RA-008

A Deterministic Approximation Algorithm for Maximum 2-Path Packing

Ruka Tanahashi* Zhi-Zhong Chen†

Abstract

This paper deals with the maximum-weight 2-path packing problem (M2PP), which is the problem of computing a set of vertex-disjoint paths of length 2 in a given edge-weighted complete graph so that the total weight of edges in the paths is maximized. Previously, Hassin and Rubinstein gave a randomized cubic-time approximation algorithm for M2PP which achieves an expected ratio of $\frac{35}{67} - \epsilon \approx 0.5223 - \epsilon$ for any constant $\epsilon > 0$. We refine their algorithm and derandomize it to obtain a *deterministic* cubic-time approximation algorithm for the problem which achieves a *better* ratio (namely, $0.5265 - \epsilon$ for any constant $\epsilon > 0$).

1 Introduction

Let G be an edge-weighted complete graph whose number of vertices is a multiple of 3. A *2-path packing* of G is a set of $\frac{1}{3}|V(G)|$ vertex-disjoint paths of length 2 in G. Given G, M2PP requires the computation of a 2-path packing P of G such that the total weight of edges on the paths in P is maximized over all 2-path packings of G.

M2PP is a classic NP-hard problem; indeed, its decision version is contained in Garey and Johnson's famous book on the theory of NP-completeness [2]. Hassin and Rubinstein [4] have presented a randomized cubic-time approximation algorithm for M2PP which achieves an expected ratio of $\frac{35}{67} - \epsilon$ for any constant $\epsilon > 0$. In this paper, we improve their result in twofold by presenting a *deterministic* cubic-time approximation algorithm for M2PP which achieves a *better* ratio (namely, $0.5265 - \epsilon$ for any constant $\epsilon > 0$).

To obtain our deterministic approximation algorithm for M2PP, we first obtain a new randomized cubic-time approximation algorithm for M2PP by refining the algorithm due to Hassin and Rubinstein. Like their algorithm, our new randomized algorithm starts by computing a maximum cycle cover $\mathcal{C}$ in the input graph G, then processes $\mathcal{C}$ to obtain three 2-path packings of G, and finally outputs the maximum weighted packing among the three packings. Unlike their algorithm, our algorithm processes triangles in $\mathcal{C}$ in a different way than the other cycles in $\mathcal{C}$. By carefully analyzing the new algorithm, we can show that it achieves an expected ratio of $0.5265(1 - \epsilon)$ for any constant $\epsilon > 0$. We then derandomize the algorithm using the pessimistic estimator method [5]; the derandomization is nontrivial.

2 Basic Definitions

Throughout the remainder of this paper, a graph means an undirected graph without parallel edges or self-loops each of whose edges has a nonnegative weight.

Let G be a graph. We denote the vertex set of G by $V(G)$ and denote the edge set of G by $E(G)$. For a set F of edges in G, $G - F$ denotes the graph obtained from G by removing the edges of F. The *degree* of a vertex v in G is the number of edges incident to v in G. The *weight* of a set F of edges in G, denoted by $w(F)$, is the total weight of edges in F. If F consists of a single edge e, we write $w(e)$ instead of $w(\{e\})$. The *weight* of a subgraph H of G, denoted by $w(H)$, is $w(E(H))$.

A *cycle* in G is a connected subgraph of G in which each vertex is of degree 2. A *path* in G is either a single vertex of G or a connected subgraph of G in which exactly two vertices are of degree 1 and the others are of degree 2. A *path component* of G is a connected component of G that is a path. The *length* of a cycle or path C, denoted by $|C|$, is the number of edges in C. We call a cycle C of G a *triangle* if $|C| = 3$, and call it a 4^+*-cycle* otherwise. A *cycle cover* of G is a subgraph H of G with $V(H) = V(G)$ in which each vertex is of degree 2. A *maximum-weight cycle cover* of G is a cycle cover of G whose weight is maximized over all cycle covers of G. A *matching* of G is a (possibly empty) set of pairwise nonadjacent edges of G. A *maximum-weight matching* of G is a matching of G whose weight is maximized over all matchings of G.

*Department of Mathematical Sciences, Tokyo Denki University, Hatoyama, Saitama 350-0394, Japan. Email: ruka@rnc.r.dendai.ac.jp

†Supported in part by the Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture of Japan, under Grant No. 20500021. Department of Mathematical Sciences, Tokyo Denki University, Hatoyama, Saitama 350-0394, Japan. Email: chen@r.dendai.ac.jp

The *distance* between two vertices u and v in G is the shortest length of a path between u and v in G.

For a random event A, $\Pr[A]$ denotes the probability that A occurs. For a random event A and one or more random events $B_1, \ldots, B_h$, $\Pr[A \mid B_1, \ldots, B_h]$ denotes the probability that A occurs given the occurrences of $B_1, \ldots, B_h$. For a random variable X, $\mathcal{E}[X]$ denotes the expected value of X. For a random variable X and one or more random events $B_1, \ldots, B_h$, $\mathcal{E}[X \mid B_1, \ldots, B_h]$ denotes the expected value of X given the occurrences of $B_1, \ldots, B_h$.

3 A Randomized Approximation Algorithm for M2PP

Throughout the remainder of this paper, we fix an instance G of M2PP and an arbitrary constant $\epsilon > 0$. Moreover, we fix a maximum-weight 2-path packing $\mathcal{O}pt$ of G.

The outline of Hassin and Rubinstein's algorithm [4] (H&R-algorithm for short) is as follows:

(1) Compute a maximum-weight cycle cover $\mathcal{C}$ of G. (*Comment:* $w(\mathcal{C}) \geq w(\mathcal{O}pt)$.)

(2) Modify $\mathcal{C}$ by breaking each cycle C in $\mathcal{C}$ with $|C| > \frac{1}{\epsilon}$ into cycles of length at most $1 + \frac{1}{\epsilon}$ such that the total weight of the cycles is at least $(1-\epsilon) \cdot w(C)$. (*Comment:* $w(\mathcal{C}) \geq (1-\epsilon) \cdot w(\mathcal{O}pt)$.)

(3) Process $\mathcal{C}$ to obtain three 2-path packings P_1, P_2, and P_3 of G and then output the maximum weighted packing among them. (*Comment:* The names P_1, P_2, and P_3 are inherited from the H&R-algorithm.)

Our algorithm differs from H&R-algorithm only in the computation of P_3. Before detailing our new computation of P_3, we first review their results on P_1 and P_2.

Lemma 3.1 [4] *Let* $\alpha \cdot w(\mathcal{C})$ *be the total weight of edges in triangles in* $\mathcal{C}$. *Then,* $w(P_1) \geq (\frac{1}{2} + \frac{1}{6}\alpha) w(\mathcal{C}) \geq (\frac{1}{2} + \frac{1}{6}\alpha)(1-\epsilon) \cdot w(\mathcal{O}pt)$.

Lemma 3.2 [4] *Let* $\beta \cdot w(\mathcal{O}pt)$ *be the total weight of those edges* $\{u, v\}$ *such that some path of length 2 in* $\mathcal{O}pt$ *contains both* u *and* v *and some cycle in* $\mathcal{C}$ *contains both* u *and* v. *Then,* $w(P_2) \geq \beta \cdot w(\mathcal{O}pt)$.

We next detail our new computation of P_3 which is basically a refinement of the computation of P_3 in H&R-algorithm and is also a modification of an algorithm in [1] for a different problem. The first step is as follows:

1. Compute a maximum-weight matching M_1 in a graph G_1, where $V(G_1) = V(G)$ and $E(G_1) = \{\{u, v\} \in E(G) \ : \ u \text{ and } v \text{ belong to different cycles in } \mathcal{C}\}$.

Note that $w(M_1)$ is heavy when $\mathcal{O}pt$ contains a heavy set of edges between cycles in $\mathcal{C}$. So, we want to add the edges of M_1 to $\mathcal{C}$. However, adding the edges of M_1 to $\mathcal{C}$ yields a graph which may have a lot of vertices of degree 3 and is hence far from a 2-path packing of G. To remedy this situation, we want to compute a set R of edges in $\mathcal{C}$ and a subset M of M_1 such that adding the edges of M to $\mathcal{C} - R$ yields a graph $\mathcal{C}'$ in which each connected component is a cycle or path. The next four steps of our algorithm are for computing R, M, and $\mathcal{C}'$. Before describing the details, we need to define several notations. Let $C_1, \ldots, C_r$ be the cycles in $\mathcal{C}$. Moreover, throughout the remainder of this paper, let p be the smallest positive real number satisfying the inequality $3p^2 - 2p^3 \geq \frac{3}{16}$; the reason why we select p in this way will become clear in Lemma 5.2. Note that $0.276 < p < 0.277$; hence $(1-p)^2 > \frac{1}{2}$. Now, we are ready to describe Steps 2 through 5 of our algorithm.

2. For each cycle C_i in $\mathcal{C}$, process C_i (independently of the other cycles in $\mathcal{C}$) by performing the following steps:
   (a) Initialize R_i to be the empty set.
   (b) If $|C_i| = 3$, then for each edge e of C_i, add e to R_i with probability p. (*Comment:* After this step, $0 \leq |R_i| \leq 3$. In contrast, $|R_i| = 1$ in H&R-algorithm.)
   (c) If $|C_i| \geq 4$, then perform the following steps:
      i. Choose one edge e_1 from C_i uniformly at random.
      ii. Starting at e_1 and going clockwise around C_i, label the other edges of C_i as $e_2, \ldots, e_c$, where c is the number of edges in C_i.
      iii. Add the edges e_j with $j \equiv 1 \pmod 4$ and $j \leq c - 3$ to R_i. (*Comment:* R_i is a matching of C_i and $|R_i| = \lfloor \frac{|C_i|}{4} \rfloor$.)
      iv. If $c \equiv 1 \pmod 4$, then add e_{c-1} to R_i with probability $\frac{1}{4}$. (*Comment:* R_i remains a matching in C_i. Moreover, $\mathcal{E}[|R_i|] = \frac{|C_i|-1}{4} + 1 \cdot \frac{1}{4} = \frac{|C_i|}{4}$.)
      v. If $c \equiv 2 \pmod 4$, then add e_{c-1} to R_i with probability $\frac{1}{2}$. (*Comment:* R_i remains a matching in C_i. Moreover, $\mathcal{E}[|R_i|] = \frac{|C_i|-2}{4} + 1 \cdot \frac{1}{2} = \frac{|C_i|}{4}$.)
      vi. If $c \equiv 3 \pmod 4$, then add e_{c-2} to R_i with probability $\frac{3}{4}$. (*Comment:* R_i remains a matching in C_i. Moreover, $\mathcal{E}[|R_i|] = \frac{|C_i|-3}{4} + 1 \cdot \frac{3}{4} = \frac{|C_i|}{4}$.)

3. Let $R = R_1 \cup \cdots \cup R_r$.
4. Let M be the set of all edges $\{u, v\} \in M_1$ such that both u and v are of degree 0 or 1 in graph $\mathcal{C} - R$.
5. Let $\mathcal{C}'$ be the graph obtained from $\mathcal{C} - R$ by adding the edges in M. (*Comment:* Each connected component of $\mathcal{C}'$ is a cycle or path. Moreover, every triangle in $\mathcal{C}'$ is also a triangle in $\mathcal{C}$ while every 4^+-cycle C in $\mathcal{C}'$ contains at least two edges in M.)

Note that our algorithm processes those cycles C_i of $\mathcal{C}$ with $|C_i| \geq 4$ as in the H&R-algorithm. So, we have the following lemma:

Lemma 3.3 *For every cycle C_i of $\mathcal{C}$ with $|C_i| \geq 4$, the following hold:*

(1) For every edge e of C_i, $\Pr[e \in R] = \frac{1}{4}$.

(2) For every vertex v of C_i, v is incident to at most one edge of R and the probability that v is incident to one edge of R is $\frac{1}{2}$.

By the comments on Step 5, $\mathcal{C}'$ is not so far from a 2-path packing. We can now finish computing P_3 by performing the following steps:

6. For each cycle C in $\mathcal{C}'$ with $|C| \geq 4$, choose one edge in $E(C) \cap M$ uniformly at random and delete it from $\mathcal{C}'$.
7. If $\mathcal{C}'$ has at least one path component, then perform the following two steps:
   (a) Connect the path components of $\mathcal{C}'$ into a single cycle Y by adding some edges of G.
   (b) Break Y into paths each of length 2 by removing a set F of edges from Y with $w(F) \leq \frac{1}{3} \cdot w(Y)$.
8. Remove the minimum-weight edge from each triangle in $\mathcal{C}'$. (*Comment:* After this step, each connected component of $\mathcal{C}'$ is a path of length 2.)
9. Let $P_3 = \mathcal{C}'$.

The following fact is clear from Steps 7 through 9:

Fact 3.4 *Let E_6 be the set of edges of $\mathcal{C}$ that remain in $\mathcal{C}'$ immediately after Step 6. Then, $w(P_3) \geq \frac{2}{3}w(E_6)$.*

Consider an edge $e \in M_1 \cup E(\mathcal{C})$. Let t_e be the probability that e remains in $\mathcal{C}'$ immediately after Step 6. If e appears in a triangle in $\mathcal{C}$, then by Step 2b, $t_e = 1 - p$. If $e \in E(\mathcal{C})$ does not appear in a triangle in $\mathcal{C}$, then by Statement (1) in Lemma 3.3, $t_e = \frac{3}{4}$. If $e \in M_1$, then we can claim that $t_e \geq \frac{3}{16}$. So,

$$\begin{aligned}&\mathcal{E}[w(E_6)]\\ &\geq (1-p)\alpha \cdot w(\mathcal{C}) + \frac{3}{4}(1-\alpha) \cdot w(\mathcal{C}) + \frac{3}{16}w(M_1)\\ &= \left(\frac{3}{4} + (\frac{1}{4} - p)\alpha\right) w(\mathcal{C}) + \frac{3}{16}w(M_1).\end{aligned}$$

Note that the above argument is informal because we have not proved the claim. Indeed, we will not prove the claim because we will never use it to prove anything. The claim and the above informal argument are only for helping the reader understand what we are going to do next. In fact, the next section shows how to derandomize Steps 2 through 6 (using the pessimistic estimator method [5]) to obtain E_6 deterministically so that $w(E_6) \geq (\frac{3}{4} + (\frac{1}{4} - p)\alpha)w(\mathcal{C}) + \frac{3}{16}w(M_1)$. Of course, this lower bound on $w(E_6)$ will be proved rigorously (without using the unproved claim).

4 A Crucial Lemma

This section proves a lemma that is crucial for our derandomization of the above randomized algorithm. It is similar to Lemma 3 in [4] but does not follow from the latter directly.

Lemma 4.1 *Consider an arbitrary $i \in \{1, \ldots, r\}$ with $|C_i| \geq 4$, and consider two arbitrary vertices u and v of C_i. Let A_1 be the event that the degree of v in graph $C_i - R_i$ is 1. Let A_2 be the event that u and v are the endpoints of some path component of $C_i - R_i$. Let A_3 be the event that u and v are endpoints of two different path components of $C_i - R_i$. For each $j \in \{1, 2, 3\}$, let $s_j = \Pr[A_j]$. Then, $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{4}s_1$, or equivalently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$.*

Proof. We prove the lemma by a case-analysis. Let $c = |C_i|$ and let d be the distance between u and v in C_i. Since $s_3 \leq s_1$, we need to consider only those cases where $s_2 \neq 0$. For example, $s_2 = 0$ if $d \geq 4 + (|C_i| \bmod 4)$ and thus we ignore such C_i in the rest of the proof. We say that C_i is *long* if $|C_i| \geq 8$, and is *short* otherwise. Note that if C_i is long, then at least two edges are added to R_i in Step 2c. For convenience, starting at v and going clockwise around C_i, we label the edges of C_i as f_1, f_2, …, f_c. To prove that $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$, we may assume that the edges incident to u in C_i are f_d and f_{d+1}.

Case 1: $|C_i| = 4$. In this case, $d = 1$ or 2. If $d = 1$ (cf. Figure 1(1)), then $s_2 = \frac{1}{4}$ and $s_3 = 0$. If $d = 2$, then $s_2 = s_3 = 0$. So, we always have $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$.

Case 2: $|C_i| = 5$. In this case, $d = 1$ or 2. If $d = 1$ (cf. Figures 1(2) through (5)), then $s_2 =$

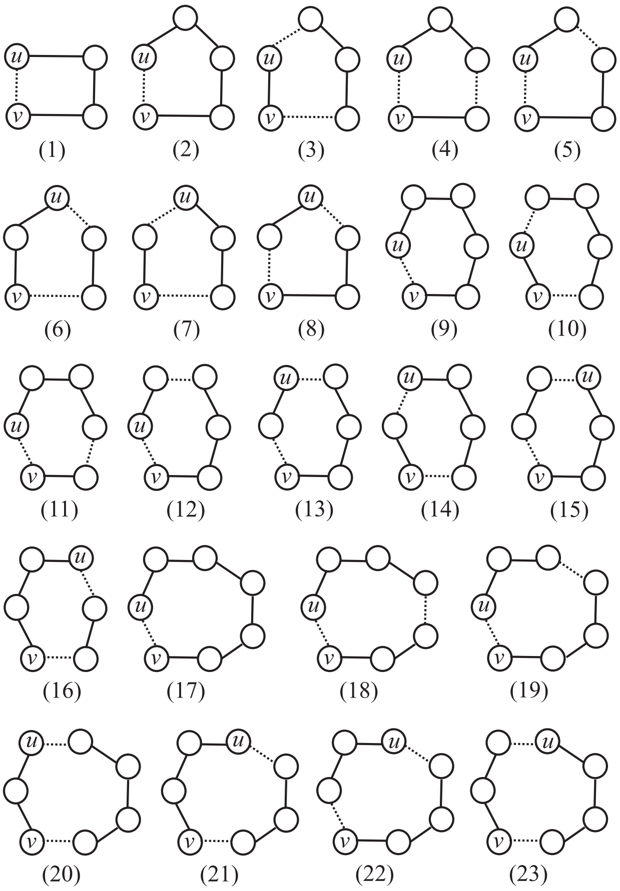

Figure 1: Short cycles C_i, where the edges in R_i are dotted.

$\frac{1}{5} \cdot \frac{3}{4} + \frac{1}{5} \cdot \frac{1}{4} = \frac{1}{5}$ and $s_3 = 2 \cdot \frac{1}{5} \cdot \frac{1}{4} = \frac{1}{10}$. If $d = 2$ (cf. Figures 1(6) through (8)), then $s_2 = \frac{1}{5} \cdot \frac{1}{4} = \frac{1}{20}$ and $s_3 = 2 \cdot \frac{1}{5} \cdot \frac{1}{4} = \frac{1}{10}$. So, we always have $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$.

Case 3: $|C_i| = 6$. In this case, $d = 1$, 2, or 3. If $d = 1$ (cf. Figures 1(9) through (12)), then $s_2 = s_3 = 2 \cdot \frac{1}{6} \cdot \frac{1}{2} = \frac{1}{6}$. If $d = 2$ (cf. Figures 1(13) and (14)), then $s_2 = 0$ and $s_3 = 2 \cdot \frac{1}{6} \cdot \frac{1}{2} = \frac{1}{6}$. If $d = 3$ (cf. Figures 1(15) and (16)), then $s_2 = 2 \cdot \frac{1}{6} \cdot \frac{1}{2} = \frac{1}{6}$ and $s_3 = 0$. So, we always have $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$.

Case 4: $|C_i| = 7$. In this case, $d = 1$, 2, or 3. If $d = 1$ (cf. Figures 1(17) through (19)), then $s_2 = \frac{1}{7} \cdot \frac{1}{4} = \frac{1}{28}$ and $s_3 = 2 \cdot \frac{1}{7} \cdot \frac{3}{4} = \frac{3}{14}$. If $d = 2$ (cf. Figure 1(20)), then $s_2 = \frac{1}{7} \cdot \frac{3}{4} = \frac{3}{28}$ and $s_3 = 0$. If $d = 3$ (cf. Figures 1(21) through (23)), then $s_2 = \frac{1}{7} \cdot \frac{3}{4} = \frac{3}{28}$ and $s_3 = 2 \cdot \frac{1}{7} \cdot \frac{3}{4} = \frac{3}{14}$. So, we always have $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$.

Case 5: C_i is long and $|C_i| \equiv 0 \pmod 4$. In this case, $s_2 = 0$ if $d \neq 3$. Moreover, if $d = 3$, then $s_2 = \frac{1}{4}$ and $s_3 = 0$. So, we always have $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{8}$.

Case 6: C_i is long and $|C_i| \equiv 1 \pmod 4$. Let $k = \frac{c-1}{4}$ and $S = \{f_1, f_4, f_5, f_c\}$. Obviously, $s_2 = 0$ if $d \geq 5$. So, it suffices to consider the following subcases:

Case 6.1: $d = 4$. In this case, let $S = \{f_1, f_4, f_5, f_c\}$. Then, $s_2 = \frac{1}{c} \cdot \frac{3}{4}$ because event A_2 occurs exactly when edge f_5 is selected as e_1 in Step 2(c)i and edge f_3 is not added to R_i in Step 2(c)iv. Moreover, if event A_3 occurs, exactly one of the following events occurs:

- $A_{3,1}$: $S \cap R_i = \{f_4, f_c\}$.
- $A_{3,2}$: $S \cap R_i = \{f_1, f_5\}$.
- $A_{3,3}$: $S \cap R_i = \{f_c, f_5\}$ and $f_3 \in R_i$.
- $A_{3,4}$: $S \cap R_i = \{f_1, f_4\}$.

Obviously, event $A_{3,1}$ occurs exactly when one of f_9, f_{13}, $\ldots$, f_c is selected as e_1 in Step 2(c)i, implying that $\Pr[A_{3,1}] = \frac{k-1}{c}$. Similarly, event $A_{3,2}$ occurs exactly when one of f_1, f_{10}, f_{14}, $\ldots$, f_{c-3} is selected as e_1 in Step 2(c)i, implying that $\Pr[A_{3,2}] = \frac{k-1}{c}$. Moreover, event $A_{3,3}$ occurs exactly when f_5 is selected as e_1 in Step 2(c)i and f_3 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,3}] = \frac{1}{c} \cdot \frac{1}{4}$. Furthermore, event $A_{3,4}$ occurs exactly when f_6 is selected as e_1 in Step 2(c)i and f_4 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,4}] = \frac{1}{c} \cdot \frac{1}{4}$. Therefore, $s_3 = 2(\frac{k-1}{c} + \cdot \frac{1}{c} \cdot \frac{1}{4}) = \frac{4k-3}{2c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{3}{8c} + \frac{4k-3}{8c} \leq \frac{1}{8}$ for $c = 4k + 1$.

Case 6.2: $d = 3$. In this case, let $S = \{f_1, f_3, f_4, f_c\}$. Then, $s_2 = \frac{k-1}{c}$ because event A_2 occurs exactly when one of f_9, f_{13}, $\ldots$, f_c is selected as e_1 in Step 2(c)i. Moreover, if event A_3 occurs, exactly one of the following events occurs:

- $A_{3,1}$: $S \cap R_i = \{f_3, f_c\}$.
- $A_{3,2}$: $S \cap R_i = \{f_1, f_4\}$.
- $A_{3,3}$: $S \cap R_i = \{f_1, f_3\}$.

Obviously, event $A_{3,1}$ occurs exactly when f_5 is selected as e_1 in Step 2(c)i and f_3 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,1}] = \frac{1}{c} \cdot \frac{1}{4}$. Similarly, event $A_{3,2}$ occurs exactly when f_6 is selected as e_1 in Step 2(c)i and f_4 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,2}] = \frac{1}{c} \cdot \frac{1}{4}$. Furthermore, event $A_{3,3}$ occurs exactly when f_3 is selected as e_1 in Step 2(c)i and f_1 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,3}] = \frac{1}{c} \cdot \frac{1}{4}$. Therefore, $s_3 = 3 \cdot \frac{1}{c} \cdot \frac{1}{4} = \frac{3}{4c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{k-1}{2c} + \frac{3}{16c} \leq \frac{1}{8}$ for $c = 4k+1$.

Case 6.3: $d = 2$. In this case, let $S = \{f_1, f_2, f_3, f_c\}$. Then, $s_2 = \frac{1}{c} \cdot \frac{1}{4}$ because event A_2 occurs exactly when f_5 is selected as e_1 in Step 2(c)i and f_3 is added to R_i in Step 2(c)iv. Moreover, if event A_3 occurs, exactly one of the following events occurs:

- $A_{3,1}$: $S \cap R_i = \{f_2, f_c\}$.
- $A_{3,2}$: $S \cap R_i = \{f_1, f_3\}$.

Obviously, event $A_{3,1}$ occurs exactly when f_2 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,1}] = \frac{1}{c} \cdot \frac{1}{4}$. Moreover, event $A_{3,2}$ occurs exactly when f_3 is selected as e_1 in Step 2(c)i and f_1 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,2}] = \frac{1}{c} \cdot \frac{1}{4}$. Therefore, $s_3 = 2 \cdot \frac{1}{c} \cdot \frac{1}{4} = \frac{1}{2c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{1}{8c} + \frac{1}{8c} \leq \frac{1}{8}$ for $c = 4k+1$.

Case 6.4: $d = 1$. In this case, $s_2 = \frac{1}{c} \cdot \frac{1}{4}$ because event A_2 occurs exactly when f_2 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv. Moreover, event A_3 occurs exactly when $\{u, v\} \in R_i$. Therefore, $s_3 = \frac{1}{4}$ by Statement (1) in Lemma 3.3. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{1}{8c} + \frac{1}{16} \leq \frac{1}{8}$ for $c = 4k+1$.

Case 7: C_i is long and $|C_i| \equiv 2 \pmod 4$. Let $k = \frac{c-2}{4}$ and $S = \{f_1, f_5, f_6, f_c\}$. Obviously, $s_2 = 0$ if $d \notin \{1, 3, 5\}$. So, it suffices to consider the following subcases:

Case 7.1: $d = 5$. In this case, let $S = \{f_1, f_5, f_6, f_c\}$. Then, $s_2 = \frac{1}{c} \cdot \frac{1}{2}$ because event A_2 occurs exactly when edge f_6 is selected as e_1 in Step 2(c)i and edge f_4 is not added to R_i in Step 2(c)iv. Moreover, if event A_3 occurs, exactly one of the following events occurs:

- $A_{3,1}$: $S \cap R_i = \{f_6, f_c\}$ and $f_4 \in R_i$.
- $A_{3,2}$: $S \cap R_i = \{f_6, f_c\}$ and $f_2 \in R_i$.
- $A_{3,3}$: $S \cap R_i = \{f_1, f_5\}$ and $f_7 \notin R_i$.
- $A_{3,4}$: $S \cap R_i = \{f_1, f_5\}$ and $f_7 \in R_i$.

Obviously, event $A_{3,1}$ occurs exactly when f_6 is selected as e_1 in Step 2(c)i and f_4 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,1}] = \frac{1}{c} \cdot \frac{1}{2}$. Similarly, event $A_{3,2}$ occurs exactly when f_2 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,2}] = \frac{1}{c} \cdot \frac{1}{2}$. Moreover, event $A_{3,3}$ occurs exactly when one of f_1, f_{11}, f_{15}, $\ldots$, f_{c-3} is selected as e_1 in Step 2(c)i, implying that $\Pr[A_{3,3}] = \frac{k-1}{c}$. Furthermore, event $A_{3,4}$ occurs exactly when f_7 is selected as e_1 in Step 2(c)i and f_5 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,4}] = \frac{1}{c} \cdot \frac{1}{2}$. Therefore, $s_3 = 3 \cdot \frac{1}{2c} + \cdot \frac{k-1}{c} = \frac{2k+1}{2c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{1}{4c} + \frac{2k+1}{8c} \leq \frac{1}{8}$ for $c = 4k+2$.

Case 7.2: $d = 3$. In this case, let $S = \{f_1, f_3, f_4, f_c\}$. Then, $s_3 = \frac{1}{c} \cdot \frac{1}{2}$ because event A_3 occurs exactly when edge f_3 is selected as e_1 in Step 2(c)i and edge f_1 is added to R_i in Step 2(c)iv. Moreover, if event A_2 occurs, exactly one of the following events occurs:

- $A_{2,1}$: $S \cap R_i = \{f_4, f_c\}$ and $f_6 \notin R_i$.
- $A_{2,2}$: $S \cap R_i = \{f_4, f_c\}$ and $f_6 \in R_i$.

Obviously, event $A_{2,1}$ occurs exactly when one of f_{10}, f_{14}, $\ldots$, f_c is selected as e_1 in Step 2(c)i, implying that $\Pr[A_{2,1}] = \frac{k-1}{c}$. Moreover, event $A_{2,2}$ occurs exactly when f_6 is selected as e_1 in Step 2(c)i and f_4 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{2,2}] = \frac{1}{c} \cdot \frac{1}{2}$. Therefore, $s_2 = \frac{k-1}{c} + \frac{1}{2c} = \frac{2k-1}{2c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{2k-1}{4c} + \frac{1}{8c} \leq \frac{1}{8}$ for $c = 4k+2$.

Case 7.3: $d = 1$. In this case, $s_2 = \frac{1}{c} \cdot \frac{1}{2}$ because event A_2 occurs exactly when f_2 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv. Moreover, event A_3 occurs exactly when $\{u, v\} \in R_i$. Therefore, $s_3 = \frac{1}{4}$ by Statement (1) in Lemma 3.3. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{1}{4c} + \frac{1}{16} \leq \frac{1}{8}$ for $c = 4k+2$.

Case 8: C_i is long and $|C_i| \equiv 3 \pmod 4$. Let $k = \frac{c-3}{4}$ and $S = \{f_1, f_6, f_7, f_c\}$. Obviously, $s_2 = 0$ if $d \notin \{2, 3, 6\}$. So, it suffices to consider the following subcases:

Case 8.1: $d = 6$. In this case, let $S = \{f_1, f_6, f_7, f_c\}$. Then, $s_2 = \frac{1}{c} \cdot \frac{1}{4}$ because event A_2 occurs exactly when edge f_7 is selected as e_1 in Step 2(c)i and edge f_4 is not added to R_i in Step 2(c)iv. Moreover, if event A_3 occurs, exactly one of the following events occurs:

- $A_{3,1}$: $S \cap R_i = \{f_7, f_c\}$ and $f_4 \in R_i$.
- $A_{3,2}$: $S \cap R_i = \{f_7, f_c\}$ and $f_3 \in R_i$.

Obviously, event $A_{3,1}$ occurs exactly when f_7 is selected as e_1 in Step 2(c)i and and f_4 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,1}] = \frac{1}{c} \cdot \frac{3}{4}$. Moreover, event $A_{3,2}$ occurs exactly when f_3 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,2}] = \frac{1}{c} \cdot \frac{3}{4}$. Therefore, $s_3 = 2 \cdot \frac{3}{4c} = \frac{3}{2c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{1}{8c} + \frac{3}{8c} \leq \frac{1}{8}$ for $c = 4k+3$.

Case 8.2: $d = 3$. In this case, let $S = \{f_1, f_3, f_4, f_c\}$. Then, if event A_2 occurs, exactly one of the following events occurs:

- $A_{2,1}$: $S \cap R_i = \{f_4, f_c\}$ and $f_7 \notin R_i$.
- $A_{2,2}$: $S \cap R_i = \{f_4, f_c\}$ and $f_7 \in R_i$.

Obviously, event $A_{2,1}$ occurs exactly when one of f_{11}, f_{15}, $\ldots$, f_c is selected as e_1 in Step 2(c)i, implying that $\Pr[A_{2,1}] = \frac{k-1}{c}$. Moreover, event $A_{2,2}$ occurs exactly when f_7 is selected as e_1 in Step 2(c)i and f_4 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{2,2}] = \frac{1}{c} \cdot \frac{3}{4}$. Therefore, $s_2 = \frac{k-1}{c} + \frac{3}{4c} = \frac{4k-1}{4c}$.

Similarly, if event A_3 occurs, exactly one of the following events occurs:

- $A_{3,1}$: $S \cap R_i = \{f_3, f_c\}$.
- $A_{3,2}$: $S \cap R_i = \{f_1, f_4\}$.

Obviously, event $A_{3,1}$ occurs exactly when f_3 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,1}] = \frac{1}{c} \cdot \frac{3}{4}$. Moreover, event $A_{3,2}$ occurs exactly when f_4 is selected as e_1 in Step 2(c)i and f_1 is added to R_i in Step 2(c)iv, implying that $\Pr[A_{3,2}] = \frac{1}{c} \cdot \frac{3}{4}$. Therefore, $s_3 = 2 \cdot \frac{3}{4c} = \frac{3}{2c}$. Consequently, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{4k-1}{8c} + \frac{3}{8c} \le \frac{1}{8}$ for $c = 4k+3$.

Case 8.3: $d = 2$. In this case, $s_2 = \frac{1}{c} \cdot \frac{3}{4}$ because event A_2 occurs exactly when f_3 is selected as e_1 in Step 2(c)i and f_c is added to R_i in Step 2(c)iv. Moreover, $s_3 = 0$. Therefore, $\frac{1}{2}s_2 + \frac{1}{4}s_3 = \frac{3}{8c} \le \frac{1}{8}$ for $c = 4k+3$. □

5 Derandomizing Steps 2 through 6

For convenience, we define a random variable x_i for each $i \in \{1, \ldots, r\}$, as follows:

- If $|C_i| = 3$, then let $x_i = R_i$.
- If $|C_i| \ge 4$, then let x_i denote the pair (e_1, b), where e_1 is the edge randomly selected in Step 2(c)i and b is the number of edges added to R_i in Steps 2(c)iv through 2(c)vi. (*Comment:* $b \le 1$.)

Obviously, x_i has 8 possible values if $|C_i| = 3$; x_i has $|C_i|$ possible values if $|C_i| \ge 4$ and $|C_i| \equiv 0 \pmod 4$; x_i has $2|C_i|$ possible values if $|C_i| \ge 4$ and $|C_i| \not\equiv 0 \pmod 4$.

5.1 Outline of the Derandomization

We may assume that our algorithm processes the cycles in $\mathcal{C}$ in the following order: $C_1, \ldots, C_r$ where the triangles precede the others. Then, the computation till the end of Step 2 can be represented by a rooted tree $\mathcal{T}$ as follows. The root of $\mathcal{T}$ corresponds to C_1 and each child of the root corresponds to C_2. In general, if a node of $\mathcal{T}$ corresponds to C_i $(1 \le i \le r-1)$, then each child of the node in $\mathcal{T}$ corresponds to C_{i+1}.

Consider a cycle $C_i \in \mathcal{C}$. Let μ be a node of $\mathcal{T}$ corresponding to C_i. Let h be the number of possible values of the random variable x_i. Then, μ has h children $\nu_1, \ldots, \nu_h$ in $\mathcal{T}$. Fix an arbitrary one-to-one correspondence between the possible values of x_i and the children of μ. The edge from μ to ν_j $(1 \le j \le h)$ in $\mathcal{T}$ is labeled with the possible value of x_i corresponding to ν_j. This finishes the construction of $\mathcal{T}$.

Fact 5.1 *Let μ be a nonleaf node of $\mathcal{T}$, let C_i be the cycle in $\mathcal{C}$ corresponding to μ, and let $\nu_1, \ldots, \nu_h$ be the children of μ in $\mathcal{T}$. For each $j \in \{1, \ldots, h\}$, let ℓ_j be the label of the edge from μ to ν_j. Define a function q_i as follows: For each $j \in \{1, \ldots, h\}$, $q_i(\ell_j) = \Pr[x_i = \ell_j]$. Then, $\sum_{j=1}^{h} q_i(\ell_j) = 1$.*

The size of $\mathcal{T}$ is exponential and we cannot afford to construct it explicitly. The essence of the pessimistic estimator method is to associate a value to each node of $\mathcal{T}$ satisfying the following four conditions:

(C1) The value of a given node of $\mathcal{T}$ can be computed in polynomial time.

(C2) The value of each leaf node μ of $\mathcal{T}$ is smaller than or equal to $\mathcal{E}[w(E_6) \mid x_1 = \ell_1, \ldots, x_r = \ell_r]$, where $\ell_1, \ldots, \ell_r$ are the labels of the edges on the path from the root to μ in $\mathcal{T}$.

(C3) The value of each nonleaf node μ of $\mathcal{T}$ is smaller than or equal to the largest value of a child of μ in $\mathcal{T}$.

(C4) The value of the root is large enough (compared to the maximum weight of a 2-path packing of G).

Instead of constructing $\mathcal{T}$ explicitly, we only construct one path Q of $\mathcal{T}$ by starting at the root and repeating the following (till arriving at a leaf node of $\mathcal{T}$): Construct the child of the current node whose value is the largest among all the children, and then move to that child. Once we have obtained Q, we start at the root and walk down path Q. While walking down Q, we process $C_1, \ldots, C_r$ where we make our choices according to the labels on the edges of Q (instead of making random choices). In this way, we arrive at a leaf node ν and obtain $R_1, \ldots, R_r$ deterministically. By repeatedly applying Condition (C3), we can see that the value of ν is at least as large as that of the root. Moreover, by Condition (C2), the value of ν is at most as large as $\mathcal{E}[w(E_6) \mid x_1 = \ell_1, \ldots, x_r = \ell_r]$, where $\ell_1, \ldots, \ell_r$ are the labels of the edges in Q. Thus, $\mathcal{E}[w(E_6) \mid x_1 = \ell_1, \ldots, x_r = \ell_r]$ is at least as large as the value of the root, and is hence large enough by Condition (C4).

Now that we have $R_1, \ldots, R_r$, we can proceed to Steps 3 through 5, obtaining R, M, and $\mathcal{C}'$. Obviously, after further performing Step 6, we obtain $\mathcal{C}'$ whose expected weight is $\mathcal{E}[w(E_6) \mid x_1 = \ell_1, \ldots, x_r = \ell_r]$. Fortunately, instead of performing Step 6, we can perform the following (deterministic) step:

6′. For each cycle C in $\mathcal{C}'$, choose the edge in $E(C) \cap M$ of minimum weight and delete it from $\mathcal{C}'$.

After Step 6’, $w(\mathcal{C}')$ is at least as large as $\mathcal{E}[w(E_6) \mid x_1 = \ell_1, \ldots, x_r = \ell_r]$ and is hence at least as large as the value of the root.

5.2 Evaluating a Node of $\mathcal{T}$

When applying the pessimistic estimator method, the difficulty is in how to define the value of a node μ of $\mathcal{T}$.

Let μ be a node of $\mathcal{T}$. Let $C_1, \ldots, C_{i-1}$ be the cycles corresponding to the ancestors of μ in $\mathcal{T}$. Note that $i = 1$ if μ is the root of $\mathcal{T}$, and $i = r + 1$ if μ is a leaf node of $\mathcal{T}$. Let Q be the path from the root to μ in $\mathcal{T}$. Suppose that we process $C_1, \ldots, C_{i-1}$ where we make our choices according to the labels on the edges of Q (instead of making random choices). In this way, we obtain $R_1, \ldots, R_{i-1}$. Based on $R_1, \ldots, R_{i-1}$, we construct an auxiliary graph H_μ as follows:

- $V(H_\mu) = V(G)$ and $E(H_\mu) \subseteq E(\mathcal{C}) \cup M_1$.
- $E(H_\mu) \cap E(\mathcal{C}) = \bigcup_{j=1}^{i-1}(E(C_j) - R_j)$.
- For each edge $\{u, v\} \in M_1$, $e \in E(H_\mu)$ if and only if either both u and v are incident to edges in $\bigcup_{j=1}^{i-1} R_j$, or one of them is incident to an edge in $\bigcup_{j=1}^{i-1} R_j$ and the other is contained in $\bigcup_{j=i}^{r} V(C_j)$.

Note that each connected component of H_μ is a path or cycle. We classify the path components K of H_μ into two types as follows:

- *Type 1*: At least one endpoint of K is contained in $\bigcup_{j=1}^{i-1} V(C_j)$.
- *Type 2:* Both endpoints of K are contained in $\bigcup_{j=i}^{r} V(C_j)$.

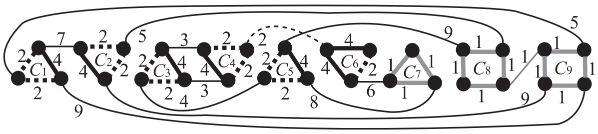

Figure 2: An example of $E(\mathcal{C}) \cup M_1$, where $C_i = C_7$, the edges in M_1 are thin, the edges in $\mathcal{C}$ are bold, the edges in $\bigcup_{j=1}^{i-1} R_j$ are bold broken, the number near each edge is its type, and the edges of H_μ are those of type h with $3 \le h \le 9$.

Moreover, we define the type of each edge $e \in E(\mathcal{C}) \cup M_1$ at μ and assign a coefficient $c_\mu(e)$ to e as follows (see Figure 2):

- *Type 1:* Neither endpoint of e is contained in $\bigcup_{j=1}^{i-1} V(C_j)$. Define $c_\mu(e)$ as follows:
  - If e appears in some triangle C_j, then $c_\mu(e) = 1 - p$.
  - If e appears in some 4^+-cycle C_j, then $c_\mu(e) = \frac{3}{4}$.
  - If $e \in M_1$, then $c_\mu(e) = \frac{3}{16}$.
- *Type 2:* At least one endpoint of e is contained in $\bigcup_{j=1}^{i-1} V(C_j)$ but e is not an edge in H_μ. Define $c_\mu(e) = 0$.
- *Type 3:* $e \in M_1$ and e appears in a cycle of H_μ. Define $c_\mu(e) = \frac{b-1}{b}$, where b is the number of edges in both M_1 and the cycle of H_μ containing e.
- *Type 4:* Either e is an edge of both $\mathcal{C}$ and H_μ, or e is an edge in M_1 and appears in a type-1 path component of H_μ and both endpoints of e are contained in $\bigcup_{j=1}^{i-1} V(C_j)$. Define $c_\mu(e) = 1$.
- *Type 5:* e is an edge in M_1 and appears in a type-1 path component of H_μ and one endpoint of e is contained in some 4^+-cycle C_j with $i \le j \le r$. Define $c_\mu(e) = \frac{1}{2}$.
- *Type 6:* e is an edge in M_1 and appears in a type-1 path component of H_μ and one endpoint of e is contained in some triangle C_j with $i \le j \le r$. Define $c_\mu(e) = 2p - p^2$.
- *Type 7:* e is an edge in M_1 and appears in a type-2 path component of H_μ, and neither endpoint of e is contained in $\bigcup_{j=i}^{r} V(C_j)$. Define $c_\mu(e) = \frac{3}{4}$.
- *Type 8:* e is an edge in M_1 and appears in a type-2 path component of H_μ, and one endpoint of e is contained in some triangle C_j with $i \le j \le r$. Define $c_\mu(e) = \frac{3}{2}p - \frac{1}{2}p^2$.
- *Type 9:* e is an edge in M_1 and appears in a type-2 path component of H_μ, and one endpoint of e is contained in some 4^+-cycle C_j with $i \le j \le r$. Define $c_\mu(e) = \frac{3}{8}$.

Now, we are ready to define the *pessimistic estimator*, namely, a function f mapping each node μ of $\mathcal{T}$ to a real number as follows: $f(\mu) = \sum_{e \in E(\mathcal{C}) \cup M_1} c_\mu(e) w(e)$. We call $f(\mu)$ the *value* of node μ.

5.3 Verifying Conditions (C1) through (C4)

Clearly, $f(\mu)$ can be computed in linear time. Thus, Condition (C1) is satisfied.

To see that Condition (C2) is also satisfied, consider an arbitrary leaf node μ of $\mathcal{T}$. Let $\ell_1, \ldots, \ell_r$ be the labels of the edges on the path from the root to μ in $\mathcal{T}$. For each edge $e \in E(\mathcal{C}) \cup M_1$, $c_\mu(e) = \Pr[e \in E_6 \mid x_1 = \ell_1, \ldots, x_r = \ell_r]$ because e is of type 2, 3, or 4 at μ. Consequently, Condition (C2) is satisfied.

To see that Condition (C3) is also satisfied, it suffices to prove the following two lemmas:

Lemma 5.2 *Let* $\mu, C_i, \nu_1, \ldots, \nu_h, \ell_1, \ldots, \ell_h, q_i$ *be as in Fact 5.1. Suppose that* $|C_i| = 3$*. Then, for every* $e \in E(\mathcal{C}) \cup M_1$, $c_\mu(e) \le \sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e)$.

PROOF. Consider an arbitrary edge $e \in E(\mathcal{C}) \cup M_1$. If the types of e at μ and its children are the same, then by Fact 5.1, $c_\mu(e) = \sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e)$. So, assume that the type of e at μ differs from the type of e at some child of μ. By this assumption, e cannot be of type 2, 3, or 4 at μ. Moreover, since $|C_i| = 3$, e cannot be of type 5 at μ. According to the type of e at μ, we distinguish several cases as follows:

Case 1: e is of type 1 at μ. In this case, one of the following three subcases occurs:

Case 1.1: $e \in E(C_i)$. In this case, for each child ν_j of μ, e can be of type 2 or 4 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = 1 - p$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 4 at ν_j. So, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) = p \cdot 0 + (1-p) \cdot 1 = 1 - p = c_\mu(e)$.

Case 1.2: $e \in M_1$, one endpoint of e appears in C_i, and the other endpoint appears in some triangle C_j with $i + 1 \le j \le r$. In this case, for each child ν_j of μ, e can be of type 2, 6, or 8 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = (1-p)^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 2 at ν_j. For the same reason, $\sum_j q_i(\ell_j) \ge p^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 6 at ν_j. So, $\sum_j q_i(\ell_j) \le 2p(1-p)$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 8 at ν_j. Hence, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge (1-p)^2 \cdot 0 + p^2 \cdot (2p - p^2) + 2p(1-p) \cdot (\frac{3}{2}p - \frac{1}{2}p^2) = 3p^2 - 2p^3 \ge \frac{3}{16} = c_\mu(e)$, where the last inequality holds because of our choice of p.

Case 1.3: $e \in M_1$, one endpoint of e appears in C_i, and the other endpoint appears in some 4^+-cycle C_j with $i + 1 \le j \le r$. In this case, for each child ν_j of μ, e can be of type 2, 5, or 9 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = (1-p)^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 2 at ν_j. For the same reason, $\sum_j q_i(\ell_j) \ge p^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 5 at ν_j. So, $\sum_j q_i(\ell_j) \le 2p(1-p)$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 9 at ν_j. Hence, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge (1-p)^2 \cdot 0 + p^2 \cdot \frac{1}{2} + 2p(1-p) \cdot \frac{3}{8} = \frac{3}{4}p - \frac{1}{4}p^2 \ge \frac{3}{16} = c_\mu(e)$, where the last inequality holds because $p > 0.276$.

Case 2: e is of type 6 at μ. In this case, C_i contains exactly one endpoint of e. Moreover, for each child ν_j of μ, e can be of type 2 or 4 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) \ge 2p - p^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 4 at ν_j. So, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) = (1-p)^2 \cdot 0 + (2p - p^2) \cdot 1 = 2p - p^2 = c_\mu(e)$.

Case 3: e is of type 7 at μ and C_i contains both endpoints of the path component of H_μ containing e. In this case, for each child ν_j of μ, e can be of type 3 or 4 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) \ge 1 - p^2(1-p) - p(1-p)^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 4 at ν_j. Since $c_{\nu_j}(e) \ge \frac{1}{2}$ when e is of type 3 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge (1 - p^2(1-p) - p(1-p)^2) \cdot 1 + (p^2(1-p) + p(1-p)^2) \cdot \frac{1}{2} = 1 - \frac{1}{2}p + \frac{1}{2}p^2 \ge \frac{3}{4} = c_\mu(e)$, where the last inequality holds because $p > 0.276$.

Case 4: e is of type 7 at μ and C_i contains only one endpoint of the path component of H_μ containing e. In this case, for each child ν_j of μ, e can be of type 4 or 7 at ν_j. Since $c_{\nu_j}(e) \ge \frac{3}{4}$ when e is of type 4 or 7 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge \frac{3}{4} = c_\mu(e)$.

Case 5: e is of type 8 at μ and C_i contains both endpoints of the path component of H_μ containing e. In this case, for each child ν_j of μ, e can be of type 2, 3, or 4 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) \ge (2p - p^2) - p^2(1-p) - p(1-p)^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 4 at ν_j. For the same reason, $\sum_j q_i(\ell_j) = p^2(1-p) + p(1-p)^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 3 at ν_j. Since $c_{\nu_j}(e) \ge \frac{1}{2}$ when e is of type 3 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge \left(2p - p^2 - p^2(1-p) - p(1-p)^2\right) \cdot 1 + \left(p^2(1-p) + p(1-p)^2\right) \cdot \frac{1}{2} = \frac{3}{2}p - \frac{1}{2}p^2 = c_\mu(e)$.

Case 6: e is of type 8 at μ, C_i contains only one endpoint u of the path component of H_μ containing e, and u is also an endpoint of e. In this case, for each child ν_j of μ, e can be of type 2, 4, or 7 at ν_j. Because of the way the algorithm processes triangles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = (1-p)^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 2 at ν_j. For the same reason, $\sum_j q_i(\ell_j) \ge p^2$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 4 at ν_j. So, $\sum_j q_i(\ell_j) \le 2p(1-p)$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 7 at ν_j. So, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge p^2 \cdot 1 + 2p(1-p) \cdot \frac{3}{4} = \frac{3}{2}p - \frac{1}{2}p^2 = c_\mu(e)$.

Case 7: e is of type 8 at μ, C_i contains only one endpoint u of the path component of H_μ containing e, and u is not an endpoint of e. In this case, for each child ν_j of μ, e can be of type 6 or 8 at ν_j. Since $c_{\nu_j}(e) \ge \frac{3}{2}p - \frac{1}{2}p^2$ when e is of type 6 or 8 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge \frac{3}{2}p - \frac{1}{2}p^2 = c_\mu(e)$.

Case 8: e is of type 9 at μ. In this case, for each child ν_j of μ, e can be of type 5 or 9 at ν_j. Since $c_{\nu_j}(e) \ge \frac{3}{8}$ when e is of type 5 or 9 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \ge \frac{3}{8} = c_\mu(e)$. □

Lemma 5.3 *Let $\mu, C_i, \nu_1, \ldots, \nu_h, \ell_1, \ldots, \ell_h, q_i$ be as in Fact 5.1. Suppose that $|C_i| \geq 4$. Then, for every $e \in E(\mathcal{C}) \cup M_1$, $c_\mu(e) \leq \sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e)$.*

PROOF. Consider an arbitrary edge $e \in E(\mathcal{C}) \cup M_1$. As in Lemma 5.2, assume that the type of e at μ differs from the type of e at some child of μ. By this assumption, e cannot be of type 2, 3, or 4 at μ. Moreover, since $|C_i| \geq 4$ and the algorithm processes the triangles in $\mathcal{C}$ first, e cannot be of type 6 or 8 at μ. According to the type of e at μ, we distinguish several cases as follows:

Case 1: e is of type 1 at μ. In this case, one of the following two subcases occurs:

Case 1.1: $e \in E(C_i)$. In this case, for each child ν_j of μ, e can be of type 2 or 4 at ν_j. Because of the way the algorithm processes 4^+-cycles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = \frac{3}{4}$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 4 at ν_j. So, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) = \frac{1}{4} \cdot 0 + \frac{3}{4} \cdot 1 = c_\mu(e)$.

Case 1.2: $e \in M_1$ and C_i contains one endpoint of e. In this case, the other endpoint appears in some 4^+-cycle C_j with $i+1 \leq j \leq r$. Moreover, for each child ν_j of μ, e can be of type 2, 5, or 9 at ν_j. Because of the way the algorithm processes 4^+-cycles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = \frac{1}{2}$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is of type 2 at ν_j. Since $c_{\nu_j}(e) \geq \frac{3}{8}$ no matter whether e is of type 5 or 9 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \geq \frac{1}{2} \cdot 0 + \frac{1}{2} \cdot \frac{3}{8} = \frac{3}{16} = c_\mu(e)$.

Case 2: e is of type 5 at μ. In this case, exactly one endpoint of e appears in C_i. Moreover, for each child ν_j of μ, e can be of type 2 or 4 at ν_j. Because of the way the algorithm processes 4^+-cycles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = \frac{1}{2}$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 4 at ν_j. So, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) = \frac{1}{2} \cdot 0 + \frac{1}{2} \cdot 1 = \frac{1}{2} = c_\mu(e)$.

Case 3: e is of type 7 at μ and C_i contains only one endpoint of the path component of H_μ containing e. In this case, for each child ν_j of μ, e can be of type 4 or 7 at ν_j. Since $c_{\nu_j}(e) \geq \frac{3}{4}$ no matter whether e is of type 4 or 7 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \geq \frac{3}{4} = c_\mu(e)$.

Case 4: e is of type 7 at μ and C_i contains both endpoints of the path component of H_μ containing e. In this case, for each child ν_j of μ, e can be of type 3, 4, or 7 at ν_j. Because of the way the algorithm processes 4^+-cycles in $\mathcal{C}$, $\sum_j q_i(\ell_j) \geq \frac{1}{2}$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 4 at ν_j. Since $c_{\nu_j}(e) \geq \frac{1}{2}$ no matter whether e is of type 3 or 7 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \geq \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4} = c_\mu(e)$.

Case 5: e is of type 9 at μ, C_i contains only one endpoint u of the path component of H_μ containing e, and u is not an endpoint of e. In this case, for each child ν_j of μ, e can be of type 5 or 9 at ν_j. Since $c_{\nu_j}(e) \geq \frac{3}{8}$ no matter whether e is of type 5 or 9 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \geq \frac{3}{8} = c_\mu(e)$.

Case 6: e is of type 9 at μ, C_i contains only one endpoint u of the path component of H_μ containing e, and u is an endpoint of e. In this case, for each child ν_j of μ, e can be of type 2, 4, or 7 at ν_j. Because of the way the algorithm processes 4^+-cycles in $\mathcal{C}$, $\sum_j q_i(\ell_j) = \frac{1}{2}$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e of type 2 at ν_j. Since $c_{\nu_j}(e) \geq \frac{3}{4}$ no matter whether e is of type 4 or 7 at ν_j, it follows that $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \geq \frac{1}{2} \cdot \frac{3}{4} = \frac{3}{8} = c_\mu(e)$.

Case 7: e is of type 9 at μ and C_i contains both endpoints of the path component K of H_μ containing e. In this case, for each child ν_j of μ, e can be of type 2, 3, 4, or 7 at ν_j. Let u and v be the endpoints of K. We may assume that u is an endpoint of e but v is not. We say that a child ν_j of μ is *dangerous* for e if u and v are the endpoints of some path component in the graph $(V(C_i), E(C_i) \cap E(H_{\nu_j}))$. Similarly, we say that a child ν_j of μ is *critical* for e if u and v are endpoints of two distinct path components in the graph $(V(C_i), E(C_i) \cap E(H_{\nu_j}))$. For each child ν_j of μ such that e is in H_{ν_j}, consider the following three cases:

- *Case (a):* ν_j is dangerous for e. In this case, e must be of type 3 at ν_j and the cycle of H_{ν_j} containing e contains at least two edges in M_1. So, $c_{\nu_j}(e) \geq \frac{1}{2}$.
- *Case (b):* ν_j is critical for e. In this case, e may be of type 3, 4, or 7 at ν_j. If e is of type 3 at ν_j, then the cycle of H_{ν_j} containing e contains at least four edges in M_1; hence $c_{\nu_j}(e) \geq \frac{3}{4}$. If e is of type 4 or 7 at ν_j, then obviously $c_{\nu_j}(e) \geq \frac{3}{4}$. So, we always have $c_{\nu_j}(e) \geq \frac{3}{4}$.
- *Case (c):* ν_j is neither dangerous nor critical for e. In this case, e is of type 4 at ν_j and so $c_{\nu_j}(e) = 1$.

Now, let $s_1 = \sum_j q_i(\ell_j)$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is in H_{ν_j}. Because of the way the algorithm processes 4^+-cycles in $\mathcal{C}$, $s_1 = \frac{1}{2}$. Let $s_2 = \sum_j q_i(\ell_j)$, where j ranges over all $j \in \{1, \ldots, h\}$ such that e is in H_{ν_j} and ν_j is dangerous for e. Let $s_3 = \sum_j q_i(\ell_j)$, where j ranges over all integers in $\{1, \ldots, h\}$ such that e is in H_{ν_j} and ν_j is critical for e. By Lemma 4.1, $\frac{1}{2}s_2 + \frac{1}{4}s_3 \leq \frac{1}{4}s_1$. Hence, $\sum_{j=1}^{h} q_i(\ell_j) c_{\nu_j}(e) \geq (1 - s_1) \cdot 0 + s_2 \cdot \frac{1}{2} + s_3 \cdot \frac{3}{4} + (s_1 - s_2 - s_3) \cdot 1 \geq \frac{3}{4}s_1 \geq \frac{3}{8} = c_\mu(e)$. $\square$

By Lemmas 5.2 and 5.3,

$$\begin{aligned} f(\mu) &\le \sum_{e\in E(\mathcal{C})\cup M_1} w(e)\sum_{j=1}^{h} q_i(\ell_j)c_{\nu_j}(e) \\ &= \sum_{j=1}^{h} q_i(\ell_j) \sum_{e\in E(\mathcal{C})\cup M_1} c_{\nu_j}(e)w(e) \\ &= \sum_{j=1}^{h} q_i(\ell_j) f(\nu_j). \end{aligned}$$

So, by Fact 5.1, $f(\mu) \le \max_{j=1}^{h} f(\nu_j)$. Consequently, Condition (C3) is satisfied. The following lemma shows that Condition (C4) is also satisfied:

Lemma 5.4 *The value of the root of* $\mathcal{T}$ *is at least* $\left(\frac{3}{4}+(\frac{1}{4}-p)\alpha\right)w(\mathcal{C})+\frac{3}{16}w(M_1)$. *Consequently, it is at least* $\left(\frac{27}{32}-\frac{3}{4}\epsilon-(p-\frac{1}{4})(1-\epsilon)\alpha-\frac{3}{32}\beta\right)w(\mathcal{O}pt)$.

Proof. First note that each edge $e \in E(\mathcal{C}) \cup M_1$ is of type 1 at the root of $\mathcal{T}$. Also recall that $\alpha \cdot w(\mathcal{C})$ is the total weight of edges in the triangles C_i in $\mathcal{C}$. So, the value of the root is at least $(1-p)\alpha \cdot w(\mathcal{C}) + \frac{3}{4}(1-\alpha)\cdot w(\mathcal{C}) + \frac{3}{16}w(M_1) = \left(\frac{3}{4}+(\frac{1}{4}-p)\alpha\right)w(\mathcal{C}) + \frac{3}{16}w(M_1) \ge \left(\frac{3}{4}+(\frac{1}{4}-p)\alpha\right)(1-\epsilon)w(\mathcal{O}pt) + \frac{3}{16}w(M_1)$. As observed in [4], the construction of M_1 clearly implies that $w(M_1) \ge \frac{1}{2}(1-\beta)w(\mathcal{O}pt)$. Thus, the lemma holds. □

6 Analysis of the Approximation Ratio

By Fact 3.4 and Lemma 5.4, the output 2-path packing P_3 satisfies the following inequality:

$$\begin{aligned} & w(P_3) \\ \ge\ & \frac{2}{3}\cdot\left(\frac{27}{32}-\frac{3}{4}\epsilon-(p-\frac{1}{4})(1-\epsilon)\alpha-\frac{3}{32}\beta\right)w(\mathcal{O}pt). \end{aligned}$$

So, by Lemmas 3.1 and 3.2, we have

$$\begin{aligned} & 4(p-\frac{1}{4})w(P_1)+\frac{1}{16}w(P_2)+w(P_3) \\ \ge\ & \frac{1+32p-32p\epsilon}{16}\cdot w(\mathcal{O}pt). \end{aligned}$$

Therefore, the weight of the best packing among P_1, P_2, and P_3 is at least

$$\frac{1+32p-32p\epsilon}{1+64p}\cdot w(\mathcal{O}pt) \ge \frac{1+32p}{1+64p}\cdot(1-\epsilon)w(\mathcal{O}pt).$$

Given G, $\mathcal{C}$ can be computed in $O(|V(G)|^3)$ time [3]. Moreover, given G and $\mathcal{C}$, P_1 and P_2 can be computed in $O(|V(G)|^3)$ time [4]. Furthermore, one can easily verify that P_3 can be computed from G and $\mathcal{C}$ in $O(|V(G)|^2)$ time. So, our deterministic algorithm runs in $O(|V(G)|^3)$ time.

In summary, we have proved the following theorem:

Theorem 6.1 *For any constant* $\epsilon > 0$, *there is a deterministic cubic-time approximation algorithm for* M2PP *that achieves a ratio of* $\frac{1+32p}{1+64p}\cdot(1-\epsilon) > 0.5265\cdot(1-\epsilon)$.

References

[1] Z.-Z. Chen, R. Tanahashi, and L. Wang. An Improved Randomized Approximation Algorithm for Maximum Triangle Packing. *Proceedings of 4th International Conference on Algorithmic Aspects in Information and Management* (AAIM'08), Lecture Notes in Computer Science **5034** (2008) 97-108. The full version will appear in *Discrete Applied Mathematics*.

[2] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, New York, 1979.

[3] D. Hartvigsen. *Extensions of Matching Theory*. Ph.D. Thesis, Carnegie-Mellon University, 1984.

[4] R. Hassin and S. Rubinstein. An Approximation Algorithm for Maximum Triangle Packing. *Discrete Applied Mathematics* **154** (2006) 971-979.

[5] P. Raghavan. Probabilistic Construction of Deterministic Algorithms: Approximating Packing Integer Programs. *Journal of Computer and System Sciences* **37** (1988) 130-143.