and maximum flow problems, and Lehman [17], who applied them to probabilistic networks and the bounding of connectedness probability. (See [5, 10] for complete accounts of these applications.) We examine three combinatorial enumeration problems – counting spanning trees, counting perfect matchings, and computing the Ising function – to develop solution algorithms that use the delta-wye reduction paradigm.

All of the problems studied in this paper are currently polynomially solvable by means of "determinental" methods. Counting spanning trees can be solved efficiently for general graphs using the determinental technique outlined first by Kirchoff [16]. Spanning trees of an n-vertex graph are counted by computing the determinant of an $(n-1)\times(n-1)$ matrix with integer entries. Counting spanning trees is an essential step in many methods for computing, bounding and approximating network reliability [6].

Counting perfect matchings can be solved efficiently for planar graphs [15], and more generally for graphs having a "Pfaffian orientation" [20]. Again the method involves the computation of a determinant, followed by a square root calculation (more precisely, the *Pfaffian* of an $n\times n$ matrix).

The Ising model is a classical combinatorial model in statistical physics. It was introduced by Lenz [18] and Ising [12]; for a good recent account, see [3]. The particular version of the model treated here is the "spin glass", or "zero-external field" model. The associated computational problem has associated with each edge (i,j) of G a specified *interaction energy* ω_{ij}. Each vertex of G can be assigned a *spin* of either $+$ or $-$, and a *configuration* of G is any assignment $\sigma=(\sigma_1,\ldots,\sigma_n)$ of signs (spins) to the n vertices $v_1,\ldots,v_n$ of G. The "system energy" for G is given by the *Ising partition function*

$$Z=\sum_{\sigma}\prod_{(i,j)\in E} e^{-\omega_{ij}\sigma_i\sigma_j},$$

where the sum is taken over all configurations σ of G. For general graphs, the problem of computing the Ising partition function exactly has been shown to be $\#P$-complete in several treatments [2, 13, 14], although a fully polynomial randomized approximation scheme has been developed when all of the interaction energies ω_{ij} are nonnegative [14]. The most celebrated completely solved case is when G is planar; see [15, 11]. As with the matching problem, the solution likewise involves computing the Pfaffian of a matrix. (Lovász and Plummer [20] give an excellent exposition of the graph-theoretical aspects of the Ising model solution using Pfaffians.)

The best current algorithm for computing determinants employs $O(n^{2.376})$ arithmetic operations on matrix entries [7], but appears to have too large a constant factor for practical combinatorial computations; in fact, for problems arising in practice, the naive $O(n^3)$ algorithm appears to be the sensible choice. For matrices whose support corresponds to a node–node incidence matrix of a planar graph, Lipton et al. [19] are able to use the planar separator theorem to calculate determinants in $O(n^{1.5})$ time. This gives $O(n^{1.5})$ algorithms for counting spanning trees and perfect matchings and computing the Ising function in planar graphs. There does not appear to be any known efficient algorithm for counting spanning trees in planar graphs that avoids the computation of a determinant.

All three problems given here arise as evaluations of the Tutte polynomial of the graph. Vertigan [23] has recently completely classified the complexity of all evaluations of the Tutte polynomial for planar graphs; we refer the reader to [23] for the relation between the Tutte polynomial of the graph and these problems. The Ising problem, in fact, corresponds to the problem of computing the Tutte polynomial on exactly those points on which the Tutte polynomial is NP-hard to compute *except* on planar graphs.

In this paper, we develop the associated combinatorial information that enables us to count spanning trees, compute the Ising partition function, or count the number of perfect matchings, and which can be easily updated with each of the six basic transformations. As a result, these three problems can be solved via the Feo–Provan algorithm using $O(n^2)$ arithmetic operations, and does not require the calculation of a determinant.

The delta-wye reduction technique is not currently the most efficient method for computing these three values – since the Lipton–Rose–Tarjan modification of the determinental technique yields an $O(n^{1.5})$ algorithm – although it is arguably the simplest to implement. There is also some evidence to indicate that the Feo–Provan procedure can be improved to reduce a planar graph in $O(n^{1.5})$ transformations; if true, the delta-wye reduction method could match the best known algorithms. The primary aim of this paper, however, is to demonstrate the application of the delta-wye reduction method to solve a wide variety of combinatorial problems which have up to now been solved by determinental methods. It is hoped that this, in turn, will encourage researchers both to improve the reduction procedure itself and to discover other problems to which it can be applied.

2. Spanning trees

In this section, we develop the updates used for counting the number of spanning trees in $\Delta Y\Delta$-reducible graphs. In fact, we solve a more general problem. Let $G = (V, E)$ be an undirected graph. For each edge $e \in E$, associate two nonzero real weights i_e and o_e, called the *inclusion* and *exclusion* weights of edge e, respectively. For any spanning tree $T \subseteq E$ of G, define the *weight* of T to be $\prod_{e \in T} i_e \prod_{e \in E \setminus T} o_e$. Then the total weight of the graph G is given as

$$W = \sum_{T} \prod_{e \in T} i_e \prod_{e \in E \setminus T} o_e$$

the sum taken over all spanning trees T of G. When $i_e = o_e = 1$ for all $e \in E$, the total weight of G is precisely the number of spanning trees of G.

We next show how these edge weights are updated under each of the six basic transformations. We first observe that if, for any single edge $e \in E$, the inclusion and exclusion weights are both multiplied by a factor δ, the effect on the total weight is likewise a factor of δ. We therefore maintain a single overall *scale* $S = \prod_{e \in E} o_e$ for G. In the process, each exclusion weight is normalized to 1, and each inclusion weight is normalized to $w_e = i_e / o_e$, so that

$$W = S \sum_{T} \prod_{e \in T} w_e$$

the sum taken over all spanning trees of G. Hence it is necessary to maintain only the normalized inclusion weight on each edge, as well as the overall scale, through each of the six basic transformations.

Suppose, then, that G is the graph prior to the transformation, with normalized weights on the edges and overall scale d.

If the transformation is the deletion of a loop e with weight a, observe that e occurs in no spanning tree of G, and hence does not affect the total weight. Thus the edge e can be eliminated without affecting the overall scale, or any other weights.

If the transformation is the deletion of a pendant edge e with weight a, observe that e is in *every* spanning tree of G, and hence the resulting graph has overall scale is multiplied by a, with other edge weights remaining unchanged.

If the transformation is a parallel reduction on two edges e, f with weights a, b, observe that every spanning tree contains at most one of e or f. Hence the replacement edge e' will have weight $a + b$, and the remaining edge weights and overall scale remain unchanged.

If the transformation is a series reduction on two edges e, f with weights a, b, observe that every spanning tree contains at least one of e or f. There are ab ways to include both e and f, and $a + b$ ways to exclude one of them; there is no way to exclude both and get a spanning tree. Then normalizing as before, the replacement edge e' has weight $ab/(a + b)$, the overall scale is multiplied by $a + b$, and again no other edge weights are changed.

Now we turn to the more complicated delta-wye and wye-delta transformations. Let G_Y be a graph with scale d, having a vertex w of degree 3 (a wye) adjacent to three vertices x, y, z via edges e, f, g of weights a, b, c, respectively. Let G_Δ be the same underlying graph as G_Y, but with vertex w and edges e, f, g removed and replaced by $e' = (y, z)$, $f' = (x, z)$ and $g' = (x, y)$, having weights α, β, γ, respectively; G_Δ has overall scale δ.

Now consider the total weight of G_Y and G_Δ. A spanning tree T of G_Y is one of five types, according to the structure of $T' = T \setminus \{e, f, g\}$:

1. T' has x, y, z in the same component;
2. T' has two components with x and y in the same component;
3. T' has two components with x and z in the same component;
4. T' has two components with y and z in the same component; or
5. T' has three components, containing x, y, and z, respectively.

The same typing applies to a tree T in G_Δ, depending upon the structure of $T' = T \setminus \{e', f', g'\}$.

For $i = 1, \ldots, 5$, let Σ_i be the sum – over all forests T' of $G_Y \setminus \{e, f, g\}$ $(= G_\Delta \setminus \{e', f', g'\})$ of type i above – of the product of the weights of the edges in T'. Then observe that the total weight of G_Y is

$$d[(a + b + c)\Sigma_1 + c(a + b)\Sigma_2 + b(a + c)\Sigma_3 + a(b + c)\Sigma_4 + abc\Sigma_5],$$

while the total weight of G_Δ is

$$\delta[(\Sigma_1 + (\alpha + \beta)\Sigma_2 + (\alpha + \gamma)\Sigma_3 + (\beta + \gamma)\Sigma_4 + (\alpha\beta + \alpha\gamma + \beta\gamma)\Sigma_5].$$

Equating the coefficients of Σ_1 through Σ_5 gives the following system of five equations:

$$d(a + b + c) = \delta,$$

$$\left.\begin{aligned} da(b + c) &= \delta(\beta + \gamma), \\ db(a + c) &= \delta(\alpha + \gamma), \\ dc(a + b) &= \delta(\alpha + \beta) \end{aligned}\right\} \quad or \quad \left\{\begin{aligned} dbc &= \delta\alpha, \\ dac &= \delta\beta, \\ dab &= \delta\gamma, \end{aligned}\right.$$

$$dabc = \delta(\alpha\beta + \alpha\gamma + \beta\gamma).$$

Given a, b, c, d, we have five equations to solve for the four unknowns $\alpha, \beta, \gamma, \delta$; and given $\alpha, \beta, \gamma, \delta$ we have five equations to solve for the four unknowns a, b, c, d. There does not appear to be any reason to expect a priori that the equations are consistent; however, we shall show that they are.

Given a, b, c, d, compute $\delta = d(a + b + c)$, satisfying (1). Then compute

$$\begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \frac{abc}{a + b + c} \begin{pmatrix} 1/a \\ 1/b \\ 1/c \end{pmatrix}.$$

These satisfy (2)–(4). It remains to verify that equation (5) is satisfied. Now $\delta(\alpha\beta + \alpha\gamma + \beta\gamma) = \delta\alpha\beta\gamma(1/\alpha + 1/\beta + 1/\gamma)$, which simplifies to $dabc$ as required.

To accomplish a delta-wye transformation, we are given $\alpha, \beta, \gamma, \delta$. Observe that $a = dabc/dbc$, $b = dabc/dac$ and $c = dabc/dab$ to get

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = (\alpha\beta + \alpha\gamma + \beta\gamma) \begin{pmatrix} 1/\alpha \\ 1/\beta \\ 1/\gamma \end{pmatrix}.$$

Then use (5) to compute

$$d = \frac{\delta}{(\alpha\beta + \alpha\gamma + \beta\gamma)(1/\alpha + 1/\beta + 1/\gamma)}.$$

In this way (2)–(5) are satisfied; to verify (1), observe that $a + b + c = (\alpha\beta + \alpha\gamma + \beta\gamma)(1/\alpha + 1/\beta + 1/\gamma)$. Again, the other edge weights are unchanged.

Using the six exact transformations developed, we obtain that, for any family of graphs $\mathcal{F}$ and any algorithm which delta-wye reduces any n-vertex graph in $\mathcal{F}$ to an edge in at most $f(n)$ transformations, we obtain an algorithm using $O(f(n))$ arithmetic operations for counting spanning trees of graphs in the family. The Feo–Provan algorithm therefore gives an algorithm for counting spanning trees using $O(n^2)$ arithmetic operations. Since the only arithmetic operations are additions, multiplications, and divisions of positive numbers, then in order to maintain accuracy of r decimal digits it is only required to maintain $r + O(\log n)$ significant digits accuracy in the intermediate numbers. Since the number of spanning trees of an n-vertex simple graph cannot exceed n^{n-2} then $r = O(n \log n)$ digits accuracy is sufficient. Hence the technique developed here, when applied to counting spanning trees, manipulates numbers no larger than those encountered in the determinant calculation via Kirchoff's formula.

This algorithm is closely related to the standard method for computing resistance of an electrical network using delta-wye and wye-delta transformations [4, 10]. In fact, to compute the resistance between two nodes s and t, let the inclusion weight of each edge be its conductance (the reciprocal of its resistance). Then add a new edge (s, t) whose inclusion weight is the variable x. All exclusion weights are unity. Then applying our algorithm, we obtain an expression of the form $k + lx$ for the total weight of spanning trees. The resistance is simply the ratio l/k (see [4, Ch. 2]).

3. The Ising problem

The Ising function given in Section 1 can be computed using the delta-wye reduction technique by applying similar machinery to that for counting spanning trees. Let $v_{ij}^1 = e^{-\omega_{ij}}$ and $v_{ij}^2 = e^{\omega_{ij}}$. Then the Ising partition function can be written

$$Z = \sum_{\sigma} \prod_{(i,j) \in E : \sigma_i \neq \sigma_j} v_{ij}^1 \prod_{(i,j) \in E : \sigma_i = \sigma_j} v_{ij}^2.$$

This brings out the similarity in structure between the Ising function and the spanning tree function. Let $X \subseteq V$ be the set of vertices i having $\sigma_i = +$, so that $\bar{X} = V \setminus X$ is the set of vertices i having $\sigma_i = -$. Define the *cut* $(X, \bar{X})$ to be the set of edges $\{(i,j) \in E: i \in X, j \in \bar{X}\}$. Then the Ising partition function can now be written as

$$Z = \sum_{X \subseteq V} \prod_{e \in (X, \bar{X})} v_e^1 \prod_{e \in E \setminus (X, \bar{X})} v_e^2.$$

Thus v_e^1 takes the place of i_e and v_e^2 takes the place of o_e, with the sum being taken over all vertex subsets X rather than all trees.

Continuing with this analogy, we scale Z by scale factor $S = \prod_{e \in E} v_e^2$, using a single normalized weight $w_e = v_e^1 / v_e^2$ for each edge $e \in E$. The Ising partition function now becomes

$$Z = S \sum_{X \subseteq V} \prod_{e \in (X, \bar{X})} w_e. \qquad (*)$$

Now consider the effect on the weights and scale factor of the current graph G when applying each of the six basic transformations.

If G has a loop e with weight a, deletion of e has no effect on either the weights or scale of Z.

If G has a pendant edge e with weight a, deletion of e results in a multiplication of S by $1 + a$.

If G has two edges e and f in parallel, having weights a and b, respectively, then the replacement edge e' will have weight ab and the overall scale remains unchanged.

If G has two edges e and f in series, having weights a and b, respectively, then the edge e' replacing them has weight $(a + b)/(1 + ab)$, and S is multiplied by $1 + ab$.

To update the values for the delta-wye and wye-delta transformations, let G_Y have a vertex w of degree 3 adjacent to three vertices x, y, z by edges e, f, g with weights a, b, c, respectively, and scale d. Let G_Δ have the same edges as G_Y except that e, f, g are replaced by $e' = (y, z)$, $f' = (x, z)$ and $g' = (x, y)$ with weights α, β, γ, respectively, and scale δ. Similar to the spanning tree analysis, let Σ_x, Σ_y, Σ_z, and Σ_0 be, respectively, the sum of the product of the weights of sets $(X, \bar{X}) \setminus \{e, f, g\}$ in G_Y ($= (X, \bar{X}) \setminus \{e', f', g'\}$ in G_Δ) with X and $\bar{X}$ separating, respectively, x from $\{y, z\}$, y from $\{x, z\}$, z from $\{x, y\}$, and no subset of $\{x, y, z\}$. Then the total weight of G_Y is

$$d[(a + bc)\Sigma_x + (b + ac)\Sigma_y + (c + ab)\Sigma_z + (1 + abc)\Sigma_0],$$

while the total weight of G_Δ is

$$\delta[\beta\gamma\Sigma_x + \alpha\gamma\Sigma_y + \alpha\beta\Sigma_z + \Sigma_0].$$

Equating the coefficients of Σ_x, Σ_y, Σ_z, and Σ_0, we obtain the four equations

$$\begin{aligned} \delta\beta\gamma &= A = d(a + bc), & \delta\alpha\gamma &= B = d(b + ac), \\ \delta\alpha\beta &= C = d(c + ab), & \delta &= D = d(1 + abc). \end{aligned} \qquad (**)$$

When all four equations hold, the Ising partition functions of G_Y and G_Δ agree.

To transform G_Y to G_Δ, compute A, B, C, D using the right-hand sides of the Eqs. (**). Then observe that

$$\alpha^2 = \frac{BC}{AD}, \qquad \beta^2 = \frac{AC}{BD}, \qquad \gamma^2 = \frac{AB}{CD},$$

so that $\alpha = \sqrt{BC/AD}$, $\beta = \sqrt{AC/BD}$, $\gamma = \sqrt{AB/CD}$, and $\delta = D$.

To transform G_Δ to G_Y, compute A, B, C, D using the left-hand sides of (**). Now set

$$\tilde{A} = A - B - C + D = d(1+a)(1-b)(1-c);$$

$$\tilde{B} = -A + B - C + D = d(1-a)(1+b)(1-c);$$

$$\tilde{C} = -A - B + C + D = d(1-a)(1-b)(1+c);$$

$$\tilde{D} = A + B + C + D = d(1+a)(1+b)(1+c).$$

Then proceed as above, observing that

$$\hat{A} = \frac{\tilde{B}\tilde{C}}{\tilde{A}\tilde{D}} = \left(\frac{1-a}{1+a}\right)^2, \qquad \hat{B} = \frac{\tilde{A}\tilde{C}}{\tilde{B}\tilde{D}} = \left(\frac{1-b}{1+b}\right)^2, \qquad \hat{C} = \frac{\tilde{A}\tilde{B}}{\tilde{C}\tilde{D}} = \left(\frac{1-c}{1+c}\right)^2.$$

So that we get $a = (1-\sqrt{\hat{A}})/(1+\sqrt{\hat{A}})$, $b = (1-\sqrt{\hat{B}})/(1+\sqrt{\hat{B}})$, $c = (1-\sqrt{\hat{C}})/(1+\sqrt{\hat{C}})$, and $d = \tilde{D}/(1+a)(1+b)(1+c)$.

The value of the Ising partition function can therefore be preserved through each of the six basic transformations, using only a constant number of primitive arithmetic operations and square root computations for each. Analogous to Section 3, we get that the Feo–Provan algorithm computes Z in $O(n^2)$ arithmetic operations. Since these operations involve subtractions as well as square roots, however, it becomes more difficult to estimate the number of digits accuracy necessary in the intermediate calculations to obtain a required number of digits accuracy in Z. There are compelling reasons to believe that, with the appropriate representations for the intermediate data, it might be possible to perform rational arithmetic computations at each step, with perhaps one square root taken at the end. We leave the search for such a representation as an interesting open problem.

4. Perfect matchings

The connection between the Ising partition function and the number of perfect matchings in a graph has been exploited by Jerrum [13] to prove that computing the Ising function is $\#$P-complete for general graphs. We exploit the reverse of Jerrum's transformation to allow delta-wye reduction techniques to compute the number of perfect matchings in a planar graph. Let G be a graph with an even number of vertices (a necessary condition for G to have a perfect matching). Then G contains an even number of vertices of even degree.

We form a graph $\hat{G}$ containing only odd degree vertices, and having the same number of perfect matchings as G, as follows. As long as even degree vertices remain, choose two even degree vertices x, y. Choose any sequence of vertices $x = v_0, v_1, \ldots, v_{l-1}, v_l = y$; for each $0 \leqslant i < l$, add two new vertices u_i and c_i, and edges (v_i, c_i), (c_i, v_{i+1}) and (c_i, u_i). This leaves unchanged the number of perfect matchings of G (since u_i must be matched to c_i), and reduces the number of even degree vertices by two. Repeating this until all vertices have odd degree yields a graph $\hat{G}$. We remark that if G is planar, $\hat{G}$ can be chosen to be planar by ensuring that v_i and v_{i+1} are on a common face for each $0 \leqslant i < k$. In fact, $O(n)$ edges suffice to form $\hat{G}$ from G.

To count the perfect matchings in $\hat{G}$, we note that since $\hat{G}$ has all vertices of odd degree, then the complement of a perfect matching with respect to $\hat{G}$ is an *Eulerian subgraph* (subgraph all of whose vertex degrees are even) of maximum edge cardinality $k = \hat{m} - \hat{n}/2$, where $\hat{n}$ and $\hat{m}$ are the number of vertices and edges of $\hat{G}$, respectively. Now suppose that $\hat{G}$ is planar, and let $G' = (V', E')$ be the planar dual of $\hat{G}$. Counting Eulerian subgraphs of $\hat{G}$ with k edges is equivalent to counting *cuts* $(X, \bar{X})$ in G' of cardinality k.

Now to count the cuts of G', we can use version $(*)$ of the Ising partition function, with weights $w_e = 2^{\hat{m}}$ and scale $S = 1$. Eq. $(*)$ now becomes

$$Z = \sum_{X \subseteq V} \prod_{e \in (X, \bar{X})} 2^{\hat{m}} = \sum_{X \subseteq V} 2^{|(X, \bar{X})|\hat{m}}.$$

This is simply the numerical form of a generating function for cuts, where the coefficient of $(2^{\hat{m}})^i$ is exactly *twice* the number of cuts of cardinality i (since each distinct edge set $(X, \bar{X})$ is represented by two terms $(X, \bar{X})$ and $(\bar{X}, X)$ in the Ising partition function). Since the number of cuts of each cardinality cannot exceed $2^{\hat{m}}$, and each cut of cardinality i contributes at least $2^{\hat{m}}$ times as much as a cut of smaller cardinality, then the number of cuts of maximum cardinality (and hence the number of perfect matchings of G) is exactly $\frac{1}{2}\lfloor Z/2^{k\hat{m}} \rfloor$. We can in fact determine in a similar manner the number of cuts of *any* cardinality in G, although the ones of cardinality k are the only ones that apply to this problem.

The computation of the Ising partition function – and hence the number of cardinality cuts in G – requires only reducibility by the six basic transformations, while counting perfect matchings requires in addition a surface duality to dualize into a cut counting problem. For planar graphs, the duality is straightforward, and yields an algorithm for counting perfect matchings that requires only $O(n^2)$ primitive arithmetic operations and square roots. The final value of Z will be an integer of size at most $2^{\hat{m}^2 + \hat{n}}$, that is, having $O(n^2)$ digits. As in the previous section, however, the number of digits accuracy in the intermediate steps is difficult to determine.

Finally, we remark on a further important difference between the approach using Pfaffians and using the delta-wye reduction method: while $K_{3,3}$ has no Pfaffian orientation [20], it is a $\Delta Y\Delta$-reducible graph. This suggests that the approach developed here expands the class of graphs for which perfect matchings can be

counted efficiently. A classification of $\Delta Y\Delta$-reducible graphs is needed to examine the extension of our approach.

Acknowledgements

Research of the first author is supported by NSERC Canada under grant number A0579. Research of the second author is partially supported by NSF Grant CCR-9200572; work performed while on leave at the University of Waterloo.

References

[1] S.B. Akers Jr, The use of wye-delta transformations in network simplification, Oper. Res. 8 (1960) 311–323.
[2] F. Barahona, On the computational complexity of Ising spin glass models, J. Phys. A 15 (1982) 3241–3253.
[3] R.J. Baxter, Exactly Solved Models in Statistical Mechanics (Academic Press, London, 1982).
[4] B. Bollobás, Graph Theory: An Introductory Course (Springer, Berlin, 1979).
[5] M.K. Chari, T.A. Feo and J.S. Provan, The delta-wye approximation procedure for two-terminal reliability, Oper. Res., to appear.
[6] C.J. Colbourn, The Combinatorics of Network Reliability (Oxford University Press, Oxford, 1987).
[7] D. Coppersmith and S. Winograd, Matrix multiplication via arithmetic progressions, in: 19th ACM Symposium on the Theory of Computing (1987) 1–6.
[8] E.S. Elmallah and C.J. Colbourn, On two dual classes of planar graphs, Discrete Math. 80 (1990) 21–40.
[9] G.V. Epifanov, Reduction of a plane graph to an edge by a star-triangle transformation, Soviet Math. Dokl. 166 (1966) 13–17.
[10] T.A. Feo and J.S. Provan, Delta-wye transformations and the efficient reduction of two-terminal planar graphs, Oper. Res. 41 (1993) 572–582.
[11] M.E. Fisher, On the dimer solution of planar Ising models, J. Math. Phys. 7 (1966) 1776–1781.
[12] E. Ising, Beitrag zur Theorie des Ferromagnetismus, Z. Phys. 31 (1925) 253–258.
[13] M. Jerrum, 2-dimensional monomer-dimer problems are computationally intractable, J. Statist. Phys. 48 (1987) 121–134.
[14] M. Jerrum and A. Sinclair, Polynomial-time approximation algorithms for the Ising model, SIAM J. Comput. 22 (1993) 1087–1116.
[15] P.W. Kasteleyn, Graph theory and crystal physics, in: F. Harary, ed., Graph Theory and Theoretical Physics (Academic Press, London, 1967).
[16] G. Kirchoff, Uber die Auflosung der Gleichungen auf welche man sei der Untersuching der Linearen Verteilung Galvansicher Strome Gefuhrt wind, Poggendorg's Ann. Phys. Chem. 72 (1847) 497–508.
[17] A.B. Lehman, Wye-delta transformations in probabilistic networks, J. SIAM 11 (1962) 773–805.
[18] W. Lenz, Beitrag zur Verstandnis der magnetischen Erscheinungen in fester Korpern, Z. Phys. 21 (1920) 613–615.
[19] R.J. Lipton, D. Rose and R.E. Tarjan, Generalized nested dissection, SIAM J. Numer. Anal. 16 (1979) 346–358.
[20] L. Lovász and M.D. Plummer, Matching Theory (North-Holland, Amsterdam, 1986).
[21] K. Truemper, On the delta-wye reduction for planar graphs, J. Graph Theory 13 (1989) 141–148.
[22] J. Valdes, R.E. Tarjan and E.L. Lawler, The recognition of series parallel digraphs, SIAM J. Comput. 11 (1982) 298–313.
[23] D. Vertigan, The computational complexity of Tutte invariants for planar graphs, SIAM J. Comput., to appear.