Random Methods for Parameterized Problems*

Qilong Feng[1], Jianxin Wang[1], Shaohua Li[1], and Jianer Chen[1,2]

[1] School of Information Science and Engineering,
Central South University,
Changsha 410083, P.R. China

[2] Department of Computer Science and Engineering
Texas A&M University
College Station, Texas 77843-3112, USA

Abstract. In this paper, we study the random methods for parameterized problems. For the Parameterized P_2-Packing problem, by randomly partitioning the vertices, a randomized parameterized algorithm of running time $O^*(6.75^k)$ is obtained, improving the current best result $O^*(8^k)$. For the Parameterized Co-Path Packing problem, we study the kernel and randomized algorithm for the degree-bounded instance, and then by using the iterative compression technique, a randomized algorithm of running time $O^*(3^k)$ is given for the Parameterized Co-Path Packing problem, improving the current best result $O^*(3.24^k)$.

1 Introduction

Random techniques have been used widely in designing parameterized algorithms for many NP-hard problems [2], [3], [11], [12]. In this paper, we are mainly focused on the randomized algorithms for parameterized problems, which the Parameterized P_2-Packing problem and the Parameterized Co-Path Packing problem are used to illustrate the random methods given in this paper. We firstly give definitions of the above two problems.

> Parameterized P_2-Packing: Given a graph $G = (V, E)$ and an integer k, find a set S of P_2s (a P_2 is a simple path of length two) with size k in G such that no two P_2s from S have common vertices, or report that no such set exists.

> Parameterized Co-Path Packing (PCPP): Given a graph $G = (V, E)$ and an integer k, find a subset $F \subseteq V$ of size at most k such that each connected component in graph $G[V \backslash F]$ is a path, or report that no such subset exists.

* This work is supported by the National Natural Science Foundation of China under Grant (61232001, 61103033, 61173051), Postdoc Foundation of Central South University.

D.-Z. Du and G. Zhang (Eds.): COCOON 2013, LNCS 7936, pp. 89–100, 2013.

For the P_2-Packing problem, Hassin and Rubinstein [10] gave an approximation algorithm with ratio 35/67 for the maximum P_2-Packing problem. De Bontridder *et al.* [5] gave an approximation algorithm with ratio 2/3. For a general subgraph H, Fellows *et al.* [6] presented an algorithm of running time $O^*(2^{O(|H|k \log k + k|H| \log |H|)})$ for the H-Packing problem.[1] Prieto and Sloper [13] gave a parameterized algorithm of time $O^*(39.43^k)$ for the Parameterized P_2-Packing problem. Henning *et al.* [7] gave an improved parameterized algorithm with running time $O^*(14.67^k)$. Feng et al. [8] presented a parameterized algorithm with running time $O^*(8^k)$ for the Parameterized P_2-Packing problem, which is the current best result.

The Parameterized Co-Path Packing problem has important application in Bioinformatics [4]. From approximation algorithm point of view, an approximation algorithm with ratio 2 in [9] can be applied to solve the Minimum Co-Path Packing problem. Recently, Chen et al. [1] gave a kernel of size $32k$ for the Parameterized Co-Path Packing problem, and presented an algorithm of running time $O^*(3.24^k)$.

In this paper, for the Parameterized P_2-Packing problem, by using a different way to partition the vertices in the instance, a simple randomized algorithm of running time $O^*(6.75^k)$ is given, which improves the current best result $O^*(8^k)$. For the Parameterized Co-Path Packing problem, the vertices with different degrees are studied. Especially, a randomized algorithm is given for the Parameterized Co-Path Packing problem in degree-bounded graph. By applying iterative compression technique, a randomized algorithm of running time $O^*(3^k)$ is given, which improves the current best result $O^*(3.24^k)$.

2 Randomized Algorithm for Parameterized P_2-Packing

For a P_2 $l = (x, y, z)$, x, z are called the End-vertices of l, and y is called the Mid-vertex of l. For a set $\mathcal{P}$ of P_2s, if no two P_2s in $\mathcal{P}$ have common vertices, then $\mathcal{P}$ is called a P_2-Packing in G. In order to solve the Parameterized P_2-Packing problem efficiently, we introduce the following problem.

> Constrained P_2-Packing on Bipartite Graphs: Given a bipartite graph $B = (L \cup R, E)$ and an integer k, either construct a P_2-Packing $\mathcal{P}$ of size k with End-vertices in L, or report that no such packing exists.

Since the weighted version of Constrained P_2-Packing on Bipartite Graphs can be solved in polynomial time [8], by assigning each edge in B with weight 1, the Constrained P_2-Packing on Bipartite Graphs problem can also be solved in polynomial time.

For a given instance (G, k) of the Parameterized P_2-Packing problem, assume that $\mathcal{P}$ is a P_2-Packing of size k in G. Then, the P_2s in $\mathcal{P}$ have k Mid-vertices and $2k$ End-vertices, denoted by H_1, H_2 respectively. We want to partition the

[1] Following a recent convention, for a function f, we will use the notion $O^*(f)$ for the bound $O(f \cdot n^{O(1)})$.

vertices in V into two parts V_1, V_2, such that H_1 is contained in V_1, and H_2 is contained in V_2. We give the following random strategy to divide the vertices in V. For any vertex v in V, put v into V_1 with probability 2/3, and put v into V_2 with probability 1/3. After deleting all the edges with two ends in V_1 or V_2, an instance $B = (V_1 \cup V_2, E')$ of the Constrained P_2-Packing on Bipartite Graphs problem can be obtained, where E' is the set of edges with one end in V_1 and the other end in V_2, which can be solved in polynomial time. The specific random algorithm solving the Parameterized P_2-Packing problem is given Figure 1.

Algorithm RP2P(G, k)
Input: a graph G, and an integer k
Output: a P_2-Packing of size k, or report no such packing exists.
1. **loop** $c \cdot 6.75^k$ times
1.1 $V_1 = V_2 = \emptyset$;
1.2 **for** each vertex v in V **do**
put v into V_1 with probability 2/3, and put v into V_2 with probability 1/3;
1.3 delete all the edges with two ends either both in V_1 or both in V_2;
1.4 a bipartite graph $B = (V_1 \cup V_2, E')$ can be constructed, where E' is the set of edges with one end in V_1 and the other end in V_2;
1.5 construct a P_2-Packing $\mathcal{P}$ of size k in B with End-points in V_1;
1.6 if step 1.5 is successful, then return $\mathcal{P}$;
2. return("no such packing exists").

Fig. 1. A randomized algorithm for the Parameterized P_2-Packing problem

Theorem 1. *The Parameterized P_2-Packing problem can be solved in time $O^*(6.75)^k$ with probability larger than $1 - (1/e)^c$.*

Proof. If the given instance (G, k) is a No-instance for the Parameterized P_2-Packing problem, then no matter how the vertices in V are partitioned, a P_2-Packing of size k cannot be found in B. Thus, step 2 will correctly return that graph G contains no P_2-Packing of size k.

Now assume that $\mathcal{P}$ is a P_2-Packing of size k in G. Then there are $2k$ End-vertices and k Mid-vertices contained in $\mathcal{P}$. Let H_1, H_2 be the sets of End-vertices, Mid-vertices of $\mathcal{P}$ respectively, $|H_1| = 2k$, and $|H_2| = k$. If the vertices in $\mathcal{P}$ can be partitioned correctly, i.e., all the vertices in H_1 are put into V_1 and all the vertices in H_2 are put into V_2, then a bipartite graph $B = (V_1 \cup V_2, E')$ containing a P_2-Packing of size k can be constructed in step 1.4. Then, the P_2-Packing found in step 1.5 can be correctly returned by step 1.6.

We now analyze the probability that the algorithms fails in finding a P_2-Packing of size k in G. Since each vertex in V is put into V_1, V_2 with probability 2/3, 1/3 respectively, for each iteration of step 1, the vertices in H_1, H_2 can be correctly partitioned in step 1.2 with probability $(2/3)^{2k}(1/3)^k = (4/27)^k$.

Then, the probability that the vertices in H_1, H_2 are not correctly partitioned in each iteration is $1-(4/27)^k$. Therefore, the probability that H_1 and H_2 are not partitioned correctly in $c \cdot 6.75^k$ iterations is $(1-(4/27)^k)^{c \cdot 6.75^k} = ((1-1/6.75^k)^{6.75^k})^c \leq (1/e)^c$. Therefore, the vertices in H_1, H_2 can be partitioned into V_1, V_2 respectively with probability larger than $1-(1/e)^c$.

At last, we analyze the time complexity of algorithm **RP2P**. Step 1.2 can be done in $O(n)$ time, where n is the number of vertices in G. It takes $O(n+m)$ time to do steps 1.3, 1.4, where m is the number of edges in G. For step 1.5, a P_2 Packing of size k in G can be found in $O(k(m'+nlogn)$ [8], where m' is the number of edges in B. Therefore, the running time of algorithm **RP2P** is $O(6.75^k k(m'+nlogn)) = O^*(6.75^k)$. □

3 Randomized Algorithm for Parameterized Co-path Packing Problem

For a vertex v, let $N(v)$ denote the neighbors of vertex v, i.e., $N(v) = \{u|(u,v) \in E\}$, and let $N[v] = N(v) \cup \{v\}$. For a graph $G=(V,E)$, and a subset $V' \subseteq V$, let $G[V']$ denote the subgraph induced by the vertices in V'. In order to solve the problem efficiently, we first study the Parameterized Co-Path Packing problem in degree bounded graph.

> Degree-Bounded Parameterized Co-Path Packing (DBPCP): Given a graph $G=(V,E)$ and an integer k, where each vertex v in V has degree at most three, find a subset $F \subseteq V$ of size at most k such that each connected component in graph $G[V \backslash F]$ is a path, or report that no such subset exists.

3.1 Kernelizaiton Algorithm for DBPCP

For the Degree-Bounded Parameterized Co-Path Packing problem, some structure properties can be obtained, as follows.

Lemma 1. *Given an instance (G,k) of Degree-Bounded Parameterized Co-Path Packing problem, the number of vertices with degree three in G is bounded by $4k$.*

Given an instance (G,k) of Degree-Bounded Parameterized Co-Path Packing problem, for a path $P=\{v_1,v_2,\cdots,v_{h-1},v_h\}$ in G, if the vertices in $\{v_2,\cdots,v_{h-1}\}$ have degree two and the degrees of v_1,v_h are not two, then path P is called a *degree-two-path*.

Lemma 2. *For a degree-two-path $P=\{v_1,v_2,\cdots,v_{h-1},v_h\}$ in G, if the two vertices v_1, v_h both have degree three and $h>4$, then edges in $\{(v_i,v_{i+1})|i=2,3,\cdots,h-2\}$ can be contracted; if one vertex of $\{v_1,v_h\}$ has degree one and the other has degree three, then edges in $\{(v_i,v_{i+1})|i=1,2,\cdots,h-2\}$ can be contracted.*

Based on Lemma 2, we can get the following reduction rule.

Rule 1. For a degree-two-path $P = \{v_1, v_2, \cdots, v_{h-1}, v_h\}$ in G, if the two vertices v_1, v_h both have degree three and $h > 4$, then contract the edges in $\{(v_i, v_{i+1}) | i = 2, 3, \cdots, h-2\}$. If one vertex of $\{v_1, v_h\}$ has degree one and the other has degree three, then contract the edges in $\{(v_i, v_{i+1}) | i = 1, 2, \cdots, h-2\}$.

Let $G' = (V', E')$ be the graph obtained after applying reduction Rule 1.

Lemma 3. *In the graph G', the number of vertices with degree two is bounded by $12k$.*

Lemma 4. *For any vertex v with degree three in G', if there are two vertices u, w with degree one in $N(v)$, then the vertex in $N(v) \backslash \{u, w\}$ can be deleted.*

Based on Lemma 4, we can get the following reduction rule.

Rule 2. For any vertex v with degree three in G', if there are two vertices u, w with degree one in $N(v)$, then delete the vertex in $N(v) \backslash \{u, w\}$, $V' = V' \backslash \{u, v, w\}$, $k = k - 1$.

Let $G'' = (V'', E'')$ be the graph obtained after applying reduction Rule 2.

Lemma 5. *In the graph G'', the number of vertices with degree one is bounded by $4k$.*

By applying reduction Rule 1 and Rule 2 repeatedly, a kernel of size $20k$ for the Degree-Bounded Parameterized Co-Packing problem can be obtained.

Theorem 2. *The Degree-Bounded Parameterized Co-Path Packing problem admits a kernel of size $20k$.*

3.2 Randomized Algorithm for DBPCP

Assume that $(G = (V, E), k)$ is a reduced instance of the Degree-Bounded Parameterized Co-Path Packing problem by repeatedly applying reduction Rule 1 and Rule 2, and assume that F ($|F| \leq k$) is a solution of the Degree-Bounded Parameterized Co-Path Packing problem. Let E' be the set of edges in E with at least one endpoint having degree three. The edges in E' can be divided into two parts A, B such that $A \cup B = E'$, and for each edge e in A, at least one endpoint of e is contained in F, and for each edge e' in B, no endpoint of e is contained in F.

Lemma 6. *In the reduced graph G, for the sets A, B, $|A|/|B| \geq 1/2$.*

Lemma 7. *For an edge $e = (u, v) \in B$, and for any vertex x from $\{u, v\}$ with degree three, then at least one vertex in $N(x) \backslash (\{u, v\} \backslash \{x\})$ is contained in F.*

The general idea to randomly solve the Degree-Bounded Parameterized Co-Path Packing problem is as follows. Arbitrarily choose an edge e from E'. Then, with probability $|A|/|E'|$, the edge e is from A, and with probability $|B|/|E'|$, edge e is from B. If the edge e is from A, then at least one endpoint of e is contained

Algorithm R-DBPCP(G, k)
Input: a graph G, and an integer k
Output: a subset $F \subseteq V$ of size at most k such that each component in $G[V \backslash F]$ is a path, or report no such subset exists.
1. **for** each k_1, k_2 with $k_1 + k_2 \leq k$ **do**
2. **loop** $c \cdot 2^{k_1}$ times
2.1 $F = C = D = \emptyset$;
2.2 let E' be the set of edges in E with at least one endpoint having degree three;
2.3 **while** E' is not empty **do**
2.4 randomly choose an edge $e = (u, v)$ from E';
2.5 put e into C with probability 1/2, and put e into D with probability 1/2;
2.6 **if** e is contained in C **then**
2.7 randomly choose a vertex from $\{u, v\}$ to put into F;
2.8 **if** e is contained in D **then**
2.9 find a vertex w from $\{u, v\}$ with degree three;
2.10 randomly choose a vertex y from $N(w) \backslash (\{u, v\} \backslash \{w\})$ to put into F;
2.11 let E' be the set of edges in $G[V \backslash F]$ with at least one endpoint having degree three;
2.12 **if** $|F| \leq k_1$ **then**
2.13 denote the remaining graph by G';
2.14 **if** the number of cycles in G' is at most k_2 **then**
2.15 **for** each cycle C in G' **do**
2.16 delete a vertex v' from C, and add v' to F;
2.17 return(F); break;
3. return("no such subset exists").

Fig. 2. A randomized algorithm for DBPCP problem

in F. Randomly choose an endpoint of e to put into F. If the edge e is from B, find an endpoint x of e with degree three, and randomly choose a vertex from $N(x) \backslash (\{u, v\} \backslash \{x\})$ to put into F. The specific randomized algorithm solving the Degree-Bounded Parameterized Co-Path Packing problem is given in Figure 2.

Theorem 3. *The Degree-Bounded Parameterized Co-path Packing problem can be solved randomly in time* $O^*(2^k)$.

Proof. First note that if the input instance is a no-instance, step 2 could not find a subset $F \subseteq V$ with size at most k such that in graph $G[V \backslash F]$, each component is a path, which is rightly handled by step 3.

Now suppose that a subset $F \subseteq V$ can be found in G such that each component is a path in $G[V \backslash F]$. Then, there must exist two subsets $F', F'' \subseteq F$ ($F' \cup F'' = F$) such that in $G[V \backslash F']$, each vertex has degree at most two, and after deleting the edges in F'', all the cycles in $G[V \backslash F']$ are destroyed. Thus, there must exist k_1, k_2 with $k_1 + k_2 \leq k$ such that $|F'| = k_1$, $|F''| = k_2$.

By arbitrarily choose an edge e from E', with probability $|A|/|E'|$, the edge e is from A, and with probability $|B|/|E'|$, edge e is from B. By Lemma 7, for the sets A, B, $|A|/|B| \geq 1/2$. Therefore, the probability that edge e is from A is at least 1/2. In step 2.5, edge e is put into C with probability 1/2, and is put into D with probability 1/2. Since at least one endpoint of edge e has degree three, at least one vertex from $N[u] \cup N[v]$ is contained in F. Assume that $\{v_1, \cdots, v_i\}$ $(1 \leq i)$ is the subset of vertices from $N[u] \cup N[v]$, which are contained in F. We now prove that by steps 2.5-2.10, one vertex from $\{v_1, \cdots, v_i\}$ $(1 \leq i)$ can be rightly added into F with probability 1/2. If the edge e picked in step 2.4 is from A, then with probability 1/2, e is put into C. Without loss of generality, assume that $F \cap \{u, v\}$ is $\{u\}$. Then, in step 2.7, for the vertices $\{u, v\}$, u can be rightly added into F with probability 1/2. Thus, if the edge e picked in step 2.4 is from A, then with probability 1/4, u can be rightly added into F in step 2.7. On the other hand, if the edge e picked in step 2.4 is from B, then with probability 1/2, e is put into D in step 2.5. In this case, no vertex from $\{u, v\}$ is contained in F. Without loss of generality, assume that vertex u has degree three. Consequently, at least one vertex from $N(u)\backslash\{v\}$ is added into F. Assume that vertex x from $N(u)\backslash\{v\}$ is contained in F. Then, in step 2.10, for the vertices in $N(u)\backslash\{v\}$, vertex x can be rightly added into F with probability 1/2. Therefore, if the edge e picked in step 2.4 is from B, then with probability 1/4, vertex x is rightly added into F. Thus, for the edge e picked in step 2.4, the probability that at least one vertex in $\{v_1, \cdots, v_i\}$ is rightly put into F is $1/4+1/4 = 1/2$. Then, the probability that all vertices in F_1' are deleted is at least $(1/2)^{k_1}$. Therefore, the probability that the vertices in F_1' are not rightly deleted is $1-(1/2)^{k_1}$. Therefore, after $c \cdot 2^{k_1}$ operations, none of the executions of steps 2.1-2.11 can rightly handle the vertices in F_1' is $(1-(1/2)^{k_1})^{c \cdot 2^{k_1}} = ((1-(1/2)^{k_1})^{2^{k_1}})^c \leq (1/e)^c$. Therefore, after $c \cdot 2^{k_1}$ operations, the algorithm can correctly handle the vertices in F_1' with probability larger than $1-(1/e)^c$.

For each loop in step 2, if $|F| \leq k_1$ in step 2.12 and there exist cycles in the remaining graph, the cycles can be destroyed by choosing any vertex in the cycle.

Step 2.2 can be done in time $O(n+m)$, and step 2.3 can be done in $O(m(m+n))$, where m, n are the number of edges and vertices in graph G respectively. Step 2.12-2.16 can be done in time $O(n+m)$. Therefore, algorithm R-DBPCP runs in time $O(2^{k_1} m(n+m)) = O^*(2^k)$. □

4 Randomized Algorithm for the PCPP Problem

In this section, using iterative compression technique, we give a randomized algorithm of running time $O^*(3^k)$ for the Parameterized Co-Path Packing problem. We first give the following definitions.

Parameterized Co-Path Packing Compression (PCPPC): Given a graph $G = (V, E)$, an integer k, and a subset $F \subseteq V$ of size $k+1$, where in graph $G[V\backslash F]$, each connected component is a path, find a subset $F' \subseteq V$ of size at most k such that each connected component in graph $G[V \backslash F']$ is a path, or report that no such subset exists.

Special Parameterized Co-Path Packing Compression (SPCPPC): Given a graph $G = (V, E)$, an integer k, and a subset $F \subseteq V$ of size $k+1$, where in graph $G[V \backslash F]$, each connected component is a path, find a subset $F' \subseteq V$ of size at most k such that $F' \cap F = \emptyset$, and each connected component in graph $G[V \backslash F']$ is a path, or report that no such subset exists.

We now give an algorithm solving the Special Parameterized Co-Path Packing Compression problem.

Lemma 8. *Given an instance $(G = (V, E), k, F)$ of the Special Parameterized Co-Path Packing Compression problem, if there exists a vertex v in $V \backslash F$ with degree at least five, then v can be deleted; if there exists a vertex v in $V \backslash F$ with degree four and $|N(v) \cap F| = 3$, then v can be deleted.*

For the instance $(G = (V, E), k, F)$ of the Special Parameterized Co-Path Packing Compression problem, if there exists a vertex v in F with degree at least three, then v is called a *special vertex* in F.

Lemma 9. *For a special vertex v in F, if $|F \cap N(v)| = 2$, then the vertices in $N(v) \backslash F$ can be deleted; if $|F \cap N(v)| = 1$, then at least $|N(v) \backslash F| - 1$ vertices in $N(v) \backslash F$ can be deleted.*

Given an instance $(G = (V, E), k, F)$ of the Special Parameterized Co-Path Packing Compression problem, assume that F' is the solution of the problem. In the following, we first give the algorithm to deal with the vertices with degree four in $V \backslash F$ and having two neighbors in $V \backslash F$: for a vertex v with degree four in $V \backslash F$ and and having two neighbors in $V \backslash F$, either v is contained in F, or the two vertices in $N(v) \backslash F$ are contained in F. The specific algorithm is given in Figure 3.

Theorem 4. *Algorithm Bran-V$(G, k_1, F, \emptyset)$ runs in time $O(1.62^{k_1} n)$ and returns a collection of at most 1.62^{k_1} sets of vertices, where n is the number of vertices in G.*

Proof. For an instance $(G = (V, E), k, F)$ of the Special Parameterized Co-Path Packing Compression problem, assume that F' is the solution of the problem. For a vertex v of $V \backslash F$ with degree four and having two neighbors in $V \backslash F$, assume that two neighbors u, w of v are contained in $V \backslash F$. Either vertex v is in F' or the vertices u, w are in F', which corresponds the two branchings in step 4 and step 5 respectively. Assume that V' is the set of all vertices with degree four in G, and let $V'' \subseteq V'$ be the set of vertices in $\bigcup_{v \in V'} N[v]$ that are contained in F'. Let $k_1 = |V''|$ and let $T(k_1)$ be the size of search tree obtained by calling algorithm Bran-V(G, k_1, F, Q). It is easy to get the following recurrence relation: $T(k_1) = T(k_1 - 1) + T(k_1 - 2)$. Then, $T(k_1) = 1.62^{k_1}$. Therefore, algorithm Bran-V$(G, k_1, F, \emptyset)$ runs in time $O(1.62^{k_1} n)$ and returns a collection of at most 1.62^{k_1} sets of vertices. □

Algorithm Bran-V(G, k_1, F, Q)
Input: a graph $G = (v, E)$, an integer k, a subset F of vertices, and a subset Q of vertices
Output: the collection of vertices in $V \backslash F$, each contains k_1 vertices
1. **if** $(|Q| > k_1)$ **then** abort; **else** return Q;
2. arbitrarily pick a vertex v in $V \backslash F$ with degree four and having two neighbors in $V \backslash F$;
3. let u, w be the two vertices in $N(v) \backslash F$;
4. Bran-V$(G \backslash \{v\}, k_1, F, Q \cup \{v\})$;
5. Bran-V$(G \backslash \{u, v\}, k_1, F, Q \cup \{u, v\})$;

Fig. 3. Branching of vertices with degree four

For an instance $(G = (V, E), k, F)$ of the Special Parameterized Co-Path Packing Compression problem, if all the vertices in G have degree at most three, the algorithm R-DBPCP can be modified slightly to find the solution of the problem.

Theorem 5. *For an instance* $(G = (V, E), k, F)$ *of the Special Parameterized Co-Path Packing Compression problem, if all the vertices in* G *have degree at most three, then a subset* F' *of size* k *can be found in* $O^*(2^k)$ *time with probability larger than* $1 - (1/e)^c$ *such that* $F \cap F' = \emptyset$ *and each connected component in* $G[V \backslash F']$ *is a path.*

The proof of Theorem 5 is similar to the proof of Theorem 3, which is neglected here.

The general idea solving the Special Parameterized Co-Path Packing Compression problem is that: the vertices with degree large than four, and the special vertices are handled firstly. Then, by calling algorithm Bran-V$(G, k_1, F, \emptyset)$, the vertices with degree four and having two neighbors in $V \backslash F$ can be dealt with. Then, in the remaining graph, each vertex has degree at most three, which can be handled by algorithm R-DBPCP. The specific algorithm solving the Special Parameterized Co-Path Packing Compression problem is given in Figure 4.

Theorem 6. *The Special Parameterized Co-Path Packing Compression problem can be solved randomly in* $O^*(2^k)$ *time.*

Proof. If the input instance is a no-instance, step 5 could not find a subset $F' \subseteq V$ with size at most k such that $F' \cap F = \emptyset$, and in graph $G[V \backslash F']$, each component is a path, which is rightly handled by step 6.

Now suppose that a subset $F' \subseteq V$ can be found in G such that $F' \cap F = \emptyset$, and in graph $G[V \backslash F']$, each component is a path. Then, there must exist three subsets F'_1, F'_2, F'_3 ($|F'_1| = k_1$, $|F'_2| = k_2$, $|F'_3| = k_3$,) such that $F'_1 + F'_2 + F'_3 = F'$ and F'_1, F'_2, F'_3 have the following properties: (1) after deleting the vertices in F'_1, there does not exist a special vertex in the remaining graph; (2) after deleting the vertices in $F'_1 \cup F'_2$, all the vertices in the remaining graph have degree bounded by three.

Algorithm R-SPCPPC(G, k, F)
Input: a graph G, and an integer k
Output: a subset $F' \subseteq V$ of size at most k such that $F \cap F' = \emptyset$ and each component in $G[V \backslash F']$ is a path, or report no such subset exists.
1. $F' = \emptyset$;
2. **if** there exists a vertex v in $V \backslash F$ with degree at least five **then**
   $F' = F' \cup \{v\}$, $V = V \backslash \{v\}$, $k = k - 1$;
3. **if** there exists a vertex v in $V \backslash F$ with degree four and $|N(v) \cap F| = 3$ **then**
   $F' = F' \cup \{v\}$, $V = V \backslash \{v\}$, $k = k - 1$;
4. **if** there exists a special vertex v with $|F \cap N(v)| = 2$ **then**
   $F' = F' \cup (N(v) \backslash F)$, $V = V \backslash (N(v) \backslash F)$, $k = k - |N(v) \backslash F|$;
5. **for** each k_1, k_2, k_3 with $k_1 + k_2 + k_3 \leq k$ **do**
5.1. **loop** $c \cdot 2^{k_1}$ times
5.2. $H = H'' = \emptyset$;
5.3. let V' be the set of special vertices such that for each v in V', $|F \cap N(v)| = 1$;
5.4. **while** V' is not empty **do**
5.5. randomly choose a vertex u from $N(v) \backslash F$, and put u into H;
5.6. let V' be the set of special vertices in $G[V \backslash (F \cup H)]$ such that for each v in V', $|F \cap N(v)| = 1$;
5.7. **if** $|H| \leq k_1$ **then**
5.8. call algorithm Bran-V$(G[V \backslash H], k_2, F, \emptyset)$;
5.9. **for** each set H' returned by algorithm Bran-V$(G, k_2, F, \emptyset)$ **do**
5.10. let $G' = (V', E') = G[V \backslash (H \cup H')]$;
5.11. let H'' be the set obtained by calling algorithm R-DBPCP;
5.12. **if** $|H| + |H'| + |H''| \leq k$ **then**
   $F' = F' \cup (H \cup H' \cup H'')$;
5.13. return(F'); stop the loop in step 5.1;
6. return("no such subset exists").

Fig. 4. A randomized algorithm for SPCPPC problem

The correctness of steps 2-4 can be obtained directly by Lemma 8 and Lemma 9. For a special vertex v with $|F \cap N(v)| = 1$, by Lemma 9, at least $|N(v) \backslash F| - 1$ vertices in $N(v) \backslash F$ can be deleted. Assume that $T \subseteq N(v) \backslash F$ are the subset of vertices contained in F'. Since the degree of v is at least three, by randomly choosing a vertex u from $N(v) \backslash F$, the probability that u is from T is at least 1/2. Then, the probability that all vertices in F'_1 are deleted is at least $(1/2)^{k_1}$, i.e., when steps 5.4-5.6 are done, all the special vertices are destroyed with probability $(1/2)^{k_1}$. In step 5.8, algorithm Bran-V($G[V \backslash H], k_2, F, \emptyset$) is called to deal with the vertices with degree four and $|N(v) \cap (V \backslash F)| = 2$. By Theorem 4, a collection of at most 1.62^{k_1} sets of vertices can be returned. Then, in step 5.10, there does not exist a vertex with degree four. Therefore, algorithm R-DBPCP can be used to find a subset $F'_3 \subseteq V'$ such that $F'_3 \cap F = \emptyset$ and each connected component in $G'[V' \backslash F'_3]$ is a path. By Theorem 5, the subset F'_3 can be found with probability larger than $1 - (1/e)^c$. Therefore, in step 5.12, if $|H| + |H'| + |H''| \leq k$, then a solution F' can be returned such that $F' \cap F = \emptyset$ and each connected component in $G[V \backslash F']$ is a path. For the step 5.1, the probability that the vertices in F'_1 are not rightly deleted is $1 - (1/2)^{k_1}$. Therefore, after $c \cdot 2^{k_1}$ operations, the probability that none of the executions of steps 5.5-5.6 can rightly handle the vertices in F'_1 is $(1 - (1/2)^{k_1})^{c \cdot 2^{k_1}} = ((1 - (1/2)^{k_1})^{2^{k_1}})^c \leq (1/e)^c$. Therefore, after $c \cdot 2^{k_1}$ operations, the algorithm can correctly handle the vertices in F'_1 with probability larger than $1 - (1/e)^c$.

Now we analyze the running time of algorithm R-SPCPPC(G, k, F). Steps 2-4 can be done in $O(n+m)$ time. The while loop in step 5.4 can be done in $O(n(n+m))$ time. By Theorem 4, algorithm Bran-V($G, k_1, F, \emptyset$) runs in time $O(1.62^{k_1} n)$, and by Theorem 5, algorithm R-DBPCP can be done in time $O^*(2^{k_3})$. Therefore, step 5 can be done in $O(2^{k_1} 1.62^{k_2} 2^{k_3} k^3 (n+m)) = O(2^{k_1+k_2+k_3} k^3 (n+m)) = O^*(2^k)$. □

Based on the algorithm solving the Special Parameterized Co-Path Packing Compression problem, the Parameterized Co-Path Packing Compression problem can be solved.

Theorem 7. *The Parameterized Co-Path Packing Compression problem can be solved randomly in $O^*(3^k)$ time.*

Based on the iterative compression technique, the Parameterized Co-Path Packing problem can be solved.

Theorem 8. *The Parameterized Co-Path Packing problem can be solved randomly in $O^*(3^k)$ time.*

5 Conclusion

In this paper, we study the randomized techniques for the parameterized problems. For the P_2-Packing problem, a randomized algorithm of running time $O^*(6.75^k)$ is given, improving the current best result $O^*(8^k)$. For the Parameterized Co-Path Packing problem, a randomized algorithm of running time

$O^*(3^k)$ is given, improving the current best result $O^*(3.24^k)$. How to apply the randomized methods in this paper to solve other parameterized problems is an interesting topic, which is also our future research.

References

1. Chen, Z.-Z., Fellows, M., Fu, B., Jiang, H., Liu, Y., Wang, L., Zhu, B.: A Linear Kernel for Co-Path/Cycle Packing. In: Chen, B. (ed.) AAIM 2010. LNCS, vol. 6124, pp. 90–102. Springer, Heidelberg (2010)
2. Chen, J., Lu, S.: Improved parameterized set splitting algorithms: A probabilistic approach. Algorithmica 54(4), 472–489 (2008)
3. Chen, J., Lu, S., Sze, S.H., Zhang, F.: Improved algorithms for path, matching, and packing problems. In: Proc. of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), pp. 298–307 (2007)
4. Chauve, C., Tannier, E.: A methodological framework for the reconstruction of contiguous regions of ancestral genomes and its application to mammalian genome. PLoS Comput. Biol. 4, e1000234 (2008)
5. De Bontridder, K., Halldórsson, B., Lenstra, J., Ravi, R., Stougie, L.: Approximation algorithms for the test cover problem. Math. Program., Ser. B 98, 477–491 (2003)
6. Fellows, M., Heggernes, P., Rosamond, F., Sloper, C., Telle, J.A.: Finding k disjoint triangles in an arbitrary graph. In: Hromkovič, J., Nagl, M., Westfechtel, B. (eds.) WG 2004. LNCS, vol. 3353, pp. 235–244. Springer, Heidelberg (2004)
7. Fernau, H., Raible, D.: A parameterized perspective on packing paths of length two. Journal of Combinatorial Optimization 18(4), 319–341 (2009)
8. Feng, Q., Wang, J., Chen, J.: Matching and P_2-Packing: Weighted Versions. In: Fu, B., Du, D.-Z. (eds.) COCOON 2011. LNCS, vol. 6842, pp. 343–353. Springer, Heidelberg (2011)
9. Fujito, T.: Approximating node-deletion problems for matroidal properties. J. Algorithms 31, 211–227 (1999)
10. Hassin, R., Rubinstein, S.: An approximation algorithm for maximum triangle packing. Discrete Appl. Math. 154, 971–979 (2006)
11. Marx, D., Razgon, I.: Fixed-parameter tractability of multicut parameterized by the size of the cutset. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC 2011), pp. 469–478 (2011)
12. Marx, D.: Randomized Techniques for Parameterized Algorithms. In: Thilikos, D.M., Woeginger, G.J. (eds.) IPEC 2012. LNCS, vol. 7535, p. 2. Springer, Heidelberg (2012)
13. Prieto, E., Sloper, C.: Looking at the stars. Theoretical Computer Science 351, 437–445 (2006)