J Comb Optim (2007) 13:103–122
DOI 10.1007/s10878-006-9012-x

On Ring Grooming in optical networks

Gruia Călinescu* · **Peng-Jun Wan**

Published online: 3 November 2006

Abstract An instance I of Ring Grooming consists of m sets $A_1, A_2, \ldots, A_m$ from the universe $\{0, 1, \ldots, n-1\}$ and an integer $g \geq 2$. The unrestricted variant of Ring Grooming, referred to as Unrestricted Ring Grooming, seeks a partition $\{P_1, P_2, \ldots, P_k\}$ of $\{1, 2, \ldots, m\}$ such that $|P_i| \leq g$ for each $1 \leq i \leq k$ and $\sum_{i=1}^{k} |\bigcup_{r \in P_i} A_r|$ is minimized. The restricted variant of Ring Grooming, referred to as Restricted Ring Grooming, seeks a partition $\{P_1, P_2, \ldots, P_{\lceil \frac{m}{g} \rceil}\}$ of $\{1, 2, \ldots, m\}$ such that $|P_i| \leq g$ for each $1 \leq i \leq \lceil \frac{m}{g} \rceil$ and $\sum_{i=1}^{k} |\bigcup_{r \in P_i} A_r|$ is minimized. If $g = 2$, we provide an optimal polynomial-time algorithm for both variants. If $g > 2$, we prove that both both variants are NP-hard even with fixed g. When g is a power of two, we propose an approximation algorithm called iterative matching. Its approximation ratio is exactly 1.5 when $g = 4$, at most 2.5 when $g = 8$, and at most $\frac{g}{2}$ in general while it is conjectured to be at most $\frac{g}{4} + \frac{1}{2}$. The iterative matching algorithm is also extended for Unrestricted Ring Grooming with arbitrary g, and a loose upper bound on its approximation ratio is $\lceil \frac{g}{2} \rceil$. In addition, set-cover based approximation algorithms have been proposed for both Unrestricted Ring Grooming and Restricted Ring Grooming. They have approximation ratios of at most $1 + \log g$, but running time in polynomial of m^g.

Keywords Ring grooming · Approximation algorithms · Matching

1 Introduction

Ring Grooming is a partition problem motivated by minimizing the expensive SONET add-drop multiplexers in optical networks (Wan et al., Zhang and Qiao, 1998). An instance I of Ring Grooming consists of m sets $A_1, A_2, \ldots, A_m$ from the universe $\{0, 1, \ldots, n-1\}$ and an integer $g \geq 2$. The cost of a partition $\{P_1, P_2, \ldots, P_k\}$ of $\{1, 2, \ldots, m\}$ is given by

*Work supported by a DIMACS postdoctoral fellowship.

G. Călinescu (✉) · P.-J. Wan
Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616
e-mail: calinescu@iit.edu

P.-J. Wan
e-mail: wan@cs.iit.edu

$\sum_{i=1}^{k} |\bigcup_{r\in P_i} A_r|$. The unrestricted variant of Ring Grooming, referred to as Unrestricted Ring Grooming, seeks a minimum-cost partition of $\{1, 2, \ldots, m\}$ into groups of sizes at most g. The restricted variant of Ring Grooming, referred to as Restricted Ring Grooming, seeks a minimum-cost partition of $\{1, 2, \ldots, m\}$ into $\lceil \frac{m}{g} \rceil$ groups of sizes at most g.

In this paper, we first study the computational complexity of Ring Grooming. When $g = 2$ both variants of Ring Grooming are shown to be solvable in polynomial time, and an optimal algorithm based on matching is provided. When $g > 2$, both variants of Ring Grooming are proved to be NP-hard even if g is a fixed constant. Thus, we will seek polynomial-time approximation algorithms for the two variants of Ring Grooming. We establish the following general relation between approximalities of the two variants: Any α-approximation algorithm for Restricted Ring Grooming can lead to an α-approximation algorithm for Unrestricted Ring Grooming; and any α-approximation algorithm for Unrestricted Ring Grooming is also a 2α-approximation algorithm for Restricted Ring Grooming. When g is a power of two, an iterative matching algorithm is proposed for Restricted Ring Grooming. Its approximation ratio is exactly 1.5 when $g = 4$, at most 2.5 when $g = 8$, and at most $\frac{g}{2}$ in general while it is conjectured to be at most $\frac{g}{4} + \frac{1}{2}$. The iterative matching algorithm is also extended for Unrestricted Ring Grooming with arbitrary g, and a loose upper bound on its approximation ratio is $\lceil \frac{g}{2} \rceil$. In addition, set-cover based approximation algorithms have been proposed for both Unrestricted Ring Grooming and Restricted Ring Grooming. They have approximation ratios of at most $1 + \log g$, but running time in polynomial of m^g.

In the remaining of this section, we introduce some terms and notations. Consider an instance I of Ring Grooming given by m sets $A_1, A_2, \ldots, A_m$ from the universe $\{0, 1, \ldots, n-1\}$ and a constant integer parameter $g > 1$. Let P be any subset of $\{1, 2, \ldots, m\}$. It is called as a d-group if $|P| = d$. The cost of P is given by $|\bigcup_{r\in P} A_r|$, and the savings of P is given by $\sum_{r\in P} |A_r| - |\bigcup_{r\in P} A_r|$. Let $\pi = \{P_1, P_2, \ldots, P_k\}$ be any partition of the indices $\{1, 2, \ldots, m\}$. It is called as a d-grouping if $|P_i| = d$ for all $1 \le i \le k$. The cost of π is sum of the costs of all groups in π, i.e., $\sum_{i=1}^{k} |\bigcup_{r\in P_i} A_r|$, and the savings of π is the sum of the savings of all groups in π, i.e.,

$$\sum_{i=1}^{k} \left(\sum_{r\in P_i} |A_r| - \left| \bigcup_{r\in P_i} A_r \right| \right) = \sum_{r=1}^{m} |A_r| - \sum_{i=1}^{k} \left| \bigcup_{r\in P_i} A_r \right|.$$

Let G be any graph. We use $V(G)$ and $E(G)$ to denote the vertex set and the edge set of G respectively. For any $v \in V(G)$, its degree is denoted by $\deg_G(v)$. For any $U \subseteq V(G)$, let $G[U]$ denote the subgraph of G induced by U, $E_G(U)$ denote the set of edges with both endpoints in U, and $\Gamma_G(U)$ denote the *cut* between U and $V(G)\backslash U$, i.e., the set of edges with exactly one endpoint in U. A cut containing c edges is referred to as a c-cut. For any two vertices u and v of G, the *connectivity* between u and v, denote by $c_G(u, v)$ is the maximum number of edge-disjoint paths in G from u to v. In all these notations, the subscript G may be omitted if it can be understood from the context.

2 Computational complexities

When $g = 2$, simple polynomial-time algorithm exists for Restricted Ring Grooming, which also outputs an optimal solution to Unrestricted Ring Grooming. By adding one dummy empty set if necessary, we assume that m is even. Then any solution to Restricted Ring Grooming is a *perfect* matching of $\{1, 2, \ldots, m\}$ and vice versa. To establish the relation

between Ring Grooming and matching, we define a weighted complete graph K_m with vertex set $\{1, 2, \ldots, m\}$ by setting the weight of any edge $e = (i, j)$ equal to $w(e) = |A_i \cap A_j|$. Then the weight of the any perfect matching is equal to its savings as a solution to Restricted Ring Grooming. Thus a maximum-weighted perfect matching gives rise to an optimal solution to Ring Grooming, Restricted or Unrestricted.

However when $g > 2$, both Unrestricted Ring Grooming and Restricted Ring Grooming are NP-hard, as shown in the next lemma.

Lemma 1. *For any* fixed *constant $g > 2$, both Unrestricted Ring Grooming and Restricted Ring Grooming are NP-hard.*

Proof: The reduction is made from Partition Into Isomorphic Subgraphs (Garey and Johnson, 1979). Partition Into Isomorphic Subgraphs is the following problem: given two graphs G and H with $|V(G)| = r|V(H)|$ for some positive integer r, determine whether there is a partition of $V(G)$ into r disjoint sets $V_1, V_2, \ldots, V_r$ such that for all $1 \le i \le r$, $G[V_i]$ is isomorphic to H. The problem remains NP-Complete for any fixed H with at least three vertices, even if the maximum degree of G is bounded by a constant depending on $|V(H)|$ (Kann, 1994). We are going to use as H the clique with g vertices to prove the NP-hardness of both Restricted Ring Grooming and Restricted Ring Grooming for any fixed $g \ge 3$, even if all sets are bounded by a constant depending on g.

Given a graph G with $|V(G)| = rg$, construct an instance of the Ring Grooming as follows: the universe $\{0, 1, \ldots, n-1\}$ is $E(G)$, and the sets $\{A_v \mid v \in V(G)\}$ are defined by $A_v = \{e \in E(G) \mid e \text{ incident to } v\}$, for all $v \in V(G)$. For any $U \subseteq V(G)$, let A_U denote the group of sets $\{A_v \mid v \in U\}$. Then the total savings of the group A_U is $|E(G[U])|$ since each edge of $G[U]$ gives a savings of one. Notice that $|E(G[U])| \le \frac{|U|(|U|-1)}{2}$, and the equality holds if and only $G[U]$ is a clique.

Consider any partition of $V(G)$ into k disjoint sets $\pi = \{V_1, V_2, \ldots, V_k\}$ with $|V_i| \le g$ for all $1 \le i \le k$. Then the total savings of π is at most $\sum_{i=1}^{k} \frac{|V_i|(|V_i|-1)}{2} = \frac{1}{2}\sum_{i=1}^{k} |V_i|^2 - \frac{1}{2}|V(G)|$, and it is achievable if and only if for all $1 \le i \le k$ the graph $G[V_i]$ is a clique. As $|V_i| \le g$ for all $1 \le i \le k$, it's easy to verify that $\sum_{i=1}^{k} |V_i|^2 \le rg^2$ and the equality holds if and only if $k = r$ and $|V_i| = g$ for all $1 \le i \le k = r$. Thus the total savings of π is $\frac{g(g-1)}{2}r$ if and only if $k = r$ and for all $1 \le i \le k$ the graph $G[V_i]$ is a clique of size g. This completes the proof. □

Because the NP-hardness of the Ring Grooming when $g > 2$, we will seek polynomial-time approximation algorithms. In the remaining of this section, we establish a general relation between approximalities of the two variants. Although no restriction is put on the number of groups in Unrestricted Ring Grooming, one can always convert any partition into one with at most $\lfloor \frac{2m}{g+1} \rfloor$ groups without increasing the cost. Indeed, we observe that any two groups whose total size is at most g can be merged into a larger group and the cost of resulting partition is not increased. By repeatedly performing this merging, we can reach a partition in which the total size of any pair of groups is at least $g + 1$. Let k be the number of groups and $d_1, d_2, \ldots, d_k$ be their sizes. Then we have $\binom{k}{2}$ inequalities $d_i + d_j \ge g + 1$. Sum up these inequalities and notice that each term d_i appears in exactly $k - 1$ of these inequalities. We have $(k-1)\sum_{i=1}^{k} d_i \ge \binom{k}{2}(g+1) = \frac{k(k-1)}{2}(g+1)$, which implies that $k \le \lfloor \frac{2m}{g+1} \rfloor$.

Conversely, one can always convert any partition into one with exactly $\lceil \frac{m}{g} \rceil$ groups and at most twice the cost as follows. Let $\{P_1, P_2, \ldots, P_k\}$ be a solution to Unrestricted Ring Grooming. Reorder the indices $1, 2, \ldots, m$ such that for any $1 \le i \le k$, P_i consists

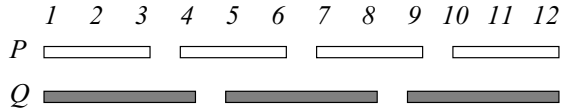


Fig. 1 Conversion from a solution to Unrestricted Ring Grooming into a solution to Restricted Ring Grooming

of consecutive indices. We construct $\{Q_1, Q_2, \ldots, Q_{\lceil \frac{m}{g} \rceil}\}$, a solution to Restricted Ring Grooming, by setting $Q_j = \{g(j-1)+1, g(j-1)+2, \ldots, gj\}$, for $1 \le j \le \lfloor \frac{m}{g} \rfloor$, and if $m \bmod g > 0$, $Q_{\lceil \frac{m}{g} \rceil} = \{g\lfloor \frac{m}{g} \rfloor + 1, g\lfloor \frac{m}{g} \rfloor + 2, \ldots, m\}$. This construction is illustrated in Fig. 1. As $|P_i| \le g$ for all $1 \le i \le k$, each group P_i is entirely contained in either one group Q_j, or two consecutive groups Q_j and Q_{j+1}. For $1 \le j \le \lceil \frac{m}{g} \rceil$, let j' be the index such that $g(j-1)+1 \in P_{j'}$, and let j'' be the index such that $gj \in P_{j''}$ if $j \le \lfloor \frac{m}{g} \rfloor$ or k if and if $m \bmod g > 0$ and $j = \lceil \frac{m}{g} \rceil$. Then for $1 \le j \le \lceil \frac{m}{g} \rceil$, $Q_j \subseteq \bigcup_{i=j'}^{j''} P_i$, and therefore $|\bigcup_{r \in Q_j} A_r| \le \sum_{i=j'}^{j''} |\bigcup_{r \in P_i} A_r|$. As each term $|\bigcup_{r \in P_i} A_r|$ can contribute to either only one term $|\bigcup_{r \in Q_j} A_r|$ or two terms $|\bigcup_{r \in Q_j} A_r|$ and $|\bigcup_{r \in Q_{j+1}} A_r|$, we have $\sum_{j=1}^{\lceil \frac{m}{g} \rceil} |\bigcup_{r \in Q_j} A_r| \le 2\sum_{i=1}^{k} |\bigcup_{r \in P_i} A_r|$. Thus the cost of $\{Q_1, Q_2, \ldots, Q_{\lceil \frac{m}{g} \rceil}\}$ is at most twice the cost of $\{P_1, P_2, \ldots, P_k\}$.

Based on above discussion, we have the following lemma.

Lemma 2. *Any polynomial-time α-approximation algorithm for Restricted Ring Grooming can lead to a polynomial-time α-approximation algorithm for Unrestricted Ring Grooming; and any polynomial-time α-approximation algorithm for Unrestricted Ring Grooming is also 2α-approximation algorithm for Restricted Ring Grooming.*

Proof: Fix an instance I of Ring Grooming, let $opt_u(I)$ and $opt_r(I)$ to denote its minimum unrestricted cost and minimum restricted cost respectively. Assume that the instance I contains m sets. Construct the instance I' by adding $\lfloor \frac{2m}{g+1} \rfloor g - m$ dummy empty sets to I. We claim that $opt_r(I') = opt_u(I)$. Let $OPT_u(I)$ be an optimal unrestricted solution with at most $\lfloor \frac{2m}{g+1} \rfloor$ groups to the instance I. A restricted solution to the instance I' can be obtained as follows. To each group in $OPT_u(I)$, we add dummy empty sets from I' until the group contains g sets. The remaining dummy empty sets are then partitioned into groups of size g. The cost of such solution is exactly $opt_u(I)$. Thus, $opt_r(I') \le opt_u(I)$. On the other hand, $opt_r(I') \ge opt_u(I') = opt_u(I)$. Thus, $opt_r(I') = opt_u(I)$.

Given any polynomial-time α-approximation algorithm for Restricted Ring Grooming, we apply it to the instance I' and let π' denote the output partition. Then the cost of π' is at most $\alpha \cdot opt_r(I') = \alpha \cdot opt_u(I)$. Let π denote the unrestricted solution to the instance I obtained from π' by removing the indices of the dummy empty sets from the groups of π'. Then π has the same cost as π'. So the cost of π is at most $\alpha \cdot opt_u(I)$.

Given any polynomial-time α-approximation algorithm for Unrestricted Ring Grooming, we apply it to the instance I and let π denote the output partition. Then the cost of π is at most $\alpha \cdot opt_u(I) \le \alpha \cdot opt_r(I)$. We convert π to a restricted partition with cost at most twice that of π. Then the cost of the obtained restricted partition is at most $2\alpha \cdot opt_r(I)$. □

In the next two sections, we develop some graph-theoretical results which will be used later for the design and analysis of our approximation algorithms.

3 Edge-connectivities

Let $G = (V, E)$ be any multigraph. Although we are interested mainly on loopless multigraph, we remark on that a loop contributes two to the degree of the node it is adjacent to, and that removing all loops does not affect any cut or pairwise edge-connectivity. It's well known that

$$|E_G(U)| = \frac{1}{2}\left(\sum_{u\in U} \deg_G(u) - |\Gamma_G(U)|\right), \quad \forall U \subseteq V;$$
$$c_G(u, v) = \min_{U\subseteq V(G), u\in U, v\notin U} |\Gamma_G(U)|.$$

Thus, if the degree of each node in G is even, so are all cuts and all pairwise edge connectivities. Given any positive integer c, we can partition V into classes such that for any two vertices in the same class, their connectivity is at least c in G. We call one class a c-cluster. An odd (even) c-cluster has an odd (even respectively) number of vertices.

Consider any two edges $e_1, e_2 \in E$. Let v_1 and v_2 be the two endpoints of e_1, and v_3 and v_4 be the two endpoints of e_2. Let e'_1, e'_2, e''_1 and e''_2 be four new edges between v_1 and v_3, v_2 and v_4, v_1 and v_4, and v_2 and v_3 respectively. A *switch* of e_1 and e_2 refers to the replacing of e_1 and e_2 either by e'_1 and e'_2, or by e''_1 and e''_2 (see Fig. 2). A switch is said to be *permissible* if no pairwise edge connectivity drops after the switch.

Lemma 3. *Let $G = (V, E)$ be any multigraph in which all nodes have even degrees. Then any pair of its edges allows a permissible switch.*

Proof: We prove the lemma by contradiction. Assume that there exist two edges $e_1, e_2 \in E$ such that after either switch, some pairwise edge connectivity drops. Let v_i for $1 \le i \le 4$, e'_i and e''_i for $1 \le i \le 2$ be the same in the above notation. Let G' (G'') be the graph obtained from G by replacing e_1 and e_2 by e'_1 and e'_2 (e''_1 and e''_2 respectively). Note that in both G' and G'' all nodes have even degrees.

We first claim that all v_i's must be distinct. In fact, first of all, e_1 and e_2 are not adjacent to any common vertex; for otherwise both G' and G'' are isomorphic to G and thus the lemma is true. Secondly, e_1 and e_2 can not both be loops; for otherwise both G' and G'' would contain the subgraph $G - \{e_1, e_2\}$ which preserves the pairwise edge-connectivities of G. At last, neither of e_1 and e_2 can be a loop; for otherwise, say e_1 is a loop and e_2 is not, replacing e_2 by the concatenation of e'_1 and e'_2 (or e''_1 and e''_2 in G'') in any path going through e_2 will at least maintain the same number of edge-disjoint paths between any pair of nodes.

Let u and v be the two vertices which have the *smallest* edge-connectivity among all pair of vertices whose edge-connectivity is decreased (by two) after either switch. Without loss of generality, assume that $c_{G'}(u, v) = c_G(u, v) - 2$. We first observe that the edge connectivity

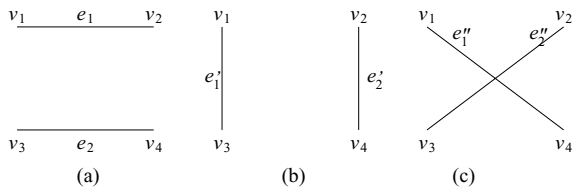


Fig. 2 The two possible switches of two edges

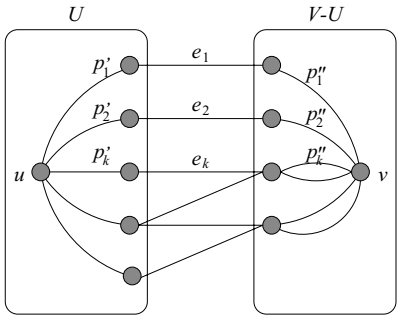
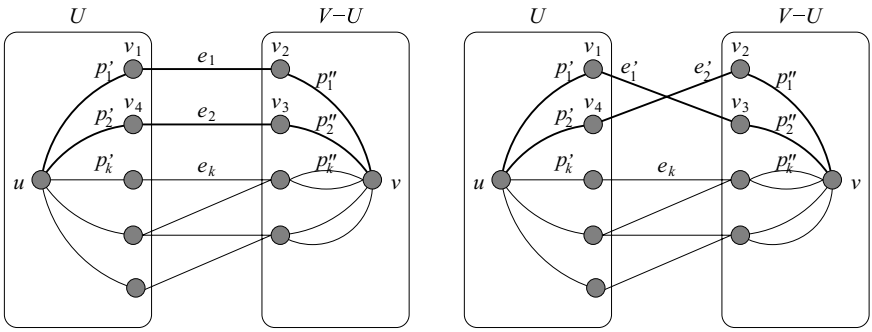


Fig. 3 The breaking-up of the edge-disjoint paths

between u and v in the graph $G - \{e_1, e_2\}$ is exactly $c_G(u, v) - 2$, since on one hand it is at least $c_G(u, v) - 2$, and on the other hand it is at most $c_{G'}(u, v)$. Let U be any subset of V such that $u \in U$, $v \notin U$, and exactly $c_G(u, v) - 2$ edges across between U and $V - U$ in $G - \{e_1, e_2\}$. Then both e_1 and e_2 are in $\Gamma_G(U)$; for otherwise $|\Gamma_G(U)|$ is at most $c_G(u, v) - 1$, which is impossible. Let e_k for $1 \le k \le c_G(u, v)$ denote the $c_G(u, v)$ edges in $\Gamma_G(U)$. Consider any $c_G(u, v)$ edge-disjoint paths from u to v in G, p_k for $1 \le k \le c_G(u, v)$. Then each path p_k goes through a unique edge in $\Gamma_G(U)$. By proper relabeling, assume that path p_k passes through edge e_k for $1 \le k \le c_G(u, v)$. The edge e_k breaks the path p_k into three parts: the subpath p_k' in U, e_k itself, and the subpath p_k'' in $V - U$, as illustrated in Fig. 3.

Without loss of generality, assume that $v_1 \in U$. Then $v_2 \notin U$. We prove by contradiction that $v_3 \in U$ and thus $v_4 \notin U$. Suppose to the contrary. Then there are still $c_G(u, v)$ edge-disjoint paths between u and v in G': one is p_1', followed by e_1' and then p_2''; the second is p_2', followed by e_2' and then p_1''; and the other $c_G(u, v) - 2$ paths are p_k for $3 \le k \le c_G(u, v)$ (see Fig. 4). All these paths are edge-disjoint, $c_{G'}(u, v) = c_G(u, v)$, which is a contradiction.

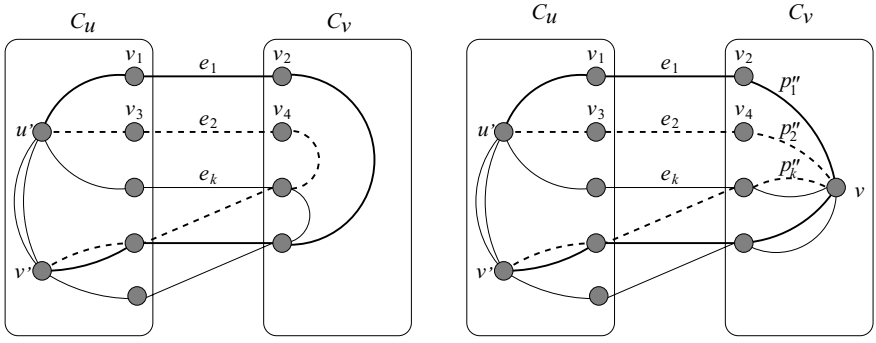
We will prove by contradiction that in G'' any pairwise edge connectivity does not decrease compared to G. Suppose to the contrary that there exist two vertices u' and v' with $c_{G''}(u', v') < c_G(u', v')$. From the selection of u and v, we have $c_G(u', v') \ge c_G(u, v)$. Let C_u and C_v denote the connected components of u and v, respectively, in $G - \Gamma_G(U)$. Then both u' and v' must be in $C_u \cup C_v$, for otherwise their connectivity is not affected. Without loss of generality, assume that $u' \in C_u$. If $v' \in C_v$, then $c_G(u', v')$ is at most $c_G(u, v)$ and thus must be equal to $c_G(u, v)$. This implies that for any set of $c_G(u', v')$ edge-disjoint paths between u'



(a). Before the switch. (b). After the switch.

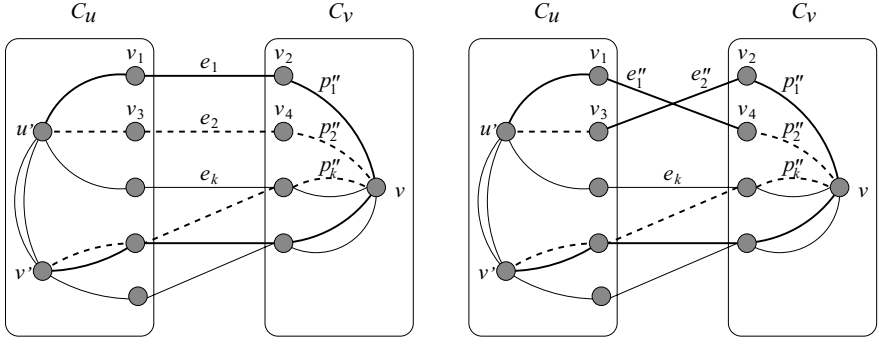
Fig. 4 Replacing e_1 and e_2 by e_1' and e_2' does not decrease the edge-connectivity between u and v

and v' in G, each of them passes through one unique edge in $\Gamma_G(U)$. Following the similar argument as in Fig. 4, we can show that $c_{G''}(u', v') = c_G(u', v')$, which is a contradiction. So v' must be in C_u as well. We first observe that among any set of $c_G(u', v')$ edge-disjoint paths between u' and v' in G, exactly two pass through e_1 and e_2 respectively, for otherwise $c_{G''}(u', v')$ would be at most one less than $c_G(u', v')$, hence be equal to $c_G(u', v')$ exactly. Secondly, we observe that for any set of $c_G(u', v')$ edge-disjoint paths between u' and v' in G, those paths across between C_u and C_v can be converted such that the part of each these paths within C_v is a concatenation of some unique pair of p_i'' and p_j'' (see Fig. 5). Consider any such set of $c_G(u', v')$ edge-disjoint paths between u' and v' in G, and let q_1 and q_2 be the two paths among them which pass through e_1 and e_2 respectively. If both e_1 and e_2 are between u' and v in q_1 and q_2 respectively, once again the similar argument as in Fig. 4 would lead to the conclusion that $c_{G''}(u', v') = c_G(u', v')$ (see Fig. 6). So without loss of generality, assume that e_1 is between u' and v in q_1 and e_2 is between v' and v in q_2. However, in this case we can construct a new path between u' and v' which does not pass through e_1 or e_2 as follows: the subpath of q_2 between u' and v, followed by the subpath of q_1 between v' and v (see Fig. 7). This new path is edge-disjoint from all other paths between u' and v'.



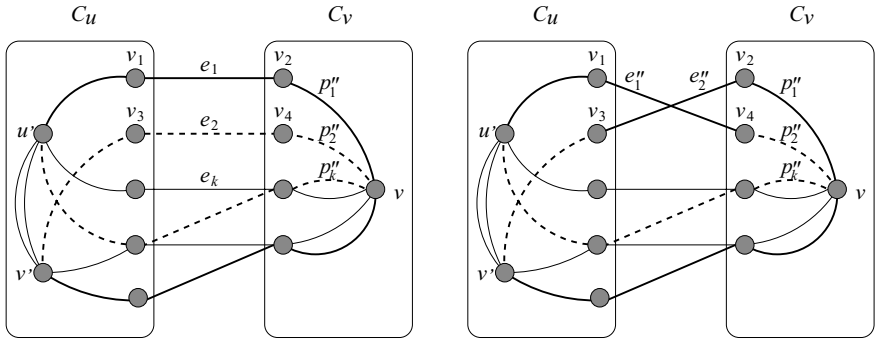
(a). Before the replacement. (b). After the replacement.

Fig. 5 Replacing each subpath within C_v by a concatenation of some unique pair of p_i'' and p_j''



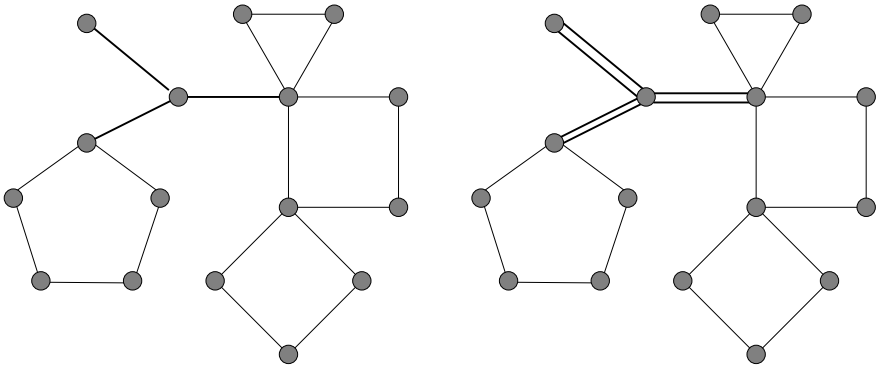
(a). Before the switch. (b). After the switch.

Fig. 6 If both e_1 and e_2 are between u' and v, replacing e_1 and e_2 by e_1'' and e_2'' also does not decrease the edge-connectivity between u' and v'



(a). Before the switch. (b). After the switch.

Fig. 7 If e_1 is between u' and v and e_2 between v' and v, replacing e_1 and e_2 by e_1'' and e_2'' also does not decrease the edge-connectivity between u' and v'



(a). Cactus defined in the literature. (b). Cactus defined by us.

Fig. 8 The comparison between the cactus defined in the literature and the cactus defined by us

Thus $c_{G''}(u', v')$ is at least $c_G(u', v') - 1$, and therefore must be equal to $c_G(u', v')$, again a contradiction. This completes the proof. □

We now describe the cactus representation of all minimum cuts of a graph, introduced by Dinits et al. (1976). In the literature, a cactus is a *simple* graph with the property that any two-*vertex*-connected component is an induced cycle (in other words, no cycle has chords). To clarify notation, a cycle cannot repeat vertices, but can consist of exactly two parallel edges in a graph. Also, a cycle cannot repeat vertices. We use a slightly modified version: a *cactus* is a multigraph such that each edge appears in exactly one cycle. After removing one edge from each couple of parallel edges from a cactus as defined by us, one obtains a cactus as defined in the literature. See Fig. 8 for the comparison. One important property of cactus is that any two cycles are either disjoint or touching at one vertex. Thus we can obtain a tree from a cactus, referred to as its *dual*, by putting a center at each cycle and adding one edge between two centers which correspond to two touching cycles.

Suppose that G is a two-*edge*-connected multigraph. Its *reduced representation*, denoted by G^*, is defined as follows: the vertices of G^* are the 3-clusters of G; each edge of G whose

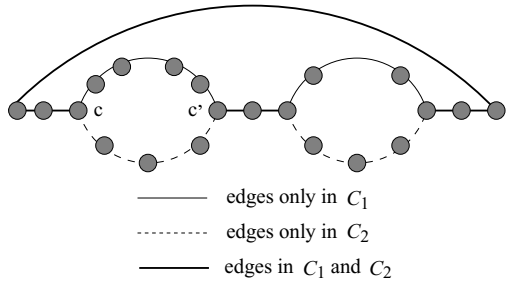


Fig. 9 Three edge-disjoint paths between c and c'

endpoints are in different 3-clusters is added to the edge set of G^*. G^* can have parallel edges. From Dinits et al. (1976) we know that the reduced representation of G^* is a cactus, and it is called in the literature the cactus representation. In addition, all the cuts of size two of G correspond to all the cuts of size two of G^*. As the original reference is in Russian, and we only need a special case, we provide a proof for the sake of completeness.

Theorem 4 (Dinits et al., 1976). *Suppose that G is a two-edge-connected multigraph. Then its reduced representation G^* is a cactus, and the 2-cuts of G correspond to the 2-cuts of G^*.*

Proof: Let $(U, \overline{U})$ be a 2-cut of G. A 3-cluster of G is either contained in U or disjoint from U. Therefore $(U, \overline{U})$ induces a partition of $V(G^*)$ and this partition defines a 2-cut of G^*. From a 2-cut of G^* it is immediate to obtain a 2-cut of G: a partition of $V(G^*)$ induces a partition of $V(G)$.

An 1-cut of G^* does not exist since it will induce a 1-cut of G. Therefore each edge of G^* is in a cycle. Assume by contradiction that there are two distinct cycles C_1 and C_2 in G^* which contain a common edge. Walking along C_1 from the common edge in an arbitrary direction. Let c be the first vertex from which C_1 and C_2 branch, and c' be the first vertex after c where C_1 and C_2 meet again. Then, in G^*, we have three edge disjoint paths from c to c': two on C_2 and one on C_1 that is in the previous walk from c to c' (see Fig. 9). Let u be a vertex of G in the 3-cluster represented by c and v be a vertex of G in the 3-cluster represented by c'. There is no 2-cut in G separating u from v, since there is no 2-cut in G^* separating c from c'. Therefore $c_G(u, v) > 2$. We obtained a contradiction to the fact that u and v are in different 3-clusters. In conclusion, no edge of G^* appears in two distinct cycles of G^*, and therefore G^* is a cactus. □

Next, we investigate what happens to the cactus representation when we validly switch two edges. For any vertex $v \in V$, let v^* be the 3-cluster contains v. Let e be any edge of G with endpoints u and v. If $u^* = v^*$, i.e., u and v are in the same 3-cluster, let e^* denote null and in this case we call e is *degenerated*; otherwise, let e^* denote the edge of G^* obtained from e and in this case we call e is nondegenerate. We say e is *related* to a 3-cluster c if either e^* is degenerated and $u^* = v^* = c$, or e^* is *nondegenerate* and c is on the unique cycle of the G^* containing e^*. Then any edge of G is related to some vertex of the cactus. The following lemma shows that under certain circumstances the switch leads to an "improved" G^*:

Lemma 5. *Suppose that G is a two-edge-connected multigraph and it has two edges e_1 and e_2 which allow a permissible switch. If there is no vertex of G^* to which e_1 and e_2 are both related, then any permissible switch of e_1 and e_2 either increases the number of the edges of G^* or decreases the number of vertices of G^*.*

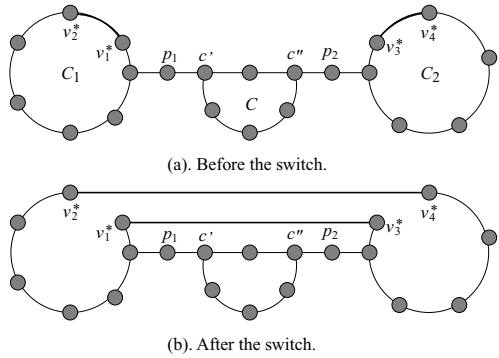


Fig. 10 After the permissible switch, there are three edge-disjoint paths from c' to c''

Proof: Assume by contradiction that after a permissible switch, neither does the number of vertices of G^* decrease, nor does the number of the edges of G^* increase. As the permissible switch does not split any 3-clusters, the number of vertices of G^* will remain the same after the permissible switch. Let v_i for $1 \leq i \leq 4$, e_i' and e_i'' for $1 \leq i \leq 2$ be the same in the previous notation, and assume that the switch which replaces replacing e_1 and e_2 by e_1' and e_2' is permissible. As e_1 and e_2 are not related to any common 3-cluster, $v_i^* \neq v_j^*$ for any $i = 1, 2$ and $j = 3, 4$. In addition, $v_1^* \neq v_2^*$ and $v_3^* \neq v_4^*$; for otherwise after doing the switch two edges are added while at most one is removed, and consequently the number of edges of G^* is increased. Thus all v_i^*'s are distinct, and both e_1^* and e_2^* are edges of G^*. For $i = 1$ and 2, let C_i be the unique cycle of G^* containing e_i^*. From the hypothesis, C_1 and C_2 are disjoint. Let p be a simple path in G^* from v_1^* to v_3^*. Pick any edge in p not contained in C_1 or C_2, and let C be the unique cycle of G^* containing this edge (see Fig. 10). As we walk along p from v_1^* to v_3^*, let c' be the first vertex of C and c'' the last vertex of C. From p we obtain that in G^*, there are edge disjoint paths p_1, from c' to v_1^*, and p_2, from c'' to v_3^*. Then after the permissible switch we have three edge-disjoint paths from c' to c'': two in C, and one obtained by concatenating p_1, $e_1'^*$ and p_2. This means that G^* is no longer a cactus, a contradiction. □

It can be easily verified that if a set of edges are *pairwise* related to a common 3-cluster, then they are *all* related to a common 3-cluster.

After applying a finite number of times permissible switches in two-edge-connected graph G as in the above lemma, we must reach the situation in which, any pair of edges in G which allow a permissible switch are related to a common vertex of G^*, and therefore their vertices are all contained in clusters on either one cycle or two touching cycles of G^*.

4 Inside matchings and valid matchings

Let $G = (V, E)$ be any loopless graph. A *fractional matching* is a function x: $E \longrightarrow Q$ satisfying that for any $v \in V$,

$$\sum_{e:e \text{ incident to } v} x(e) \leq 1.$$

If the equality holds for $v \in V$, then x is said to be perfect. A fractional matching x is a regular matching if $x(e) \in \{0, 1\}$ for any $e \in E$. Let w be any edge-weight function. The weight of a fractional matching x is defined as $w(x) = \sum_{e \in E} w(e)x(e)$. The following is a well-known theorem of Edmonds (for a reference, see (Graham et al., 1995), pages 201–206) regarding fractional matchings.

Theorem 6 (Edmonds). *If the fractional matching x satisfies:*

$$\sum_{e \in E(U)} x(e) \leq \left\lfloor \frac{|U|}{2} \right\rfloor, \quad \forall U \subseteq V$$

then it can be converted to a regular matching of at least the same weight.

In the remaining of this paper, all matchings are meant to be regular and perfect unless otherwise specified explicitly.

Now we consider a *complete* weighted graph K_m with even m. The number of different perfect matchings in K_m is

$$(m-1)!! = (m-1) \cdot (m-3) \cdot \cdots \cdot 3 \cdot 1.$$

As each edge is contained in $(m-3)!!$ different perfect matchings, the total weight of all these perfect matchings is $(m-3)!! \sum_{e \in E} w(e)$. Another useful fact is that for any perfect matching x in K_m, there are exactly $(m-2)!!$ different regular perfect matchings in K_m, each of which together with x form a cycle containing of all nodes in K_m. Let $\pi = \{V_1, V_2, \ldots, V_k\}$ be a partition of $V(K_m)$. For any $v \in V(K_m)$, let $\pi(v)$ be the unique value i with $v \in V_i$.

A *π-inside matching* of K_m is a perfect matching of K_m satisfying that the endpoints of each edge of the matching are within the same group of π. For an example, see Fig. 11. A necessary and sufficient condition for the existence of a π-inside perfect matching is that the size of any group is even. Notice that any inside matching can be generated by picking up a perfect matching of each group and put them together. Thus the total number of inside matchings is $\prod_{i=1}^{k} (|V_i| - 1)!!$.

A matching x is said to be *π-valid*, or simply *valid* if π is understood from the context, if the endpoints of each edge of x are within different groups of π. For an example, see Fig. 12. A necessary condition for the existence of a π-valid perfect matching is that the size of any group is at most $\frac{m}{2}$. This also turns to be a sufficient condition, which is proved below.

Let $|V_i| = d_i$ for $1 \leq i \leq k$. We use $f(d_1, d_2, \ldots, d_k)$ to denote the number of π-valid switchings. We observe that merging any two groups into one group will not increase the

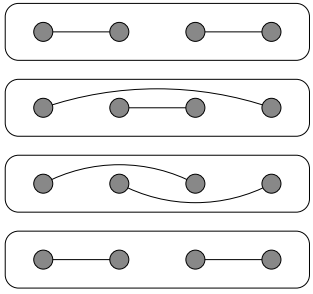


Fig. 11 The dots represent the vertices, and the partition π is given by the rectangles. The matching in the figure is a π-inside matching, as each edge has both endpoints in the same group of π

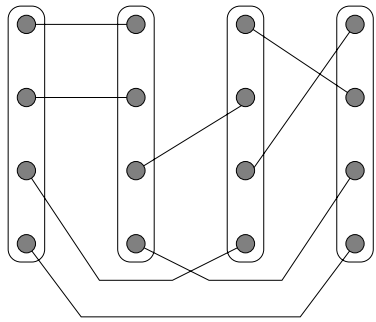


Fig. 12 The dots represent the vertices and the partition π is given by the rectangles. The matching in the figure is a π-valid matching, as no edge has both endpoints in the same group of π

number of valid matchings. So we only need to consider the case that total size of any two groups is greater than $\frac{m}{2}$. Then k must be at most three. If $k = 2$, there are exactly $(\frac{m}{2})!$ π-valid perfect matchings. So we assume that $k = 3$. In this case, any π-valid perfect matching can be generated as follows. Split V_1 into two parts V_{12} and V_{13}, V_2 into two parts V_{21} and V_{23}, V_3 into two parts V_{31} and V_{32} such that $|V_{12}| = |V_{21}|$, $|V_{13}| = |V_{31}|$, $|V_{23}| = |V_{32}|$. This is always possible and indeed the split must satisfy that $|V_{12}| = |V_{21}| = \frac{m}{2} - d_3$, $|V_{13}| = |V_{31}| = \frac{m}{2} - d_2$, $|V_{23}| = |V_{32}| = \frac{m}{2} - d_1$. Then for any pair of i and j, generate a bipartite matching between V_{ij} and V_{ji}. Combine the three generated bipartite matchings to obtain a π-valid perfect matching. Thus a π-valid perfect matching always exists. The number of these valid matchings can be further counted as follows. For each split, the number of different bipartite matching is $(\frac{m}{2} - d_1)! \cdot (\frac{m}{2} - d_2)! \cdot (\frac{m}{2} - d_3)!$. On the other hand, the number of different splits is

$$\binom{d_1}{\frac{m}{2} - d_3}\binom{d_2}{\frac{m}{2} - d_1}\binom{d_3}{\frac{m}{2} - d_2} = \frac{d_1! \cdot d_2! \cdot d_3!}{((\frac{m}{2} - d_1)! \cdot (\frac{m}{2} - d_2)! \cdot (\frac{m}{2} - d_3)!)^2}.$$

As different splits always lead to different matchings, the total number of matchings generated is

$$f(d_1, d_2, d_3) = \frac{d_1! \cdot d_2! \cdot d_3!}{\left(\frac{m}{2} - d_1\right)! \cdot \left(\frac{m}{2} - d_2\right)! \cdot \left(\frac{m}{2} - d_3\right)!}.$$

The above formula reveals a nice "monotone" property: if $d_1 \le d_2 < \frac{m}{2}$, then $f(d_1, d_2, d_3) \ge f(d_1 - 1, d_2 + 1, d_3)$. It can be proved by observing that

$$\frac{f(d_1, d_2, d_3)}{f(d_1 - 1, d_2 + 1, d_3)} = \frac{d_1\left(\frac{m}{2} - d_1 + 1\right)}{(d_2 + 1)\left(\frac{m}{2} - d_2\right)}.$$

and

$$d_1\left(\frac{m}{2} - d_1 + 1\right) - (d_2 + 1)\left(\frac{m}{2} - d_2\right) = (1 + d_2 - d_1)\left(\frac{m}{2} - d_3\right) \ge 0.$$

This "monotone" property has a useful interpretation: moving one vertex from the smaller group to a larger group will not increase the number of valid matchings. Thus we can repeatedly move one vertex from the smallest group to the second largest group without increasing the number of valid matchings. At the end we reach a partition consisting of only

two groups. At this time, we have $(\frac{m}{2})!$ valid matchings. Therefore, as long as the sizes of all groups in π are at most $\frac{m}{2}$, there are always at least $(\frac{m}{2})!$ π-valid perfect matchings.

The above monotone property is generally true for any k: if $d_1 \le d_2 < \frac{m}{2}$, then $f(d_1, d_2, d_3, \ldots, d_k) \ge f(d_1 - 1, d_2 + 1, d_3, \ldots, d_k)$. We prove it by induction on d_1. Let π' be the new partition obtained from π by moving one (arbitrary) vertex v in V_1 into V_2. When $d_1 = 1$, any π'-valid matching is also π-valid, and thus the number of valid matchings will not increase. So the monotone property is true when $d_1 = 1$. Now assume that $d_1 > 1$. On one hand, all π-valid matchings which become invalid with respect to π after the moving must contain an edge between v and one vertex in V_2 become invalid. The number of these matchings is $d_2 f(d_1 - 1, d_2 - 1, d_3, \ldots, d_k)$. On the other hand, all π'-valid matchings which are invalid with respect to π must contain an edge between v and one vertex $V_1 - v$. The number of these matchings is $(d_1 - 1) f(d_1 - 2, d_2, d_3, \ldots, d_k)$. Thus

$$\begin{aligned}
&f(d_1, d_2, d_3, \ldots, d_k) - f(d_1 - 1, d_2 + 1, d_3, \ldots, d_k) \\
&\quad = d_2 f(d_1 - 1, d_2 - 1, d_3, \ldots, d_k) - (d_1 - 1) f(d_1 - 2, d_2, d_3, \ldots, d_k) \\
&\quad \ge d_2 f(d_1 - 2, d_2, d_3, \ldots, d_k) - (d_1 - 1) f(d_1 - 2, d_2, d_3, \ldots, d_k) \\
&\quad = (1 + d_2 - d_1) f(d_1 - 2, d_2, d_3, \ldots, d_k) \ge 0,
\end{aligned}$$

where the first inequality follows from induction hypothesis. Thus the monotone property is true.

Now consider two partitions of $V(K_m)$: $\pi = \{V_1, V_2, \ldots, V_k\}, \sigma = \{U_1, U_2, \ldots, U_l\}$. Notice that partition π induces a partition $\pi|_{U_j}$ of U_j. If $|U_j|$ is even and is no less than $2|V_i|$ for $1 \le i \le k$, then there is a $\pi|_{U_j}$-valid perfect matching of U_j. Therefore if for all $1 \le j \le l$, $|U_j|$ is even and is no less than $2|V_i|$ for $1 \le i \le k$, then there exists a π-valid σ-inside matching. Any such matching can be generated by picking up a $\pi|_{U_j}$-valid perfect matching of U_j and combining them together. For an example, see Fig. 13.

Let x be any perfect matching of K_m. We construct a weighted multigraph $H(\pi, x)$ as follows: the vertex set is $\{1, 2, \ldots, k\}$; for any edge $e = (u, v)$ in x, add an edge between $\pi(u)$ and $\pi(v)$ in $H(\pi, x)$ and set its weight equal to $w(e)$. Note that if $|V_i| = d$ for all $1 \le i \le k$, then $H(\pi, x)$ is d-regular. If x is π-valid, then the graph $H(\pi, x)$ has no loop.

Lemma 7. *Let $\pi = \{V_1, V_2, \ldots, V_k\}$ be a partition of V with $|V_i| = d$ for all $1 \le i \le k$. Let x be a π-valid perfect matching. Assume that for any odd set $U \subseteq V(H(\pi, x))$, $|\Gamma_{H(\pi,x)}(U| \ge d$. Then there is a matching in $H(\pi, x)$ of weight at least $\frac{1}{d} w(x)$.*

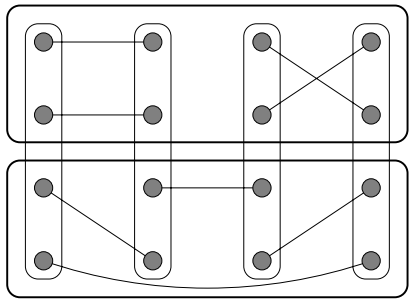


Fig. 13 The dots represent the vertices, the partition π is given by the vertical rectangles, and the partition σ is given by the horizontal thick solid rectangles. The matching in the figure is a π-valid σ-inside matching, as no edge has both endpoints in the same group of π, but every edge has both endpoints in the same group of σ

Proof: Note that $H(\pi, x)$ is d-regular and loopless. Let y be the fractional perfect matching of $H(\pi, x)$ given by $y(e) = \frac{1}{d}$ for any edge e. Then the weight of y is $\frac{1}{d}w(x)$. For any set $U \subseteq V(H(\pi, x))$,

$$\sum_{e \in E_{H(\pi,x)}(U)} y(e) = \frac{1}{d}|E_{H(\pi,x)}(U)| = \frac{1}{2d}\left(\sum_{u \in U} \deg_{H(\pi,x)}(u) - |\Gamma_{H(\pi,x)}(U)|\right)$$
$$= \frac{1}{2d}(d|U| - |\Gamma_{H(\pi,x)}(U)|) = \frac{|U|}{2} - \frac{1}{2d}|\Gamma_{H(\pi,x)}(U)|.$$

Thus if U is odd,

$$\sum_{e \in E_{H(\pi,x)}(U)} y(e) \leq \frac{|U|}{2} - \frac{1}{2d}d = \frac{|U| - 1}{2} = \left\lfloor \frac{|U|}{2} \right\rfloor.$$

The lemma then follows from Theorem 6. □

Let π and x be as specified in Lemma 7. Let (v_1, v_2) and (v_3, v_4) be two edges of x. Let x' (x'') be the matching obtained from x by replacing these two edges with (v_1, v_3) and (v_2, v_4) ((v_1, v_4) and (v_2, v_3) respectively). If $\pi(v_1) = \pi(v_3)$ or $\pi(v_2) = \pi(v_4)$, then x'' is π-valid and $H(\pi, x'')$ is isomorphic to $H(\pi, x)$. Similarly, if $\pi(v_1) = \pi(v_4)$ or $\pi(v_2) = \pi(v_3)$, then x' is π-valid and $H(\pi, x')$ is isomorphic to $H(\pi, x)$. If the four $\pi(v_i)$'s are all distinct, then both x' and x'' are π-valid and from Lemma 3, either $H(\pi, x')$ or $H(\pi, x'')$ does not decrease any pairwise edge connectivity compared to $H(\pi, x)$. Thus in either case, either x' is π-valid and $H(\pi, x')$ does not decrease any pairwise edge connectivity compared to $H(\pi, x)$, and/or x'' is π-valid and $H(\pi, x'')$ does not decrease any pairwise edge connectivity compared to $H(\pi, x)$. If x' (x'' respectively) is π-valid and $H(\pi, x')$ ($H(\pi, x'')$ respectively) does not decrease any pairwise edge connectivity compared to $H(\pi, x)$, we call the replacing of (v_1, v_2) and (v_3, v_4) with (v_1, v_3) and (v_2, v_4) ((v_1, v_4) and (v_2, v_3) respectively) as a *permissible switch* of (v_1, v_2) and (v_3, v_4). Then any pair of edges in a π-valid matching allows a permissible switch.

5 The iterative matching algorithm

We define a weighted complete graph K_m with vertex set $\{1, 2, \ldots, m\}$ by setting the weight of any edge $e = (i, j)$ equal to $w(e) = |A_i \cap A_j|$. Let $\pi = \{V_1, V_2, \ldots, V_k\}$ be a partition of $V = \{1, 2, \ldots, m\}$. The *intersection graph* of π, denoted by $K(\pi)$, is a weighted complete graph defined as follows: its vertex set is $\{1, 2, \ldots, k\}$, for any $e = (i, j)$ its weight is $|(\bigcup_{r \in V_i} A_r) \cap (\bigcup_{r \in V_j} A_r)|$.

In this section, we assume that g is a power of two and $m \bmod g = 0$. The iterative matching produces a solution to Restricted Ring Grooming. An initial partition π_0 consists of m groups, each one with one index. The algorithm runs in $\log g$ iterations. For any $1 \leq i \leq \log g$, the i-th iteration finds a maximum-weighted perfect matching of $K(\pi_{i-1})$, and then constructs a new partition π_i as follows: for each edge (j_1, j_2) of the obtained matching, a group of π_i is generated by merging the j_1-th group and the j_2-th group. Note that each π_i is a 2^i-grouping, and the partition $\pi_{\log g}$ is a solution to Restricted Ring Grooming.

The analysis of the iterative matching algorithm is based on savings. Let S_i be the weight of any maximum-weighted perfect matching of $K(\pi_{i-1})$. Then it's easy to verify that the total savings of π_i is $\sum_{j=1}^{i} S_j$ for any $1 \le i \le \log g$. In particular, the total savings of $\pi_{\log g}$ output by the iterative matching is $\sum_{j=1}^{\log g} S_j$. Thus S_i is referred to as the savings of the i-th iteration.

Fix an optimal restricted ring grooming OPT_g which consists of $\frac{m}{g}$ groups $D_1, \ldots, D_{\frac{m}{g}}$ where for any $1 \le k \le \frac{m}{g}$, $D_k = \{k_i \mid 1 \le i \le g\}$. Let opt_g denote the minimum cost. One trivial upper bound on the approximation ratio of the iterative matching is $\frac{g}{2}$. In fact, the cost of the iterative matching is

$$\sum_{r=1}^{m} |A_r| - \sum_{j=1}^{\log g} S_j = \left(\sum_{r=1}^{m} |A_r| - S_1 \right) - \sum_{j=2}^{\log g} S_j = opt_2 - \sum_{j=2}^{\log g} S_j$$

$$\le opt_2 \le \sum_{k=1}^{\frac{m}{g}} \sum_{j=1}^{\frac{g}{2}} |A_{k_{2j-1}} \cup A_{k_{2j}}| \le \frac{g}{2} \cdot \sum_{k=1}^{\frac{m}{g}} \left| \bigcup_{i=1}^{g} A_{k_i} \right| = \frac{g}{2} \cdot opt_g.$$

In the remaining of this section, we will provide tighter analysis of the approximation ratio of the iterative matching.

Any OPT_g-inside perfect matching is simply referred to as an *inside matching*, i.e., the term inside is always with respect to the optimal partition OPT_g. From the discussion in Section 4, the total number of inside matchings is $((g-1)!!)^{\frac{m}{g}}$. We are particularly interested in the following $(g-1)!!$ inside matchings. For any $1 \le k \le \frac{m}{g}$, arrange the $(g-1)!!$ perfect matchings inside D_k in the *non-decreasing* order of their weights and let $M_j(D_k)$ denote the weight of the j-th perfect matching. For $1 \le j \le (g-1)!!$, let $M_j = \sum_{k=1}^{\frac{m}{g}} M_j(D_k)$. The sum of these $(g-1)!!$ weights is $(g-3)!! \sum_{k=1}^{\frac{m}{g}} \sum_{1 \le i < j \le g} |A_{k_i} \cap A_{k_j}|$. For any $i \ge 1$, let $\ell(i) = (\frac{g}{2^{i-1}} - 1) \cdot (g-3)!!$. The next lemma is the key to our analysis of the performance of the iterative matching.

Lemma 8. *If $S_i \ge \frac{M_{\ell(i)}}{2^{i-1}}$ for any $1 \le i \le \log g$, then the approximation ratio of the iterative matching is at most $\frac{g}{4} + \frac{1}{2}$.*

Proof: Suppose that $S_i \ge \frac{M_{\ell(i)}}{2^{i-1}}$ for any $1 \le i \le \log g$. Since $\ell(i) - \ell(i+1) = \frac{g \cdot (g-3)!!}{2^i}$, the total savings of $\pi_{\log g}$ is

$$\sum_{i=1}^{\log g} S_i \ge \sum_{i=1}^{\log g} \frac{M_{\ell(i)}}{2^{i-1}} = \frac{2}{g \cdot (g-3)!!} \sum_{i=1}^{\log g} \frac{g \cdot (g-3)!!}{2^i} M_{\ell(i)}$$

$$\ge \frac{2}{g \cdot (g-3)!!} \sum_{i=1}^{\log g} \sum_{j=\ell(i+1)+1}^{\ell(i)} M_j = \frac{2}{g \cdot (g-3)!!} \sum_{j=1}^{(g-1)!!} M_j$$

$$= \frac{2}{g} \sum_{k=1}^{\frac{m}{g}} \sum_{1 \le i < j \le g} |A_{k_i} \cap A_{k_j}|.$$

So the total cost of $\pi_{\log g}$ is at most

$$\sum_{i=1}^{m} |A_i| - \frac{2}{g} \sum_{k=1}^{\frac{m}{g}} \sum_{1 \le i < j \le g} |A_{k_i} \cap A_{k_j}| = \sum_{k=1}^{\frac{m}{g}} \left(\sum_{i=1}^{g} |A_{k_i}| - \frac{2}{g} \sum_{1 \le i < j \le g} |A_{k_i} \cap A_{k_j}| \right)$$

$$\le \left(\frac{g}{4} + \frac{1}{2} \right) \sum_{k=1}^{\frac{m}{g}} \left| \bigcup_{i=1}^{g} A_{k_i} \right| = \left(\frac{g}{4} + \frac{1}{2} \right) \cdot opt_g.$$

The above inequality follows from that

$$\sum_{i=1}^{g} |A_{k_i}| - \frac{2}{g} \sum_{1 \le i < j \le g} |A_{k_i} \cap A_{k_j}| \le \left(\frac{g}{4} + \frac{1}{2} \right) \left| \bigcup_{i=1}^{g} A_{k_i} \right|$$

which can be proved as follows. Consider any $a \in \bigcup_{i=1}^{g} A_{k_i}$. Suppose that it appears in exactly r sets among $A_{k_1}, A_{k_2}, \ldots, A_{k_g}$. Then it appears in exactly $\binom{r}{2}$ $A_{k_i} \cap A_{k_j}$'s. Thus the total contribution of a in the left-hand side is $r - \frac{2}{g}\binom{r}{2} \le \frac{g}{4} + \frac{1}{2}$, and thus the inequality holds. □

Note that $S_1 \ge M_{\ell(1)}$ is true trivially. In general, for any π_{i-1}-valid perfect matching x, S_i is no less than the weight of any matching of $H(\pi_{i-1}, x)$. Notice that each such graph $H(\pi_{i-1}, x)$ is a 2^{i-1}-regular graph. From Lemmas 7 and 8, for any $2 \le i \le \log g$, $S_i \ge \frac{M_{\ell(i)}}{2^{i-1}}$ if there is a π_{i-1}-valid perfect matching x satisfying that

- **Weight Condition**: $w(x) \ge M_{\ell(i)}$;
- **Odd Condition**: $|\Gamma_{H(\pi_{i-1},x)}(U)| \ge 2^{i-1}$ for any odd set $U \subseteq V(H(\pi_{i-1}, x))$.

For the ease of argument for **Weight Condition**, we restrict our attention to inside matchings. From the discussion in Section 4, for any $2 \le i \le \log g$, there exist at least $((\frac{g}{2})!!)^{\frac{m}{g}}$ π_{i-1}-valid inside matchings. In the next we will show the existence of the π_{i-1}-valid inside matching that meets both **Weight Condition** and **Odd Condition** when $g = 4$ and 8.

5.1 The case $g = 4$

We only have to consider the second iteration. As any valid inside matching has weight at least M_1, the **Weight Condition** is met trivially. So we only have to consider the **Odd Condition**. For any π_1-valid inside matching x, $H(\pi_1, x)$ is 2-regular, and thus consists of a collection of disjoint cycles. As the size of any cut is even, the odd condition thus translates into the fact that all cycles in $H(\pi_1, x)$ are even. By applying valid switchings of pairs of edges inside the same optimal group, we can ensure that each connected component of $H(\pi_1, x)$ contains all or none of $\pi_1(k_1), \pi_1(k_2), \pi_1(k_3), \pi_1(k_4)$ for any 4-group $\{k_1, k_2, k_3, k_4\}$ of OPT_4, and therefore will have an even number of vertices. Therefore, the approximation ratio of the iteration matching is at most 1.5.

In the next, we will use a bad example to show that the above analysis is tight, that is the ratio of the iterative matching is exactly 1.5 when $g = 4$. Let $G = (V, E)$ be 4-regular graph and with girth (the size of a smallest cycle) at least five, whose existence of such graph follows from Lovasz (1979) (Problem 10.12). Let $n = 2|V| = |E|$. Assign to every edge e of G one index from $\{0, 1, \ldots, n-1\}$, and to every vertex v of G two indices from $\{0, 1, \ldots, n-1\}$, such that no index is assigned twice. Let v be a vertex of G, such that v is assigned indices

i_1 and i_4, and the four edges incident to it have assigned the indices i_2, i_3, i_5, i_6. Define four sets

$$A_{v1} = \{i_1, i_2, i_3\}, A_{v2} = \{i_3, i_4, i_5\}, A_{v3} = \{i_5, i_6, i_1\}, A_{v4} = \{i_2, i_4, i_6\}.$$

Repeat for all vertices v, obtaining $4|V|$ sets; this gives the instance of Ring Grooming. An (optimum) solution of cost $6|V|$ exists: any 4-group consists of the four sets defined by any vertex. Note that for $i \neq j$, $|A_{vi} \cap A_{vj}| = 1$. But suppose A_{v1} picks up i_2 and matches, in the first iteration, with whoever picks i_2 from the other side (i_2 is assigned to an edge). Similarly, A_{v2} picks i_3, A_{v3} picks i_5, and A_{v4} picks i_6, and each matches with an A_{ui} with $u \neq v$. The savings in this first iteration is $2|V|$: each two sets who match save exactly one. In the second iteration, consider any two 2-groups $\{A_{va}, A_{ub}\}$ and $\{A_{xc}, A_{yd}\}$, where $\{a, b, c, d\} \subseteq \{1, 2, 3, 4\}$. Note that vu and xy are edges in G. Then $|(A_{va} \cup A_{ub}) \cap (A_{xc} \cup A_{yd})| \leq 1$ which follows by cases from the structure of the first matching and the fact that G does not have short cycles. Therefore, the total savings in the second iteration cannot exceed V, and hence iterative matching produces a solution of cost at least $12|V| - 2|V| - |V| = 9|V| = \frac{3}{2}opt$.

In conclusion, we have the following theorem.

Theorem 9. *For $g = 4$, the performance ratio of the iteration matching algorithm is exactly* 1.5.

5.2 The case $g = 8$

We begin with the second iteration. Let x be any π_1-valid inside matching. Now we change x, if necessary, to ensure that $H(\pi_1, x)$ does not have any odd cycle, and x has weight at least M_{45}. The odd condition is met if each cycle of $H(\pi_1, x)$ has an even number of edges from each 8-group. So we now concentrate on one 8-group $D_k = \{k_1, k_2, \ldots, k_8\}$ of OPT_8. Let x' be the matching obtained from x by removing the four edges inside D_k. Then there exactly four paths in $H(\pi_1, x')$, with some possibly consisting of only one vertex. Permute the vertices such that we can assume the paths are: from $\pi_1(k_1)$ to $\pi_1(k_2)$, from $\pi_1(k_3)$ to $\pi_1(k_4)$, from $\pi_1(k_5)$ to $\pi_1(k_6)$, and from $\pi_1(k_7)$ to $\pi_1(k_8)$. Then there are $6!! = 48$ matchings inside D_k that link the four paths into one cycle. The best among these 48 matching beats $M_{45}(D_k)$. Add the best one to x', and let y be the resulting matching. Then y is an perfect π_1-valid inside matching, and any cycle of $H(\pi_1, y)$ has an even number of edges inside D_k. Moreover, the edge connectivity between any pair of nodes is not decreased compared to $H(\pi_1, x)$. We repeat this procedure for all 8-groups of OPT_8. At the end, we obtain a desired perfect π_1-valid inside matching.

We proceed to the third iteration. Let x be any perfect π_2-valid inside matching. This time $H = H(\pi_2, x)$ is a 4-regular graph. We first repeatedly perform permissible switchings of pairs of edges inside the same 8-group of OPT_8, until each connected component of H contains either all four or none of the edges of a perfect inside matching of any 8-group of OPT_8. At this time, each connected component of H has an even number of vertices and is a 4-regular 2-edge-connected. This allows us to work separately on the connected components of H. So from now on we assume H is 2-edge-connected. We then continue on permissible switches of pair of edges inside the same 8-group of OPT_8, until any two edges of H from any 8-group are related to some 4-cluster of H. The next lemma shows that at this time no 4-cluster can be odd.

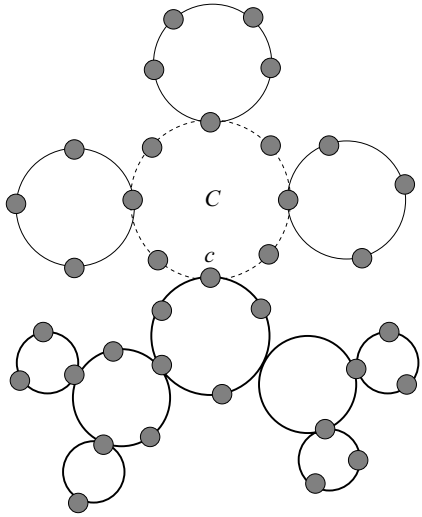


Fig. 14 The subcactus S_c represented by the collection of thick solid circiles

Lemma 10. *If any two edges of H from any* 8*-group are related to some* 4*-cluster of H, then no* 4*-cluster can be odd.*

Proof: Let C be a fixed simple cycle of H^*, the cactus representation of H. For all $c \in V(C)$, define S_c (a subcactus) to be the cactus induced by those cactus vertices c' such that any path from c' to $V(C) \setminus \{c\}$ goes through c. See Fig. 14 for an illustration. Note that $c \in V(S_c)$. Let $V_c \subseteq V(H)$ be the union of the 4-clusters of H represented by vertices of S_c. Since any two edges from the same 8-group are related to some common vertex of the cactus, if one edge from an 8-group has both endpoints in V_c, then any of the 4 edges of the 8-group has to be either with both endpoints in V_c or with endpoints in two distinct vertices of C. We say that an 8-group *belongs* to c if at least one of the four edges of H given by the 8-group has both endpoints in V_c. If an 8-group belongs to c and it has $4 - j$ edges with both endpoints in V_c, then j edges of C have to come from this 8-group. We say c has a *deficit* of size j if j edges of C have to be from 8-groups belonging to c. Note that if $c' \neq c$ is another vertex of C, the 8-groups belonging to c' are all distinct from the 8-groups belonging to c. So the sum of the deficits of the vertices of C cannot exceed $|C|$.

As a vertex of H is made from four original sets, the vertices of V_c are made from $4|V_c|$ original sets. Two of these original sets are linked to original sets outside V_c: the two edges of C incident with c. These two original sets are both matched outside V_c. Excluding them, we are left with $4|V_c| - 2$ originals sets in V_c, matched two by two. If $|V_c|$ is even, there must be an 8-group which belongs to c and without all four inside edges (as edges of H) having both endpoints in V_c, and thus c has a deficit of at least one. If $|V_c|$ is odd, then either (a) there is an 8-group with exactly two matched original sets in V_c, (b) there is one 8-group with exactly four matched original sets in V_c and another 8-group with exactly six matched original sets in V_c, or (c) there are three distinct 8-groups each with exactly six matched original sets in V_c. In all cases c has a deficit of at least three. As the sum of the deficits of the vertices of C cannot exceed $|C|$, we conclude that for each $c \in V(C)$, c has a deficit of exactly one and $|V_c|$ is even.

To complete the proof, we show that if there are odd 4-clusters, then there is a simple cycle C of the cactus and some $c \in V(C)$ with $|V_c|$ odd. Pick some vertex r (root) of the

cactus and let c be a vertex of the cactus representing an odd 4-cluster at maximum distance from r. As there is an even number of odd 4-clusters, $r \neq c$. Let T_c be the subcactus defined by the vertices c' with the property that any path from c' to r passes trough c. Note that c is the only vertex of T_c which represents an odd 4-cluster. Let C be the unique cycle containing c and not contained in T_c. Then with respect to C, $S_c = T_c$ and therefore $|V_c|$ is odd. □

Note that if all the 4-clusters are even, the two sides of any 2-cut are both the unions of 4-clusters of H, and therefore contain even number of nodes. So at this time the **Odd Condition** is met. Finally, we settle now beating M_{15} argument. We concentrate on an 8-group D_k and switch of edges. Any permissible switch can only increase connectivity. 4-clusters either merge or remain the same, and in both cases remain all of even size. Assume the current inside matching is $(k_1, k_2), (k_3, k_4), (k_5, k_6), (k_7, k_8)$. We produce another 15 matchings as follows: three by switching (k_1, k_2) with any of the other three edges; four by first switching (k_3, k_4) with (k_5, k_6) and then switching (k_1, k_2) with any of the other three edges; four by first switching (k_3, k_4) with (k_7, k_8) and then switching(k_1, k_2) with any of the other three edges; and four by first switching (k_5, k_6) with (k_7, k_8) and then switching (k_1, k_2) with any of the other three edges. Pick up the heaviest one among these 16 matchings and we beat $M_{15}(D_k)$. By repeating this procedure for all 8-groups of OPT_8, we conclude that there is an π_2-inside matching of weight at least M_{15}, and such that each odd set $S \subset V(H)$ has $|\Gamma(S)| \geq 4$. As both **Weight Condition** and **Odd Condition** are met, we have the following theorem.

Theorem 11. *For $g = 8$, the performance ratio of the iteration match is at most* 2.5.

6 Discussions

When g is greater than 8, we make the following conjecture on the approximation ratio of the **Iterative Matching**.

Conjecture 12. For any $g = 2^k$ with $k \geq 4$, the approximation ratio of the Iterative Matching for Restricted Ring Grooming is at most $\frac{g}{4} + \frac{1}{2}$.

The iterative matching can be modified to generate a solution to Unrestricted Grooming with arbitrary value of g. Let $\pi = \{V_1, V_2, \ldots, V_k\}$ be a partition of $V = \{1, 2, \ldots, m\}$ with $|V_i| \leq g$ for $1 \leq i \leq k$. Let $K'(\pi)$ be the subgraph of $K(\pi)$ which excludes all edges (i, j) in $K(\pi)$ with either $|V_i| + |V_j| > g$ or $w(i, j) = 0$. An initial partition π_0 consists of m groups, each one with one index. The l-th iteration finds a maximum-weighted matching of $K'(\pi_{l-1})$, and then construct a new partition π_l as follows: for each edge (j_1, j_2) of the obtained matching, a group of π_l is generated by merging the j_1-th group and the j_2-th group. The algorithm repeats this iteration until for some l, the graph $K'(\pi_{l-1})$ has no edge. At this moment, no further improvement can be made. However, the number of groups can be potentially further reduced by applying approximation algorithms for bin-packing problem (Coffman et al., 1997).

Following the same argument as in the beginning of this section, we can obtain a loose upper bound of $\lceil \frac{g}{2} \rceil$ on the approximation ratio of the proposed general iterative matching algorithm. However, its tight ratio remains open.

Unrestricted Ring Grooming can be formulated as a Weighted Set Cover problem as follows: the elements are $\{1, 2, \ldots, m\}$, the sets are the $(\!(\binom{m}{g})\!) = \binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{g}$

subsets of $\{1, 2, \ldots, m\}$ of size *at most* g, and each set P has cost $|\bigcup_{r\in P} A_r|$. A partition $\{P_1, P_2, \ldots, P_k\}$, solution to Unrestricted Ring Grooming, corresponds to picking the sets $P_1, P_2, \ldots, P_k$ in the set cover problem we defined, while from a Set Cover solution we can construct easily an Unrestricted Ring Grooming solution (which is a partition, not just a collection of sets) without increasing the cost. Thus the minimum cost of the defined Weighted Set Cover problem is *equal to* the minimum cost of the original Restricted Ring Grooming problem. Chvatal's algorithm (Chvátal, 1979) approximates within a factor of $1 + \log g$ this set cover problem. In conclusion, there is an algorithm for Unrestricted Ring Grooming with running time polynomial in $(\binom{m}{g})$ and performance ratio $1 + \log g$.

References

Chvátal V (1979) A greedy heuristic for the set-covering problem. Math Oper Res 4(3):233–235

Coffman EG Jr, Garey MR, Johnson DS (1997) Approximation algorithms for bin-packing—a survey. In: Approximation algorithms for NP-hard problems. PWS Publishing Company, Boston, pp 46–93

Dinits EA, Karzanov AV, Lomonosov ML (1976) On the structure of a family of minimal weighted cuts in a graph. In: Studies in discrete mathematics, pp 290–306

Garey MR, Johnson DS (1979) Computers and intractability. W.H. Freeman and Co

Graham RL, Grötschel M, Lóvasz L (1995) Handbook of combinatorics. MIT Press

Kann V (1994) Maximum bounded H-matching is MAX SNP-complete. Inf Process Lett (49):309–318

Lovasz L (1979) Combinatorial problems and exercises North Holland

Wan P-J, Liu L-W, Frieder O (1999) Grooming of arbitrary traffic in SONET/WDM rings. IEEE GLOBECOM'99 1B:1012–1016

Zhang X, Qiao C (1998) Effective and comprehensive solution to traffic grooming and wavelength assignment in SONET/WDM rings. SPIE Proc. of All-Optical Networking: Architecture, Control, and Management Issues (Boston, MA) 3531:221–232