正則２部グラフに対する単純なマッチングアルゴリズム

牧野和久, 高畑貴志, 藤重悟

560-8531 大阪府豊中市待兼山町 1-3 大阪大学大学院基礎工学研究科システム科学分野
{makino, takabatake, fujishig}@sys.es.osaka-u.ac.jp

あらまし　本論文では，Δ-正則２部グラフに対する完全マッチング問題を考察する．ただし，グラフ G は，n 節点，m 枝，すなわち，$\frac{1}{2}n\,\Delta = m$ とする．我々は，まず，Gabow の方法に基づく新しい単純な $\mathrm{O}(m \log n)$ アルゴリズムを与える．次に，Cole と Hopcroft が提案した正則２部グラフに対する辺疎化手法を取り入れることにより， そのアルゴリズムを $\mathrm{O}(m + n \log n \log \Delta)$ に改善する．我々のアルゴリズムは，動的木やスプレイ木などの高度なデータ構造を必要としない.

和文キーワード:　２部マッチング, 辺彩色, グラフアルゴリズム

A Simple Matching Algorithm for Regular Bipartite Graphs

Kazuhisa Makino, Takashi Takabatake and Satoru Fujishige

Division of Systems Science, Graduate School of Engineering Science, Osaka University,
Toyonaka, Osaka, 560-8531, Japan.
{makino, takabatake, fujishig}@sys.es.osaka-u.ac.jp

abstract　We consider the perfect matching problem for a Δ-regular bipartite graph with n vertices and m edges, i.e., $\frac{1}{2}n\,\Delta = m$. We first give a new simple $\mathrm{O}(m \log n)$ algorithm based on Gabow's approach, and then improve it to a faster $\mathrm{O}(m+n \log n \log \Delta)$ algorithm by incorporating Cole and Hopcroft's edge-sparsification for regular bipartite graphs. Our algorithms employ no sophisticated data structure such as dynamic tree and splay tree.

英文 key words:　Bipartite matching, Edge-coloring, Graph algorithm.

1. Introduction

It is well-known that any regular bipartite graph has a perfect matching and that the perfect matching problem for regular bipartite graphs is related to the edge-coloring problem for general bipartite graphs. For example, Kapoor and Rizzi [6] showed that the edge-coloring problem for a bipartite graph G with m edges and the maximum degree Δ can be solved in $T + \mathrm{O}(m \log \Delta)$ time, where T is the time required to compute a perfect matching in a k-regular bipartite graph with $\mathrm{O}(m)$ edges and $k \leq \Delta$.

In this paper, we consider the perfect matching problem for regular bipartite graphs with possible multiple edges. The perfect matching problem for regular bipartite graphs has been studied and a lot of algorithms have been proposed in the literature (see, e.g., [1, 2, 3, 4, 5, 7, 8]). Cole and Hopcroft [2] presented an $\mathrm{O}(m + n \log n \log^2 \Delta)$ algorithm for computing a perfect matching in a Δ-regular bipartite graph with n vertices and m edges. This time complexity was improved by Cole [1] and Rizzi [7] to $\mathrm{O}(m+n \log n \log \Delta)$. Schrijver [8] also presented an $\mathrm{O}(m\Delta)$ algorithm and Cole, Ost, and Schirra [3] obtained an $\mathrm{O}(m)$ algorithm by improving Schrijver's algorithm [8] by using splay trees, one for each chain, as a data structure.

We present in this paper a simple $\mathrm{O}(m+n \log n \log \Delta)$ algorithm for the perfect matching problem for Δ-regular bipartite graphs. Our algorithm can be obtained by extending Gabow's algorithm [5] for computing a perfect matching in a 2^t-regular bipartite graph for a positive integer t. Our algorithm uses no sophisticated data structure such as dynamic tree and splay tree employed in [1, 3], and runs in linear time for regular bipartite graphs with $\Delta \geq \log n \log \Delta$. It is expected that our algorithm will achieve good practical performance, while efficient implementations and computational experiments of the above algorithms (including ours) deserve further study. Note that Cole's algorithm [1] makes use of a dynamic tree and that our algorithm is simpler than Rizzi's [7].

The rest of the paper is organized as follows. In Section 2 we first present an $\mathrm{O}(m \log n)$ algorithm for computing a perfect matching in a regular bipartite graph, which will give us a basis for a faster algorithm. By using the edge-sparsification technique due to Cole and Hopcroft [2], we present a faster algorithm for computing a perfect matching in a regular bipartite graph in Section 3.

2. An $\mathrm{O}(m \log n)$ algorithm

In this section, we present an $\mathrm{O}(m \log n)$ algorithm for computing a perfect matching in a regular bipartite graph, which will be made faster in the next section. Let $G = (V, E)$ be a Δ-regular bipartite graph with n vertices and m edges. Here V has a bipartition

(V^+, V^-), i.e., any edge $e \in E$ is incident to a vertex in V^+ and a vertex in V^-. Note that $m = n\,\Delta/2$ and $|V^+| = |V^-| = n/2$.

Let us first note that a perfect matching in a 2^t-regular bipartite graph G with a positive integer t can be computed in linear time [5]. We first find an Eulerian orientation of G that consists of Eulerian tours, one for each connected component of G, and then remove those edges in G that are oriented from V^- to V^+. This gives a 2^{t-1}-regular subgraph of G. By repeating this procedure t times, we finally obtain a 1-regular subgraph (i.e., a perfect matching) of G. Since an Eulerian orientation of G can be found in $\mathrm{O}(m)$ time, Gabow's algorithm [5] requires $\mathrm{O}(m + \frac{1}{2}m + \frac{1}{4}m + \cdots) = \mathrm{O}(m)$ time. However, if a given regular graph is not 2^t-regular for any positive integer t, the above algorithm does not work, since it ends up with a $(2k+1)$-regular subgraph for some integer $k \geq 1$ that has no Eulerian orientation.

Therefore, our algorithm first makes G 2^t-regular by adding new edges to G. More precisely, let $G = (V, E)$ be a Δ-regular bipartite graph such that $2^{t-1} < \Delta \leq 2^t$ for some positive integer t. Let M_1 be a perfect matching of the complete bipartite graph $\mathrm{K}_{\frac{n}{2},\frac{n}{2}}$ with the bipartition (V^+, V^-) of V. We first construct a 2^t-regular bipartite graph $\hat{G} = (V, \hat{E})$ by adding $2^t - \Delta$ copies of M_1 to G. A new edge $\hat{e} \in \hat{E} \setminus E$ is called *dummy* if there exists no edge e in E such that e and $\hat{e}$ are parallel. Note that the number of dummy edges in $\hat{G}$ is at most $\frac{n}{2}\,(2^t - \Delta) < \frac{|\hat{E}|}{2}$. We then apply Gabow's algorithm [5] to $\hat{G}$ to find a perfect matching M_2 in $\hat{G}$. Here, for an Eulerian orientation, if the number of dummy edges oriented from V^+ to V^- is at most the half of the number of all dummy edges, to get M_2 we remove those edges that are oriented from V^- to V^+; otherwise we remove those edges that are oriented from V^+ to V^-. Note that M_2 has at most $\frac{n}{4}\,(= \frac{|V^+|}{2})$ dummy edges. In other words, the size of the matching formed by the non-dummy edges in M_2 is at least $|V^+| - \frac{|V^+|}{2}\,(= \frac{|V^+|}{2})$. We again construct a 2^t-regular bipartite graph $\hat{G}$ by adding $2^t - \Delta$ copies of M_2 to G, and apply Gabow's algorithm to it. Let M_3 be the obtained matching in $\hat{G}$. Since $\hat{G}$ contains at most $\frac{n}{4}\,(2^t - \Delta) < \frac{|\hat{E}|}{4}$ dummy edges, M_3 has at most $\frac{|V^+|}{4}$ dummy edges (i.e., the size of the matching formed by non-dummy edges in M_3 is at least $|V^+| - \frac{|V^+|}{4}\,(= \frac{3|V^+|}{4})$). By repeating this procedure at most $\lceil \log |V^+| \rceil$ times, we finally obtain a perfect matching of G.

Formally, the algorithm described above can be given as follows.

Algorithm ADD-SPLIT

Input: A Δ-regular bipartite graph $G = (V, E)$ such that $2^{t-1} < \Delta \leq 2^t$ for some positive integer t.

Output: A perfect matching M in G.

Step 1: Compute a perfect matching M of $\mathrm{K}_{\frac{n}{2},\frac{n}{2}}$ and put $k := 1$.

Step 2: Construct a 2^t-regular bipartite graph $\hat{G} = (V, \hat{E})$ by adding $2^t - \Delta$ copies of M to G.

Step 3: While $\hat{G}$ is not 1-regular do

(3-I) Find an Eulerian orientation of $\hat{G}$.

(3-II) If the number of dummy edges oriented from V^+ to V^- is at most the half of the number of all dummy edges in $\hat{G}$, then remove those edges that are oriented from V^- to V^+; otherwise remove those edges that are oriented from V^+ to V^-. Denote the resultant graph by $\hat{G} = (V, \hat{E})$ again.

Step 4: Put $M := \hat{E}$ and $k := k + 1$. If $k \leq \lceil \log |V^+| \rceil$, then go to Step 2. Otherwise return M and halt. □

Note that when M in Step 4 contains no dummy edge before we get $k > \lceil \log |V^+| \rceil$, we can output M and halt.

Theorem 2.1: *Algorithm* ADD-SPLIT *correctly computes a perfect matching of* G *in* $\mathrm{O}(m \log n)$ *time.*

Proof. Since the above argument shows the correctness of the algorithm, we consider its time complexity. It is clear that Steps 1, 2 and 4 require $\mathrm{O}(m)$ time. From the result in [5], Step 3 can be done in $\mathrm{O}(m)$ time. Since the number of iterations between Step 2 and Step 4 is $\lceil \log n \rceil$, the algorithm requires $\mathrm{O}(m \log n)$ time in total. □

In concluding this section, we remark that the time complexity can be improved to $\mathrm{O}(m \log_\Delta n)$ by effectively using the information on $\hat{G}$ obtained in the previous iteration between Step 2 and Step 4.

3. An $\mathrm{O}(m + n \log n \log \Delta)$ algorithm

An edge-sparsification technique for regular bipartite graphs was proposed by Cole and Hopcroft [2] and has been used to obtain faster algorithms for computing a perfect matching in a regular bipartite graph [1, 2, 3, 7]. We also employ this technique to devise a faster version of our algorithm.

Given a Δ-regular bipartite graph $G = (V, E)$ with $2^{t-1} < \Delta \leq 2^t$ for some positive integer t, the *edge-sparsification* produces a subgraph $G^* = (V, E^*)$ of G having an edge capacity function $c : E^* \to \{1, 2, 2^2, \cdots, 2^t\}$ such that

(i) for each $v \in V$, the sum of the edge capacities $c(e)$ for all edges e incident to v is equal to Δ, i.e.,

$$\sum \{\, c(e) \mid e \in E^* \text{ is incident to } v \} \quad = \quad \Delta,$$

(ii) for each $\ell \in \{0, 1, \cdots, t\}$, the set of all edges e with $c(e) = 2^{\ell}$ forms a forest (i.e., it contains no cycle).

Cole and Hopcroft [2] showed that the edge-sparsification can be done in linear time without using any sophisticated data structure. We identify an edge e having capacity $c(e)$ with parallel edges formed by $c(e)$ copies of e. It follows from (i) that G^* can be regarded as a Δ-regular graph, and hence it always contains a perfect matching which can also be regarded as a perfect matching in the original graph G. By (ii), G^* has at most $(n-1)\lceil \log \Delta \rceil$ edges.

Intuitively speaking, we apply Algorithm ADD-SPLIT to graph G^* instead of G. Since G^* has at most $(n-1)\lceil \log \Delta \rceil$ edges, the algorithm requires $\mathrm{O}(m + (n-1)\log \Delta \times \log n) = \mathrm{O}(m + n \log n \log \Delta)$ time, where the time required for the edge-sparsification is $\mathrm{O}(m)$.

Let us assume that $2^{t-1} < \Delta < 2^t$ for some positive integer t, since a perfect matching of a 2^t-regular bipartite graph can be obtained in linear time [5]. We need some notations to describe the details of our faster algorithm. Apply the edge-sparsification procedure to a graph formed by $(2^t - \Delta)\,(\leq 2^{t-1})$ copies of a perfect matching M in $\mathrm{K}_{\frac{n}{2},\frac{n}{2}}$. Denote by M^* the resultant graph with edge capacities. Let $\hat{G} = (V, \hat{E})$ be the graph obtained by adding M^* to G^*, and let $\hat{E}_\ell$ $(\ell = 0, 1, \cdots, t-1)$ be the set of all edges e in $\hat{E}$ with $c(e) = 2^\ell$. Note that $\hat{G}$ with edge capacities can be regarded as a 2^t-regular graph. A 2^{t-1}-regular subgraph $H = (V, F)$ of $\hat{G}$ can be computed as follows.

(1) Compute an Eulerian orientation of $\hat{E}_0$ and choose those edges that are oriented from V^+ to V^- (or V^- to V^+). Denote by R_1 the set of such edges.

(2) For each $\ell \in \{0, 1, \cdots, t-2\}$ define

$$F_\ell = \begin{cases} R_1 \cup \hat{E}_1 & \text{if } \ell = 0 \\ \hat{E}_{\ell+1} & \text{otherwise,} \end{cases} \tag{3.1}$$

and put $F = \bigcup_{\ell=0}^{t-2} F_\ell$.

Note that each edge $e \in F_\ell$ $(\ell = 0, 1, \cdots, t-2)$ has the capacity $c(e) = 2^\ell$. Therefore, Step 3 in Algorithm ADD-SPLIT can be performed as follows. Let $R_0 = \emptyset$ and for each $\ell = 1, 2, \cdots, t-1$ let R_ℓ be the edge set obtained from an Eulerian orientation of $\hat{E}_{\ell-1} \cup R_{\ell-1}$. Then we get a perfect matching of $\hat{G}$ from an Eulerian orientation of $\hat{E}_{t-1} \cup R_{t-1}$.

Now, we have the following algorithm.

Algorithm BITWISE-ADD-SPLIT

Input: A Δ-regular bipartite graph $G = (V, E)$ such that $2^{t-1} < \Delta \leq 2^t$ for some positive integer t.

Output: A perfect matching M in G.

Step 0: If $\Delta = 2^t$, then compute a perfect matching M of G by applying Gabow's algorithm to G, and halt.

Step 1: Apply the edge-sparsification procedure to G. Denote by G^* the resultant graph with edge capacities. Let M be a perfect matching of $\mathrm{K}_{\frac{n}{2},\frac{n}{2}}$ and put $k := 1$.

Step 2: Let M^* be the graph, with edge capacities, obtained from $2^t - \Delta$ copies of M by the edge-sparsification procedure and construct a 2^t-regular bipartite graph $\hat{G} = (V, \hat{E})$ by adding M^* to G^*.

Step 3: Put $R_0 := \emptyset$. For $\ell = 0, 1, \cdots, t-1$ do

- **(3-I)** Find an Eulerian orientation of $\hat{E}_\ell \cup R_\ell$.
- **(3-II)** If the number of dummy edges oriented from V^+ to V^- is at most the half of the number of all dummy edges, then remove those edges that are oriented from V^- to V^+; otherwise remove those edges that are oriented from V^+ to V^-. Denote the resultant edge set by $R_{\ell+1}$.

Step 4: Put $M := R_t$ and $k := k+1$. If $k \leq \lceil \log |V^+| \rceil$, then go to Step 2. Otherwise return M and halt. □

Similarly as in algorithm ADD-SPLIT, if M in Step 4 contains no dummy edge before we get $k > \lceil \log |V^+| \rceil$, we can output M and halt. Note further that in Step 2, the edge-sparsification for $2^t - \Delta$ copies of M can be done in $\mathrm{O}(n \log \Delta)$ time by considering the binary representation of $2^t - \Delta$.

Let us examine the properties of $\hat{E}_\ell \cup R_\ell$ ($\ell = 0, 1, \cdots, t-1$) in Step 3 of the algorithm.

Lemma 3.1: *Define $H_\ell = (V, \hat{E}_\ell \cup R_\ell)$ for $\ell = 0, 1, \cdots, t-1$. Then the following two statements hold for $\ell = 0, 1, \cdots, t-1$:*

(i) *The degree of each vertex in H_ℓ is even.*

(ii) *H_ℓ has at most $3n$ edges.*

Proof. Since $\hat{G}$ is 2^t-regular, the first statement holds. The second one is shown by induction on ℓ, since $\hat{E}_\ell$ and R_ℓ have at most $(n-1) + n/2 \leq 3n/2$ and $3n/2$ edges, respectively. □

Theorem 3.2: *Algorithm* BITWISE-ADD-SPLIT *finds a perfect matching of a Δ-regular bipartite graph G in $\mathrm{O}(m + n \log n \log \Delta)$ time.*

Proof. Since the discussion in Sections 2 and 3 shows the correctness of the algorithm, we only consider its time complexity. Steps 0 and 1 can be done in $O(m)$ time [5, 2], and Step 2 requires $O(n \log \Delta)$ time, since $|E^*|, |M^*| \le (n-1)\log \Delta$. It follows from Lemma 3.1 that Step 3 requires $O(n \log \Delta)$ time. Moreover, Step 4 requires $O(n)$ time. Since the number of iterations between Step 2 and Step 4 is $\lceil \log n \rceil$, the algorithm requires $O(m + n \log n \log \Delta)$ time in total. □

By combining it with the result by Kapoor and Rizzi [6], we have the following corollary.

Corollary 3.3: *A minimum edge-coloring of a Δ-bipartite graph can be found in* $O((m + n \log n) \log \Delta)$ *time.* □

Acknowledgments

This work was supported in part by Grants-in-Aid for Scientific Research of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] R. Cole. *Two Problems in Graph Theory.* Ph. D. thesis. Cornell University, August 1982.

[2] R. Cole and J. Hopcroft. On edge coloring bipartite graphs. *SIAM Journal on Computing*, 11: 540–546, 1982.

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

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

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

[6] A. Kapoor and R. Rizzi. Edge-coloring bipartite graphs. *Journal of Algorithms*, 34: 390–396, 2000.

[7] R. Rizzi. Finding 1-factors in bipartite regular graphs, and edge-coloring bipartite graphs. Preprint, October 1999.

[8] A. Schrijver. Bipartite edge coloring in O(Δm) time. *SIAM Journal on Computing*, 28: 841–846, 1998.