Bipartite edge-colouring in $O(\Delta m)$ time[1]

Alexander Schrijver[2]

Abstract. We show that a minimum edge-colouring of a bipartite graph can be found in $O(\Delta m)$ time, where Δ and m denote the maximum degree and the number of edges of G, respectively. It is equivalent to finding a perfect matching in a k-regular bipartite graph in $O(km)$ time.

By sharpening the methods, a minimum edge-colouring of a bipartite graph can be found in $O((p_{\max}(\Delta) + \log \Delta)m)$ time, where $p_{\max}(\Delta)$ is the largest prime factor of Δ. Moreover, a perfect matching in a k-regular bipartite graph can be found in $O(p_{\max}(k)m)$ time.

1. Introduction

In a classical paper, König [9] showed that the edges of a bipartite graph G can be coloured with $\Delta(G)$ colours, where $\Delta(G)$ is the maximum degree of G. (In this paper, 'colouring' edges presumes that edges that have a vertex in common obtain different colours.)

König's proof is essentially algorithmic, yielding an $O(nm)$ time algorithm (n and m denote the numbers of vertices and edges, respectively, of the graph). As was shown by Gabow [4], the $O(\sqrt{n}m)$ bipartite matching algorithm of Hopcroft and Karp [8] implies an $O(\sqrt{n}m \log \Delta)$ bipartite edge-colouring algorithm. This was improved by Cole and Hopcroft [1] to $O(m \log m)$, by extending methods of Gabow and Kariv [5], [6].

Fixing the maximum degree Δ, the methods found as yet are superlinear (albeit slightly). In this paper we give a linear-time method for fixed or bounded Δ. More precisely, we give an $O(\Delta m)$ method for bipartite edge-colouring. It implies (in fact, is equivalent to) finding a perfect matching in a k-regular bipartite graph in $O(km)$ time.

Ultimately one would hope for an $O(m \log k)$ (or even $O(m)$) algorithm finding a perfect matching in a k-regular bipartite graph, and for an $O(m \log \Delta)$ algorithm for bipartite edge-colouring (the first algorithm implies the second, by a method of Gabow [4] — see below). We did not find such algorithms, although our methods can be extended to obtain some supporting results.

In particular, define, for any natural number k,

$$\phi(k) := \sum_{i=1}^{t} \frac{p_i}{\prod_{j=1}^{i-1} p_j}, \tag{1}$$

where $p_1 \leq \cdots \leq p_t$ are primes with $k = p_1 \cdot \ldots \cdot p_t$. We give an $O((\phi(\Delta) + \log \Delta)m)$ bipartite edge-colouring algorithm. Note that in $\phi(\Delta) + \log \Delta$, the term $\phi(\Delta)$ dominates if Δ is prime, while $\log \Delta$ dominates if Δ is a power of 2. Note also that $\phi(\Delta) \leq 2p_{\max}(\Delta)$, where $p_{\max}(\Delta)$ denotes the largest prime factor in Δ. So fixing the maximum prime factor of Δ, there is an $O(m \log \Delta)$ bipartite edge-colouring algorithm.

Moreover, we give an $O(\phi(k)m)$ algorithm finding a perfect matching in a k-regular bipartite graph. So bounding the maximum prime factor of k, there is a linear-time perfect matching algorithm for k-regular bipartite graphs.

---

[1] *Key Words:* bipartite, edge-colouring, complexity, timetabling, perfect, matching
Subject Classification: Primary: 68R10, 90C27, Secondary: 05C70, 05C85

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

The proof idea is an extension of the following idea of Gabow [4] to find a perfect matching in a 2^t-regular bipartite graph G in linear time. First find an Eulerian orientation of G (taking $O(m)$ time), and consider those edges that are oriented from vertex-colour I to vertex-colour II (in the 2-vertex-colouring of G). This gives a 2^{t-1}-regular subgraph of G. Repeating this, we end up with a 1-regular subgraph of G, being a perfect matching in G. The time is $O(m + \frac{1}{2}m + \frac{1}{4}m + \cdots) = O(m)$.

One can similarly find a 2^t-edge-colouring in $O(tm)$ time. In extending this method to prime factors other than 2 we use some techniques of [10] for estimating the number of perfect matchings and edge-colourings of bipartite graphs.

In this paper, all graphs may have multiple edges.

2. Some practical motivation

As is well-known, bipartite edge-colouring can be applied in timetabling. A pure instance of timetabling consists of a set of teachers, a set of classes, and a list L of pairs (t, c) of a teacher t and a class c, indicating that teacher t has to teach class c during a time-slot (say, an hour) within the time-span of the schedule (say, a week). A pair (t, c) may occur several times in the list, indicating the number of hours the pair t, c should meet weekly.

A timetable then is an assignment of the pairs in the list to hours, from a set H of possible hours, in such a way that no teacher t and no class c occurs in two pairs that are assigned to the same hour. This clearly is a bipartite edge-colouring problem, and by König's theorem, there is a timetable if and only if $|H|$ is not smaller than the number of times that any teacher t or any class c occurs in L. So by the result of Cole and Hopcroft [1] a timetable can be found in $O(|L| \log |L|)$ time, and by our theorem, it can be found also in $O(|H| \cdot |L|)$ time. (In practice, several additional constraints are put on a timetable, making the problem usually NP-complete — cf. Even, Itai, and Shamir [3].)

In many countries, schools are merging, yielding an increase in size, including in numbers of teachers and of classes. So the list L grows. However, the number of hours during a week does not grow. This gives that, in this interpretation, the algorithm is linear in the size of the school.

Moreover, often H is built up from smaller units (say, days), implying that $|H|$ does not have large prime factors. ($|H|$ typically has prime factors 2,3, and 5 only, sometimes 7). This gives that applying the $O(\phi(|H| + \log |H|)|L|)$-time algorithm can be fruitful. Similarly, the method is not very sensitive to doubling or tripling the time-span (say to 2 or 3 weeks).

3. An $O(\Delta m)$ bipartite edge-colouring algorithm

Basic in the edge-colouring algorithm (as in [4]) is a subroutine finding a matching that covers all maximum-degree vertices, and that hence can serve as our first colour. To this end we show:

Theorem 1. *A perfect matching in a k-regular bipartite graph can be found in $O(km)$ time.*

Proof. Let $G = (V, E)$ be a k-regular bipartite graph. For any function $w : E \longrightarrow \mathbb{Z}_+$, let E_w be the set of edges with $w(e) > 0$. For any $F \subseteq E$, denote $w(F) := \sum_{e \in F} w(e)$.

Initially set $w(e) := 1$ for each edge e. Next apply the following iteratively:

(2) Find a circuit C in E_w. Let $C = M \cup N$ for matchings M and N with $w(M) \geq w(N)$. Reset $w := w + \chi^M - \chi^N$.

Note that, at any iteration, the equality $w(\delta(v)) = k$ is maintained for all $v \in V$ (where $\delta(v)$ is the set of edges incident with v).

To see that the process terminates, first note that at any iteration the sum

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

increases by

$$(4) \qquad \sum_{e \in M} ((w(e)+1)^2 - w(e)^2) + \sum_{e \in N} ((w(e)-1)^2 - w(e)^2) = 2w(M) + |M| - 2w(N) + |N|,$$

which is at least $|C|$ (as $w(M) \geq w(N)$). Moreover, (3) is bounded, since $w(e) \leq k$ for each edge e. So the process terminates.

At termination, we have that the set E_w is a forest, and hence is a perfect matching (since $w(e) = k$ for each end edge e of E_w). This implies that at termination the sum (3) is equal to $\frac{1}{2}nk^2 = km$.

Now by depth-first we can find a circuit C in (2) in $O(|C|)$ time on average. Indeed, keep a path P of edges e with $0 < w(e) < k$. Let v be the last vertex of P. Choose an edge $e = vu$ incident with v but not in P. If u does not occur in P, we reset $P := P \cup \{e\}$ and iterate. If u does occur in P, let C be the circuit in $P \cup \{e\}$, and apply (2) to C. Next reset $P := P \setminus C$, and iterate.

If $P = \emptyset$, choose any edge e with $0 < w(e) < k$, and set $P := \{e\}$. If no such edge e exists, we are done. ∎

For k smaller than $\sqrt{\log n}$, the $O(km)$ bound is asymptotically better than the $O(m + n \log n (\log k)^2))$ bound proved by Cole and Hopcroft [1]. (An algorithm related to, but different from, the algorithm described in Theorem 1, was proposed by Csima and Lovász [2], giving an $O(n^2 k \log k)$ time bound.)

By applying a technique of Gabow [4], one can derive from Theorem 1 the following stronger statement:

Corollary 1a. *A k-edge-colouring of a k-regular bipartite graph can be found in $O(km)$ time.*

Proof. If k is odd, first find a perfect matching M, remove M from G, and apply recursion (M will serve as colour).

If k is even, find an Eulerian orientation of G. Let $k' = \frac{1}{2}k$. Then split G into two k'-regular graphs $G_1 = (V, E_1)$ (with E_1 the set of edges oriented from vertex-colour class I to vertex-colour class II) and $G_2 = (V, E_2)$ (with $E_2 := E \setminus E_1$). Find recursively k'-edge-colourings of G_1 and G_2. The union of the two colourings is a k-edge-colouring of G.

The time is bounded as follows. Starting with G, we can find M (if k is odd), find the Eulerian orientation, and split G into G_1 and G_2, in time ckm for some constant c. Then the whole recursion takes time $2ckm$. This can be shown inductively, as $2ckm =$

$ckm + 2ck'm' + 2ck'm'$, where $m' = |E(G_1)| = |E(G_2)| = \frac{1}{2}m$. ∎

This implies the sharper statement:

Corollary 1b. *A $\Delta(G)$-edge-colouring of a bipartite graph $G = (V, E)$ can be found in $O(\Delta(G)m)$ time.*

Proof. Let $k := \Delta(G)$. First iteratively merge any two vertices in the same colour class of G, if each has degree at most $\frac{1}{2}k$. The final graph H will have at most two vertices of degree at most $\frac{1}{2}k$, and moreover, $\Delta(H) = k$ and any k-edge-colouring of H yields a k-edge-colouring of G. Next make a copy H' of H, and join each vertex v of H by $k - d_H(v)$ parallel edges with its copy v' in H' (where $d_H(v)$ is the degree of v in H). This gives the k-regular graph G', with $|E(G')| = O(|E(G)|)$. By Corollary 1a we can find a k-edge-colouring of G' in $O(k|E(G')|)$ time. This gives a k-edge-colouring of H and hence a k-edge-colouring of G. ∎

4. Towards an $O(m \log \Delta)$ method?

The results of Section 3 can be sharpened by using divisibility properties of $\Delta(G)$. First we sharpen Corollary 1a. We repeat the definition of $\phi(k)$ for any natural number k:

$$\phi(k) := \sum_{i=1}^{t} \frac{p_i}{\prod_{j=1}^{i-1} p_j}, \tag{5}$$

where $p_1 \leq \cdots \leq p_t$ are primes with $k = p_1 \cdot \ldots \cdot p_t$.

Theorem 2. *A k-edge-colouring of a k-regular bipartite graph $G = (V, E)$ can be found in $O((\phi(k) + \log k)m)$ time.*

Proof. Let $k = pk'$ with p prime. Split each vertex v into k' new vertices $v_1, \ldots, v_{k'}$, and distribute the edges incident with v over $v_1, \ldots, v_{k'}$ in such a way that each vertex v_i is incident with exactly p edges. This gives the p-regular graph $\tilde{G}$. Find a p-edge-colouring of $\tilde{G}$. The colours give a partition of E into classes $E_1, \ldots, E_p$, in such a way that each graph $G_j = (V, E_j)$ is k'-regular. Next find a k'-edge-colouring of G_p, yielding perfect matchings $M_1, \ldots, M_{k'}$.

Now we apply the following iteratively. We have a partition of E into perfect matchings $M_1, \ldots, M_{\alpha k'}$ and k'-regular graphs $E_1, \ldots, E_{p-\alpha}$. (Initially, $\alpha = 1$.) Let $q := \min\{\alpha, p - \alpha\}$. Choose r such that $qk' + r$ is a power of 2 and such that $r \leq qk'$. Let $E' := M_1 \cup \cdots \cup M_r \cup E_1 \cup \cdots \cup E_q$. Then $G' := (V, E')$ is a $qk' + r$-regular graph. Next $qk' + r$-edge-colour G', yielding colours $N_1, \ldots, N_{qk'+r}$. Now replace $M_1, \ldots, M_r$ by $N_1, \ldots, N_{qk'+r}$ and $E_1, \ldots, E_{p-\alpha}$ by $E_{q+1}, \ldots, E_{p-\alpha}$ and iterate. We stop if $\alpha = p$.

So at any iteration, α is replaced by $\alpha + q$. Moreover, at any iteration except possibly the last iteration, we have $q = \alpha$. So at any iteration except possibly the last one, q is twice as large as at the previous iteration.

By [4], the work in the iteration takes time $O(|E'|\log(qk'+r)) = O(|E'|\log k)$, since $qk'+r$ is a power of 2 and since $qk'+r \leq k$. Since $|E'| = \frac{1}{2}(qk'+r)n \leq qk'n$, over all iterations the work is $O((1+2+2^2+\cdots+2^{\log p})k'n\log k) = O(pk'n\log k) = O(m\log k)$.

To this time bound we must add the time needed to obtain $G_1,\ldots,G_p$ which takes $O(pm)$ time by Corollary 1b, since it amounts to p-edge-colouring the p-regular graph $\tilde{G}$, having m edges, and the time needed to edge-colour G_p, which takes (by induction) $O((\phi(k')+\log k')m')$ time, where $m' = m/p$ is the number of edges of G_p. Since $\phi(k) = p+\phi(k')/p$, we have the required time bound. ∎

This gives:

Corollary 2a. *A $\Delta(G)$-edge-colouring of a bipartite graph G can be found in $O((\phi(\Delta(G))+\log\Delta(G))m)$ time.*

Proof. Directly from Theorem 2 by the method of Corollary 1b. ∎

Note that

(6) $\qquad \phi(k) \leq 2p_{\max}(k)$

(where $p_{\max}(k)$ is the largest prime factor of k). This follows inductively, since if $k = pk'$, with p the smallest prime factor of k, then $\phi(k) = p+\phi(k')/p \leq p_{\max}(k) + (2p_{\max}(k')/p) \leq 2p_{\max}(k)$. This implies:

Corollary 2b. *A $\Delta(G)$-edge-colouring of a bipartite graph G can be found in $O((p_{\max}(\Delta(G))+\log\Delta(G))m)$ time.*

Proof. Directly from Corollary 2a with (6). ∎

Note that in performing this method one does not need to apply deep number-theoretic algorithms to find the prime-factorization of k. Indeed, the factors $p_1,\ldots,p_t$ can be found in $O(\phi(k)k)$ time, since the smallest prime factor p can be found in time $O(pk)$, by trying $i = 2,3,\ldots$ as divisor of k (for each i taking $O(k)$ time), until we reach p. Next we can apply recursion to $k' := k/p$, taking recursively $O(\phi(k')k')$ time. This gives $O(\phi(k)k)$ time over-all, since $\phi(k) = p+\phi(k')/p$.

A sharpening can be obtained also for finding perfect matchings in k-regular bipartite graphs.

Theorem 3. *A perfect matching in a k-regular bipartite graph G can be found in time $O(\phi(k)m)$ time.*

Proof. Write $k = pk'$ with p the smallest prime factor of k. Make the graph $\tilde{G}$ as in the proof of Theorem 2. So $\tilde{G}$ is p-regular. Find a perfect matching M in $\tilde{G}$. It gives a k'-regular subgraph $G' = (V,E')$ of G. In G' we find recursively a perfect matching.

Finding perfect matching M in $\tilde{G}$ takes time $O(pm)$ by Theorem 1. Finding matching N in G' takes time $O(\phi(k')m/p)$ by induction (as G' is k'-regular and has m/p edges). Since

$\phi(k) = p + \phi(k')/p$, the whole process takes $O(\phi(k)m)$ time. ∎

Corollary 3a. *A matching covering all maximum-degree vertices in a bipartite graph can be found in $O(\phi(\Delta)m)$ time.*

Proof. Directly from Theorem 3, using the technique of Corollary 1b. ∎

By (6), Theorem 3 can be stated in a weaker form as:

Corollary 3b. *A perfect matching in a k-regular bipartite graph can be found in $O(p_{\max}(k)m)$ time.*

Proof. Directly from Theorem 3, using (6). ∎

5. Some open questions

It would be surprising if divisibility properties of the maximum degree $\Delta(G)$ of a bipartite graph G would determine the complexity of edge-colouring G. Our results are blocked however by the primes. If $\Delta(G)$ is a prime, we do not have anything better than an $O(\Delta(G)m)$-time algorithm. So the main problem is to 'break' a prime. More precisely,

(7) Is there an $O(m \log k)$ algorithm for finding a perfect matching in a k-regular bipartite graph?

The method of Cole and Hopcroft [1] gives an $O(m + n \log n \log^2 k)$ algorithm to find a perfect matching in any k-regular bipartite graph. If there would be an $O(m \log k)$ perfect matching algorithm for k-regular bipartite graphs, there exists an $O(m \log \Delta)$ bipartite edge-colouring algorithm (by methods like in Theorem 2 above), thus answering our second question:

(8) Is there an $O(m \log \Delta)$ algorithm for bipartite edge-colouring?

Similar methods as used for proving Theorem 2 give an approximative method, namely a bipartite $(\Delta + \lfloor \log(\Delta - 1) \rfloor)$-edge-colouring algorithm, with time bound $O(m \log \Delta)$. Indeed, let $G = (V, E)$ be a bipartite graph of maximum degree Δ. In $O(m)$ time we can split E into E' and E'' such that both $G' = (V, E')$ and $G'' = (V, E'')$ have maximum degree at most $\Delta' := \lceil \frac{1}{2}\Delta \rceil$. We may assume that $|E'| \leq \frac{1}{2}m$. Let $t := \Delta' + \lfloor \log(\Delta' - 1) \rfloor$. Then t-edge-colour G' recursively, giving colours $M_1, \ldots, M_t$. Choose $s \leq t$ such that $\Delta' + s$ is a power of 2. Next $(\Delta' + s)$-edge-colour the graph H made by $M_1 \cup \cdots \cup M_s \cup E''$. With the remaining $M_{s+1}, \ldots, M_t$ it gives an edge-colouring of G with

(9) $$(\Delta' + s) + (t - s) = 2\Delta' + \lfloor \log(\Delta' - 1) \rfloor \leq \Delta + \lfloor \log(\Delta - 1) \rfloor$$

colours. Since the number of edges in G' is at most $\frac{1}{2}m$ and since edge-colouring H takes $O(m \log(\Delta' + s)) = O(m \log \Delta)$ time, this gives an $O(m \log \Delta)$ time bound.

The nonbipartite case is NP-complete, by the well-known result of Holyer [7]: it is NP-complete to decide if a 3-regular graph can be 3-edge-coloured. However, it is not difficult to see that a 3-regular graph can be 4-edge-coloured in *linear* time. Actually, any graph of maximum degree 3 can be 4-edge-coloured in $O(m)$ time.

By Vizing's theorem, each simple graph G can be $(\Delta(G)+1)$-edge-coloured. (If $\Delta(G) \leq 3$ we can delete the condition that G be simple.) This prompts the question:

(10) Is there an $O(\Delta m)$-time $(\Delta+1)$-edge-colouring algorithm for simple graphs?

Of course, the stronger question is to ask for an $O(m \log \Delta)$ algorithm.

References

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

[2] J. Csima, L. Lovász, A matching algorithm for regular bipartite graphs, *Discrete Applied Mathematics* 35 (1992) 197–203.

[3] S. Even, A. Itai, A. Shamir, On the complexity of timetable and multicommodity flow problems, *SIAM Journal on Computing* 5 (1976) 691–703.

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

[5] H.N. Gabow, O. Kariv, Algorithms for edge coloring bipartite graphs, in: *Conference Record of the Tenth Annual ACM Symposium on Theory of Computing* [10th STOC, San Diego, California, May 1–3, 1978], The Association for Computing Machinery, New York, 1978, pp. 184–192.

[6] H.N. Gabow, O. Kariv, Algorithms for edge coloring bipartite graphs and multigraphs, *SIAM Journal on Computing* 11 (1982) 117–129.

[7] I.G. Holyer, The NP-completeness of edge-colouring, *SIAM Journal on Computing* 10 (1981) 718–720.

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

[9] D. König, Graphok és alkalmazásuk a determinánsok és a halmazok elméletére [Hungarian], *Mathematikai és Természettudományi Értesitö* 34 (1916) 104–119 [German translation: Über Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre, *Mathematische Annalen* 77 (1916) 453–465].

[10] A. Schrijver, On the number of edge-colourings of regular bipartite graphs, *Discrete Mathematics* 38 (1982) 297–301.