Matching, Edge-Colouring, Dimers

Alexander Schrijver

CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands, and Department of Mathematics, University of Amsterdam, Plantage Muidergracht 24, 1018 TV Amsterdam, The Netherlands.

Abstract. We survey some recent results on finding and counting perfect matchings in regular bipartite graphs, with applications to bipartite edge-colouring and the dimer constant. Main results are improved complexity bounds for finding a perfect matching in a regular bipartite graph and for edge-colouring bipartite graphs, the solution of a problem of Erdős and Rényi concerning lower bounds for the number of perfect matchings, and an improved lower bound for the 3-dimensional dimer constant.

1 Finding a perfect matching in a regular bipartite graph

The fastest known algorithms for finding a perfect matching in a general bipartite graph have running time of order about $O(\sqrt{n}\,m)$ (Hopcroft and Karp [12], Feder and Motwani [8]) or $O(n^{2.376})$ (Ibarra and Moran [13]). For *regular* bipartite graphs, however, faster algorithms are known: Cole and Hopcroft [4] gave an $O(m \log n)$ algorithm, while Cole [3] gave an $O(2^{2^{O(k)}} n)$ algorithm, where k is the degree of the vertices. So the latter algorithm is linear-time for any fixed k. We now describe an easy $O(k^2 n)$ $(= O(km))$ algorithm ([17]). Here is the idea for $k = 3$.

Let G be a 3-regular bipartite graph. Find a circuit C in G, by finding a path $Q = v_0, e_1, v_1, \ldots$, till we arrive at a vertex v_k where we have been before (that is, $v_k = v_i$ for some $i < k$). Next delete from G every second edge of C. The remaining edges of C form the middle edges of paths of length 3 in the remaining graph G'. Replace each such path P by an edge e_P connecting the ends of P. The resulting graph G'' is 3-regular and bipartite. Find recursively a perfect matching M in G''. Replace any edge e_P that occurs in M by the two end edges of P. For each of the other paths P, add its middle edge to M. This gives a perfect matching in G', hence in G, as required.

To obtain a linear-time algorithm, one should use in the recursion the tail $v_0, e_1, v_1, \ldots, v_i$ of the path Q to find the next circuit in G''. Then the time spent on running through the tail when finding the successive circuits will not be lost, and any recursive step takes amortized time $|VC|$. Since in any recursive step, the size of the graph reduces also by $|VC|$, the algorithm is linear-time.

This gives the theorem of Cole [3]:

Theorem 1. *A perfect matching in a 3-regular bipartite graph can be found in linear time.*

We next describe the extension to k-regular bipartite graphs. This uses a weighting of the edges.

Let $G = (V, E)$ be a k-regular bipartite graph. Initially, set $w(e) := 1$ for each edge e. Next, iteratively, find a circuit C in G, split the edge set EC of C into two matchings M and N, in such a way that

$$\sum_{e \in M} w(e) \geq \sum_{e \in N} w(e), \tag{1}$$

reset $w(e) := w(e) + 1$ if $e \in M$ and $w(e) := w(e) - 1$ if $e \in N$, delete the edges e with $w(e) = 0$, and iterate.

Again we find C by following a path, and we keep its tail (if nontrivial) for the next iteration. Note that the resetting maintains the property

$$\sum_{e \ni v} w(e) = k \text{ for each vertex } v. \tag{2}$$

So as long as there exist edges e with $w(e) < k$, we can find a circuit. Hence, the iterations stop if $w(e) = k$ for each edge e. In that case, the edges form a perfect matching, and we are done.

The key to estimating the running time is considering

$$\sum_{e \in E} w(e)^2. \tag{3}$$

This sum is bounded by $\frac{1}{2}k^2|V|$. Moreover, in any iteration, this sum increases by

$$\begin{aligned} &\sum_{e \in M} ((w(e) + 1)^2 - w(e)^2) + \sum_{e \in N} ((w(e) - 1)^2 - w(e)^2) \\ &= \sum_{e \in M} (2w(e) + 1) + \sum_{e \in N} (-2w(e) + 1) \geq |M| + |N| = |EC| \end{aligned} \tag{4}$$

(by (1)). Since the amortized time of any iteration is proportional to $|EC|$, this gives an $O(k^2 n) = O(km)$ running time bound ([17]):

Theorem 2. *A perfect matching in a k-regular bipartite graph can be found in $O(km)$ time.*

2 Edge-colouring

The latter result implies an $O(km)$ algorithm for finding an optimum edge-colouring of a bipartite graph G, where k denotes the maximum degree. (An optimum edge-colouring colours the edges with k colours such that each colour forms a matching.)

First observe that one trivially obtains an $O(k^2 m)$ algorithm. Indeed, we can assume that the bipartite graph G is k-regular (as we can extend G to a k-regular bipartite graph, in linear time). Then iteratively find a perfect matching in G and delete it from G. The successive perfect matchings form the colours. This can be done by applying k times the $O(km)$ algorithm, yielding $O(k^2m)$.

However, with a method of Gabow [10], one may speed up this. If k is odd, find a perfect matching in G and delete it from G. If k is even, find an Eulerian orientation of G (that is, an orientation such that the indegree in each vertex is equal to its outdegree). This can be done in linear time. Next split the edge set E of G into the set E_1 of edges oriented from one colour class of G to the other, and the set E_2 of edges oriented in the opposite direction. Then (V, E_1) and (V, E_2) are $\frac{1}{2}k$-regular bipartite graphs, in which we can find optimum edge-colourings recursively. Combining them gives an optimum edge-colouring of G. The running time is

$$O(km + 2(\tfrac{1}{2}k\tfrac{1}{2}m) + 4(\tfrac{1}{4}k\tfrac{1}{4}m) + \cdots) = O(km). \tag{5}$$

Hence:

Theorem 3. *An optimum edge-colouring of a bipartite graph can be found in $O(km)$ time, where k is the maximum degree.*

3 Speed-up of Cole, Ost, and Schirra

The above $O(km)$ algorithm for perfect matching in k-regular bipartite graphs raises the question if there is a linear-time algorithm, independent of k. This was resolved positively by Cole, Ost, and Schirra [5], by a refinement of the method above, utilizing the data-structure of 'self-adjusting binary trees'. We outline their method.

A first improvement is not to replace $w(e)$ by $w(e) \pm 1$ for the edges in C, but by $w(e) \pm \alpha$, where α is the minimum weight of the edges in N. So at least one edge in N gets weight 0.

A second improvement is to store the paths ('chains') left in the circuit C (after removing the edges of weight 0), so that these chains can be used to speed up later circuit searches. This requires that if in a later circuit search we hit any such chain, then relatively fast we should be able to identify the ends of the chain. (If we have to follow the chain vertex by vertex till its end, no gain in running time is obtained.) This can be done by supplying these chains with the data structure of self-adjusting binary trees (cf. Tarjan [19]). To get the required running time, it turns out that these chains should have length at most k^2 — in the case that they are longer, split them into chains of length about k^2.

A third improvement is a preprocessing that reduces the number of edges of the graph from $\frac{1}{2}kn$ to at most $n \log_2 k$. This is obtained as follows. Start with setting $w(e) := 1$ for each edge e. Next, successively, for $i = 0, 1, \ldots, \lfloor \log_2 k \rfloor$, do the following. Consider the set E_i of edges of weight 2^i. Iteratively (as above) find a circuit C in E_i, split EC arbitrarily into matchings M and N, and reset $w(e) := w(e) + 2^i = 2^{i+1}$ if $e \in M$ and $w(e) := w(e) - 2^i = 0$ if $e \in N$. (So in each iteration, the set E_i changes.) In linear time we arrive at the situation that

E_i contains no circuits, implying $|E_i| \leq n-1$. Then we go over to the case $i+1$. So we end up with at most (about) $n \log_2 k$ edges, together with a weighting w satisfying (2).

For each i, the preprocessing takes time linear in the size of the initial E_i, which is at most $2^{-i}m$. Hence the preprocessing takes $O(m)$ time in total. It turns out that, using the first two improvements, the rest of the algorithm takes $O(n \log^3 k)$ only, which is faster than $O(m)$.

This gives the theorem of Cole, Ost, and Schirra [5]:

Theorem 4. *A perfect matching in a regular bipartite graph can be found in linear time.*

With the method described in Section 2, it has as consequence:

Corollary 1. *An optimum edge-colouring of a bipartite graph can be found in* $O(m \log k)$ *time, where* k *is the maximum degree.*

4 From finding to counting perfect matchings

We now go over to the problem of counting perfect matchings, or rather giving a lower bound for their number. We first relate the algorithm described in Section 1, for finding a perfect matching in a 3-regular bipartite graph, to a lower bound of Voorhoeve [21] on the number of such perfect matchings.

To this end, we modify the algorithm slightly. We may note that when following the path Q in finding the circuit, we can start immediately from the beginning with removing edges. We don't have to wait till we have a circuit. This can be made more precise as follows.

Call a bipartite graph *almost* 3*-regular* if all vertices have degree 3, except for two vertices of degree 2 (automatically belonging to different colour classes). So an almost 3-regular bipartite graph arises by deleting one edge from a 3-regular bipartite graph. Hence a linear-time algorithm for finding a perfect matching in an almost 3-regular bipartite graph yields the same for 3-regular bipartite graphs. We describe such an algorithm.

Let G be an almost 3-regular bipartite graph, and let u be any of the two vertices of degree 2. To find a perfect matching, we can assume that u is not incident with the other vertex of degree 2, and that it has two distinct neighbours, x and y say. (Otherwise, there is an easy reduction.)

Let u, s, t be the neighbours of x. Delete edge xs. Then edge ux becomes the middle edge of a path $P = (y, u, x, t)$. Replace it by a new edge e_P connecting y and t. Find recursively a perfect matching M in the new graph G'. If e_P is in M, replace it by yu and xt. If e_P is not in M, add ux to M. We end up with a perfect matching in G.

As each iteration takes constant time, and as it reduces the number of vertices by 2, this gives a linear-time algorithm. This might be easier to implement than the algorithm described earlier, since only local operations are performed.

This method is in fact inspired by the method of Voorhoeve [21] to prove that any 3-regular bipartite graph has at least

$$(\tfrac{4}{3})^n \tag{6}$$

perfect matchings, where, for convenience, n denotes *half* of the number of vertices. To prove this bound, it suffices to show that each almost 3-regular bipartite graph has at least $(\frac{4}{3})^n$ perfect matchings. Again, choose a vertex u of degree 2, and we may assume that it has two distinct neighbours of degree 3. (Otherwise, there is an easy induction.) Let $e_1, \ldots, e_4$ be the edges incident with a neighbour of u but not with u. For $i = 1, \ldots, 4$, let G_i be the graph obtained from G by deleting edge e_i. Denote the number of perfect matchings in any graph H by $\pi(H)$. Then, by induction,

$$\pi(G_i) \geq (\tfrac{4}{3})^{n-1} \tag{7}$$

for $i = 1, \ldots, 4$, since replacing the path of length 3 through u by a new edge, gives an almost 3-regular bipartite graph H_i with $2(n-1)$ vertices and with $\pi(H_i) = \pi(G_i)$. Moreover,

$$\pi(G_1) + \cdots + \pi(G_4) = 3\pi(G), \tag{8}$$

since each perfect matching M in G is maintained in precisely three of the G_i (as M contains precisely one of $e_1, \ldots, e_4$). Combining (7) and (8) gives $\pi(G) \geq (\frac{4}{3})^n$, as required.

Incidentally, this may look like an exact inductive calculation of $\pi(G)$, but strict inequality is obtained in the reduction if u has no two distinct neighbours of degree 3.

So we have proved the theorem of Voorhoeve [21]:

Theorem 5. *Any 3-regular bipartite graph on $2n$ vertices has at least $(\frac{4}{3})^n$ perfect matchings.*

With this, Voorhoeve answered a question posed by Erdős and Rényi [6] whether there exists an exponential lower bound on the number of perfect matchings in 3-regular bipartite graphs. (The best bound proved before is only linear in n.)

Erdős and Rényi formulated their question in terms of permanents, which relates to the Van der Waerden conjecture (which was not yet proved when Voorhoeve gave his bound). The *permanent* of an $n \times n$ matrix $A = (a_{i,j})$ is

$$\mathrm{per} A := \sum_{\pi} \prod_{i=1}^{n} a_{i,\pi(i)}, \tag{9}$$

where the sum ranges over all permutations π of $\{1, \ldots, n\}$. So if A is nonnegative and integer, and we make the bipartite graph G with colour classes $\{u_1, \ldots, u_n\}$ and $\{v_1, \ldots, v_n\}$ and with $a_{i,j}$ edges connecting u_i and v_j (for $i, j = 1, \ldots, n$), then $\mathrm{per} A$ is equal to the number of perfect matchings in G.

Call a matrix *k-regular* if it is nonnegative and integer and if each row sum and each column sum is equal to k. Then Erdős and Rényi asked for an exponential lower bound for the permanents of 3-regular matrices.

The Van der Waerden conjecture (van der Waerden [22]) asserts that the permanents of any $n \times n$ doubly stochastic matrix is at least

$$\frac{n!}{n^n}. \tag{10}$$

(A matrix is *doubly stochastic* if it is nonnegative and each row sum and each column sum is equal to 1.) The value (10) is attained if all entries of the matrix are equal to $\frac{1}{n}$. Van der Waerden's conjecture remained open for more than half a century, despite considerable research efforts, and was finally proved by Falikman [7].

For each k-regular matrix A, the matrix $\frac{1}{k}A$ is doubly stochastic and satisfies $\mathrm{per}\frac{1}{k}A = k^{-n}\mathrm{per}A$. So Van der Waerden's conjecture implies that the permanent of any k-regular matrix is at least

$$\frac{k^n n!}{n^n} \approx (\frac{k}{e})^n. \tag{11}$$

(This consequence in fact can be seen to be equivalent to Van der Waerden's conjecture.) Hence also Falikman's theorem implies an exponential lower bound on the number of perfect matchings in 3-regular bipartite graphs. The lower bound $(\frac{k}{e})^n$ was proved by Bang [1] and Friedland [9], thus also providing a solution of Erdős and Rényi's question.

It can be proved that the ground number $\frac{4}{3}$ in Voorhoeve's bound is best possible ([18]). To this end, let μ_3 be the largest real such that each 3-regular bipartite graph on $2n$ vertices has at least μ_3^n perfect matchings. So $\mu_3 \geq \frac{4}{3}$.

To prove the reverse inequality, fix n, and consider the collection $\mathcal{G}$ of 3-regular bipartite graphs with colour classes $\{u_1, \ldots, u_n\}$ and $\{v_1, \ldots, v_n\}$ and with (labeled) edges $e_1, \ldots, e_{3n}$. Then

$$|\mathcal{G}| = \left(\frac{(3n)!}{3!^n}\right)^2. \tag{12}$$

Indeed, it is equal to the square of the number of ordered partitions of $\{1, \ldots, 3n\}$ into n classes of size 3.

We can also precisely count for how many graphs G in $\mathcal{G}$, a given subset M of $\{1, \ldots, 3n\}$ of size n forms a perfect matching in G:

$$\left(n!\frac{(2n)!}{2^n}\right)^2. \tag{13}$$

Since M can be chosen in $\binom{3n}{n}$ ways, this implies that the number of pairs G, M with $G \in \mathcal{G}$ and M is a perfect matching in G is equal to

$$\binom{3n}{n}\left(n!\frac{(2n)!}{2^n}\right)^2. \tag{14}$$

By (12) and by definition of μ_3, (14) has as lower bound:

$$\left(\frac{(3n)!}{3!^n}\right)^2 \mu_3^n. \tag{15}$$

Therefore,

$$\mu_3 \leq \left(\binom{3n}{n}\left(n!\frac{(2n)!}{2^n}\right)^2\left(\frac{3!^n}{(3n)!}\right)^2\right)^{1/n} \stackrel{n\to\infty}{\longrightarrow} \frac{4}{3}. \tag{16}$$

(The latter limit uses Stirling's formula.) So $\mu_3 = \frac{4}{3}$.

5 General k

Erdős and Rényi also asked for the value, for any k, of the largest real μ_k such that each k-regular bipartite graph G on $2n$ vertices has at least μ_k^n perfect matchings. So by Falikman's theorem (in fact, already by the results of Bang and Friedland), $\mu_k \geq \frac{k}{e}$. On the other hand, the same method as just described gives ([18])

$$\mu_k \leq \frac{(k-1)^{k-1}}{k^{k-2}}. \tag{17}$$

In [18] it was also conjectured that equality holds:

$$\mu_k = \frac{(k-1)^{k-1}}{k^{k-2}}. \tag{18}$$

This in fact was be proved in [16]. Hence:

Theorem 6. *Each k-regular bipartite graph with $2n$ vertices has at least*

$$\left(\frac{(k-1)^{k-1}}{k^{k-2}}\right)^n \tag{19}$$

perfect matchings.

In contrast with the simplicity of Voorhoeve's method for the case $k = 3$, the proof for general k is highly complicated. It is based on a technique of assigning weights to the edges of the graph similar to the algorithm for finding a perfect matching in a k-regular bipartite graph described in Section 1.

Let us briefly relate this bound to Falikman's bound. Both bounds are asymptotically best possible, in different asymptotic directions. Let $\mu(k,n)$ denote the minimum permanent of k-regular $n \times n$ matrices. (Equivalently, of the minimum number of perfect matchings, taken over all k-regular bipartite graphs with $2n$ vertices.) So

$$\mu_k = \inf_{n\in\mathbb{N}} \mu(k,n)^{1/n}. \tag{20}$$

Then in one asymptotic direction one has by (18):

$$\inf_{n\in\mathbb{N}} \frac{\mu(k,n)^{1/n}}{k} = \frac{1}{k}\mu_k = \left(\frac{k-1}{k}\right)^{k-1}. \tag{21}$$

In another direction, by Falikman's theorem:

$$\inf_{k\in\mathbb{N}} \frac{\mu(k,n)^{1/n}}{k} = \frac{n!^{1/n}}{n}. \tag{22}$$

Note that both in (21) and in (22), the right-hand term converges to $1/e$, if k or n tends to infinity.

6 Application to the 3D dimer constant

We finally apply the lower bound described in Theorem 6 to obtain a better lower bound for the 3-dimensional dimer problem. This is one of the classical unsolved problems in solid-state chemistry. For integers d, n, consider the 'block' $H_{d,n}$, which is the graph with vertex set $\{1,\ldots,n\}^d$, two vertices being adjacent if and only if their Euclidean distance is 1. In this context, an edge is called a *dimer*, and a perfect matching a *dimer tiling*. Let $t_{d,n}$ denote the number of dimer tilings of $H_{d,n}$. So $t_{d,n} > 0$ if and only if n is even.

Hammersley [11] showed that

$$\lambda_d := \lim_{n\to\infty} \frac{1}{(2n)^d} \log t_{d,2n} \tag{23}$$

exists. In fact

$$\lim_{n\to\infty} \frac{1}{(2n)^d} \log t_{d,2n} = \sup_n \frac{1}{(2n)^d} \log t_{d,2n}. \tag{24}$$

Otherwise, there exists a k such that

$$\liminf_{n\to\infty} \frac{1}{(2n)^d} \log t_{d,2n} < \frac{1}{(2k)^d} \log t_{d,2k}. \tag{25}$$

However,

$$t_{d,2n} \geq (t_{d,2k})^{\lfloor \frac{n}{k} \rfloor^d}, \tag{26}$$

since $H_{d,2n}$ contains $\lfloor \frac{n}{k} \rfloor^d$ disjoint copies of $H_{d,2k}$ such that the rest has a perfect matching. This implies that the left-hand side in (25) is at least

$$\liminf_{n\to\infty} \frac{\lfloor \frac{n}{k} \rfloor^d}{(2n)^d} \log t_{d,2k}, \tag{27}$$

which is equal to the right-hand side of (25) — a contradiction.

So λ_d is defined. For $d = 2$, the value of λ_d was determined precisely by Kasteleyn [14] and Temperley and Fisher [20]:

$$\lambda_2 = \frac{1}{\pi} \sum_{i=0}^{\infty} \frac{(-1)^i}{(2i+1)^2} = 0.29156090\ldots. \tag{28}$$

The proof uses the fact that $H_{2,n}$ is planar, and that the graph therefore has a 'Pfaffian' orientation, making it possible to count dimer tilings by calculating a determinant.

For dimensions larger than two, no such orientation exists, and no exact formula for λ_d is known. Since $H_{d,n}$ is bipartite and 'almost' $2d$-regular, one could try to apply the results obtained earlier. In fact one has:

Theorem 7. $\lambda_d \geq \frac{1}{2}\log\mu_{2d}$.

To see this, for each $i \in \{1,\ldots,d\}$ and each $j \in \{1, 2n\}$, let $M_{i,j}$ be a perfect matching in the subgraph of $H_{d,2n}$ spanned by

$$\{x \in \{1,\ldots,2n\}^d \mid x_i = j\}. \tag{29}$$

(So this set represents a 'face' of $H_{d,2n}$.) Let $H'_{d,2n}$ be the $2d$-regular bipartite graph obtained from $H_{d,2n}$ by adding parallel edges for the edges in the $M_{i,j}$. Then $H'_{d,2n}$ has more perfect matching than $H_{d,2n}$ has, but not too much more:

$$\pi(H'_{d,2n}) \leq 2^{d(2n)^{d-1}}\pi(H_{d,2n}). \tag{30}$$

This follows from the facts that we have added $d(2n)^{d-1}$ parallel edges, and that adding any such edge at most doubles the number of perfect matchings.

Since $\pi(H'_{d,2n}) \geq \mu_{2d}^{(2n)^d/2}$ (by definition of μ_{2d}), we have

$$\pi(H_{d,2n}) \geq 2^{-d(2n)^{d-1}}\mu_{2d}^{(2n)^d/2}. \tag{31}$$

Therefore,

$$\begin{aligned}\lambda_d \geq \sup_n \frac{1}{(2n)^d}\log\left(2^{-d(2n)^{d-1}}\mu_{2d}^{(2n)^d/2}\right) &= \sup_n(\tfrac{1}{2}\log\mu_{2d} - \frac{d\log 2}{2n}) \\ &= \tfrac{1}{2}\log\mu_{2d},\end{aligned} \tag{32}$$

proving Theorem 7.

Evaluation for $d = 3$ by using $\mu_6 = 5^5/6^4$, gives the best known lower bound for λ_3:

$$\lambda_3 \geq 0.44007584\ldots. \tag{33}$$

The best known upper bound is due to Lundow [15]: $0.457547\ldots$. Computational experiments of Beichl and Sullivan [2] suggest that $\lambda_3 = 0.4466 \pm 0.0006$.

References

[1] T. Bang, *On matrix-functions giving a good approximation to the v.d. Waerden permanent conjecture*, Preprint Series 1979 No. 30, Matematisk Institut, Københavns Universitet, Copenhagen, 1979.

[2] I. Beichl, F. Sullivan, Approximating the permanent via importance sampling with application to the dimer covering problem, *Journal of Computational Physics* 149 (1999) 128–147.

[3] R.J. Cole, *Two Problems in Graph Theory*, Ph.D. Thesis, Cornell University, Ithaca, New York, 1982.

[4] R. Cole, J. Hopcroft, On edge coloring bipartite graphs, *SIAM Journal on Computing* 11 (1982) 540–546.

[5] R. Cole, K. Ost, S. Schirra, Edge-coloring bipartite multigraphs in $O(E\log D)$ time, *Combinatorica* 21 (2001) 5–12.

[6] P. Erdős, A. Rényi, On random matrices II, *Studia Scientiarum Mathematicarum Hungarica* 3 (1968) 459–464.

[7] D.I. Falikman, Proof of the van der Waerden conjecture regarding the permanent of a doubly stochastic matrix [in Russian], *Matematicheskie Zametki* 29 (1981) 931–938 [English translation: *Mathematical Notes of the Academy of Sciences of the USSR* 29 (1981) 475–479].

[8] T. Feder, R. Motwani, Clique partitions, graph compression and speeding-up algorithms, *Journal of Computer and System Sciences* 51 (1995) 261–272.

[9] S. Friedland, A lower bound for the permanent of a doubly stochastic matrix, *Annals of Mathematics* 110 (1979) 167–176.

[10] H.N. Gabow, Using Euler partitions to edge color bipartite multigraphs, *International Journal of Computer and Information Sciences* 5 (1976) 345–355.

[11] J.M. Hammersley, Existence theorems and Monte Carlo methods for the monomer-dimer problem, in: *Research Papers in Statistics* [Festschrift for J. Neyman] (F.N. David, ed.), Wiley, London, 1966, pp. 125–146.

[12] J. Hopcroft, R.M. Karp, An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs, *SIAM Journal on Computing* 2 (1973) 225–231.

[13] O.H. Ibarra, S. Moran, Deterministic and probabilistic algorithms for maximum bipartite matching via fast matrix multiplication, *Information Processing Letters* 13 (1981) 12–15.

[14] P.W. Kasteleyn, The statistics of dimers on a lattice, I: the number of dimer arrangements on a quadratic lattice, *Physica* 27 (1961) 1209–1225.

[15] P.H. Lundow, Compression of transfer matrices, *Discrete Mathematics* 231 (2001) 321–329.

[16] A. Schrijver, Counting 1-factors in regular bipartite graphs, *Journal of Combinatorial Theory, Series B* 72 (1998) 122–135.

[17] A. Schrijver, Bipartite edge-colouring in $O(\Delta m)$ time, *SIAM Journal on Computing* 28 (1999) 841–846.

[18] A. Schrijver, W.G. Valiant, On lower bounds for permanents, *Indagationes Mathematicae* 42 (1980) 425–427.

[19] R.E. Tarjan, *Data Structures and Network Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1983.

[20] H.N.V. Temperley, M.E. Fisher, Dimer problem in statistical mechanics — an exact result, *Philosophical Magazine* (8) 6 (1961) 1061–1063.

[21] M. Voorhoeve, A lower bound for the permanents of certain $(0,1)$-matrices, *Indagationes Mathematicae* 41 (1979) 83–86.

[22] B.L. van der Waerden, [Aufgabe] 45, *Jahresbericht der Deutschen Mathematiker-Vereinigung* 35 (1926) *117*.