Approximation Algorithms for Bounded Degree Phylogenetic Roots

Zhi-Zhong Chen
Department of Mathematical Sciences, Tokyo Denki University
Hatoyama, Saitama 350-0394, Japan.
Email: chen@r.dendai.ac.jp
Phone: +81-49-296-2911 Fax: +81-49-296-7072

Abstract

The Degree-Δ Closest Phylogenetic kth Root Problem (ΔCPR$_k$) is the problem of finding a (phylogenetic) tree T from a given graph $G = (V, E)$ such that (1) the degree of each internal node in T is at least 3 and at most Δ, (2) the external nodes (*i.e.* leaves) of T are exactly the elements of V, and (3) the number of disagreements, i.e., $|E \oplus \{\{u, v\} : u, v \text{ are leaves of } T \text{ and } d_T(u, v) \leq k\}|$, is minimized, where $d_T(u, v)$ denotes the distance between u and v in tree T. This problem arises from theoretical studies in evolutionary biology and generalizes several important combinatorial optimization problems such as the maximum matching problem. Unfortunately, it is known to be NP-hard for all fixed constants Δ, k such that either both $\Delta \geq 3$ and $k \geq 3$, or $\Delta > 3$ and $k = 2$. This paper presents a polynomial-time 8-approximation algorithm for ΔCPR$_2$ for any fixed $\Delta > 3$, a quadratic-time 12-approximation algorithm for 3CPR$_3$, and a polynomial-time approximation scheme for the maximization version of ΔCPR$_k$ for any fixed Δ and k.

Keywords: Phylogenies, phylogenetic roots, computational biology, approximation algorithms, randomized algorithms, graph algorithms.

1 Introduction

A phylogeny is a tree where the leaves are labeled by species and each internal node represents a speciation event whereby an ancestral species gives rise to two or more child species. The internal nodes of a phylogeny have degrees (in the sense of unrooted trees, *i.e.* the number of incident edges) at least 3. Proximity within a phylogeny in general corresponds to similarity in evolutionary characteristics. Lin *et. al.* [7] investigated the computational feasibility of reconstructing phylogenies from similarity data via a graph-theoretic approach. Specifically, interspecies similarity is represented by a graph G where the vertices are the species and the adjacency relation represents evidence of evolutionary

similarity. A *phylogeny* T is then reconstructed from G such that (1) the leaves of T are the vertices of G (*i.e.* species), (2) the degree of each internal node of T is at least 3, and (3) for any two vertices u and v of G, they are adjacent in G if and only if $d_T(u, v) \leq k$, where $d_T(u, v)$ denotes the distance between u and v in T and k is a predetermined proximity threshold.

However, graph G is derived from some similarity data, which is usually inexact in practice and may have erroneous (spurious or missing) edges. Such errors may cause G to have no phylogeny, and hence we are interested in finding an *approximate phylogeny* for G which is just a tree whose leaves are exactly the vertices of G and whose internal nodes each are of degree at least 3. For a constant $k \geq 2$, a *k-disagreement* between G and its approximate phylogeny T is an unordered pair $\{u, v\}$ of vertices of G such that either (1) $\{u, v\} \in E$ and $d_T(u, v) > k$, or (2) $\{u, v\} \notin E$ and $d_T(u, v) \leq k$. For each constant $k \geq 2$, Chen *et al.* [3] introduced the Closest Phylogenetic kth Root Problem (CPR_k) which asks for an approximate phylogeny T of a given graph G with the fewest k-disagreements (see Figure 1 for an example). They [3] showed that CPR_k is NP-hard for all $k \geq 2$.

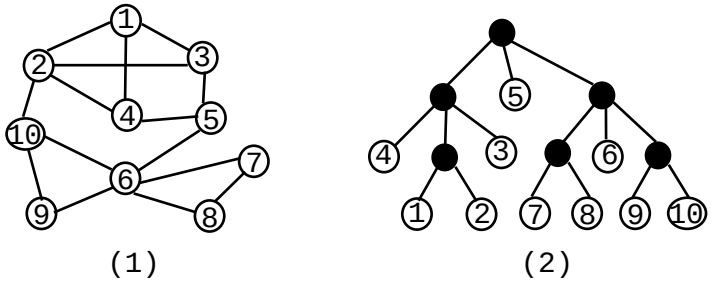

Figure 1: (1) A given graph G. (2) An approximate phylogeny of G with the fewest (namely, two) 3-disagreements.

To be clear, we hereafter call the vertices in the input graph G of CPR_k *vertices* while those in the output approximate phylogeny T *nodes*. If an approximate phylogeny has no node of degree larger than an integer Δ, then it is called an *approximate Δ-phylogeny*.

In the practice of phylogeny reconstruction, most phylogenies considered are trees of degree 3 [10], because speciation events are usually bifurcating events in the evolutionary process. More specifically, in such phylogenetic trees, each internal node has three neighbors and represents a speciation event that some ancestral species splits into two child species. Nodes of degrees higher than 3 are introduced only when the input biological (similarity) data are not sufficient to separate individual speciation events and hence several such events may be collapsed into a non-bifurcating (super) speciation event in the reconstructed phylogeny. These motivated Chen *et al.* [3] to consider a restricted version $\Delta\mathrm{CPR}_k$ of CPR_k for each fixed constant $\Delta \geq 3$ where the output must be an approximate Δ-phylogeny. Tsukiji and Chen [11] showed that $\Delta\mathrm{CPR}_k$ is NP-hard if either both $\Delta \geq 3$

and $k \geq 3$, or $\Delta \geq 4$ and $k = 2$.

1.1 Previous Results on CPR_k and Related Problems

Of special interest is CPR_2. CPR_2 is closely related to the *correlation clustering* problem which has drawn much attention recently (see [1] and the references therein). In the correlation clustering problem, we are required to modify a given graph G into a *cluster graph* by deleting and/or adding the fewest edges, where a cluster graph is a graph in which each connected component is a clique. Clearly, CPR_2 can be reworded as follows: Given a graph G, modify G into a connected cluster graph or a disconnected cluster graph with at least two connected components of size 2 or more, by deleting and/or adding the fewest edges. To the best of our knowledge, the best ratio achieved by (deterministic) polynomial-time approximation algorithms for the correlation clustering problem is 4 [2].

$3\mathrm{CPR}_2$ is essentially identical to the fundamental *maximum matching* problem for the following reason: A maximum matching of a given graph G can be easily retrieved from an approximate 3-phylogeny of G with the fewest 2-disagreements, and *vice versa*. Many efficient algorithms are known for the maximum matching problem in the literature.

Shamir *et al.* [9] study three problems related to CPR_2, called the *cluster editing*, the *cluster deletion*, and the *cluster completion* problems, respectively. Coincidentally, the cluster editing problem is the same as the correlation clustering problem. In the cluster deletion (respectively, completion) problem, we are required to remove from (respectively, add to) G the fewest edges so that it becomes a cluster graph. They show that the cluster completion problem can be solved in polynomial time while the other two are NP-hard. They also study the *p-cluster versions* of the problems where the output cluster graph must contain exactly p connected components.

A problem closely related to $\Delta\mathrm{CPR}_2$, called the *maximum clustering problem with given cluster sizes* (MCPGCS), has been extensively studied in the literature (see [6] and the references therein). Given a complete edge-weighted graph G and a sequence of integers $c_1, \ldots, c_p$, MCPGCS requires the computation of a maximum-weight cluster subgraph of G with exactly p connected components whose sizes are exactly $c_1, \ldots, c_p$, respectively. This problem has many applications ranging from final exam scheduling to VLSI design (see [12] and the references therein). $\Delta\mathrm{CPR}_2$ may be useful in some of these applications where we only want to put an upper bound on the sizes of the connected components in the output cluster subgraph.

If Π is a minimization problem requiring the modification of a given graph G by deleting and/or adding the fewest edges so that G satisfies a certain property P, then the *maximization version* of Π is the maximization problem requiring the computation of a graph $H = (V, E_H)$ from a given graph $G = (V, E)$ such that H satisfies property P and the quantity $\frac{|V|(|V|-1)}{2} - |E_H - E| - |E - E_H|$ is maximized. Bansal *et al.* [1] show that the maximization version of the correlation clustering problem admits a polynomial-time

approximation scheme. Shamir *et al.* [9] present a polynomial-time 0.878-approximation algorithm for the maximization version of the 2-cluster editing problem.

1.2 Our Contribution

In this paper, we first show that the maximization version of ΔCPR_k for any fixed $\Delta \geq 3$ and $k \geq 2$ admits a polynomial-time approximation scheme (PTAS). We obtain the PTAS by first designing a randomized PTAS for the problem and then derandomizing it using the method of conditional expectations.

We then present a polynomial-time 8-approximation algorithm for ΔCPR_2 for any fixed $\Delta \geq 3$. The algorithm is a nontrivial modification of the polynomial-time 4-approximation algorithm for the correlation clustering algorithm given in [2]. More specifically, we first obtain an LP formulation of ΔCPR_2 and then round its (fractional) solution.

ΔCPR_3 is much more difficult to approximate than ΔCPR_2, because the latter can be formulated as a minimization problem over a metric space while the former cannot. Despite this, we are able to present a quadratic-time 12-approximation algorithm for 3CPR_3. The algorithm and its analysis are quite involved.

1.3 Organization of the Paper

The next section contains basic definitions and notations. Section 3 presents a PTAS for the maximization version of ΔCPR_k. Section 4 gives a polynomial-time 8-approximation algorithm for ΔCPR_2. Section 5 describes a quadratic-time 12-approximation algorithm for 3CPR_3. The final section contains several open problems.

2 Preliminaries

Throughout this paper, a graph is always simple (i.e., has neither multiple edges nor self-loops) unless stated explicitly otherwise.

Throughout this section, G is a graph. We denote the vertex set and the edge set of G by $V(G)$ and $E(G)$, respectively. A subgraph of G is *proper* if it is not identical to G. The *neighborhood* of a vertex v in G, denoted $N_G(v)$, is the set of vertices in G adjacent to v; $deg_G(v) = |N_G(v)|$ is the *degree* of v in G. The *maximum degree* of G is the maximum degree of a vertex in G. For $U \subseteq V(G)$, the *subgraph of G induced by U* is the graph (U, F) with $F = \{\{u, v\} \in E(G) : u, v \in U\}$.

If P is a path or cycle in G, then the *length* of P is the number of edges in P. A path is *trivial* if its length is 0. An *endpoint* of a path P is a vertex v of P with $deg_P(v) \leq 1$. Note that a trivial path has a unique endpoint. A *triangle* in G is a cycle of length 3. The *distance* between two vertices u and v in G, denoted by $d_G(u, v)$, is the length of the shortest path between u and v in G. If G contains no path between u and v, then we define $d_G(u, v) = \infty$.

A *matching* of G is a set of pairwise nonadjacent edges of G. A *maximum matching* of G is a matching whose size is maximized over all matchings of G. A *clique* of G is a subgraph of G in which each pair of vertices are adjacent. The *size* of a clique C is the number of vertices in C. For each positive integer r, we use K_r to denote a clique of size r.

G is a *cane* if it can be obtained from a triangle T and a path P with $V(T) \cap V(P) = \emptyset$ by adding a new edge to connect one vertex of T to one endpoint of P. G is a *double-ended cane* if it can be obtained from two vertex-disjoint triangles T_1 and T_2 by adding a new path to connect one vertex of T_1 to one vertex of T_2.

G is a *forest* if it has no cycles. G is a *tree* if it is a connected forest. If G is a forest, then we call each $u \in V(G)$ with $deg_G(u) = 1$ a *leaf* of G, and call each $x \in V(G)$ with $deg_G(x) \geq 2$ an *internal node* of G.

Let H be a graph with $V(H) = V(G)$. For two vertices u and v of G, we call $\{u, v\}$ a *disagreement* between G and H if $\{u, v\} \in E(G) - E(H)$ or $\{u, v\} \in E(H) - E(G)$; and call $\{u, v\}$ an *agreement* between G and H if either $\{u, v\} \in E(G) \cap E(H)$, or $\{u, v\} \notin E(G)$ and $\{u, v\} \notin E(H)$. We use $D(G, H)$ (respectively, $A(G, H)$) to denote the number of disagreements (respectively, agreements) between G and H.

Let $\Delta \geq 3$ be an integer. An *approximate Δ-semi-phylogeny* of G is a forest T such that $V(G) \subseteq V(T)$, $deg_T(u) \leq 1$ for every $u \in V(G)$, and the maximum degree of T is at most Δ. If an approximate Δ-semi-phylogeny T of G is a tree and has no internal node of degree 2, then we call T an *approximate Δ-phylogeny* of G. Two vertices of G are *siblings* in an approximate Δ-semi-phylogeny T of G if they are adjacent to the same internal node in T.

Let $k \geq 2$ be an integer, let $\Delta \geq 3$ be an integer, and let T be an approximate Δ-semi-phylogeny of G. We use T^k to denote the graph whose vertices are the vertices of G and whose edges are those $\{u, v\}$ with $d_T(u, v) \leq k$. If T is an approximate Δ-phylogeny of G with $D(G, T^k) = 0$, then we call T a *kth root Δ-phylogeny* of G.

For $\Delta \geq 3$, a *Δ-phylogeny* is a tree in which the degree of each internal node is at least 3 and at most Δ. As before, for an integer $k \geq 2$ and a Δ-phylogeny T, we use T^k to denote the graph whose vertices are the leaves of T and whose edges are those $\{u, v\}$ with $d_T(u, v) \leq k$. For two integers $k \geq 2$ and $\Delta \geq 3$, a *k-densest Δ-phylogeny* is a Δ-phylogeny T such that $|E(T^k)|$ is maximized over all Δ-phylogenies with the same number of leaves as T.

3 PTAS for the Maximization Version of $\Delta\mathrm{CPR}_k$

This section presents a PTAS for the maximization version of $\Delta\mathrm{CPR}_k$ for any fixed $\Delta \geq 3$ and $k \geq 2$. Recall that a PTAS for the maximization version of $\Delta\mathrm{CPR}_k$ is an algorithm $\mathcal{A}$ such that for every given graph G and error parameter $\epsilon > 0$, $\mathcal{A}$ outputs an approximate Δ-phylogeny T of G in time polynomial in $|V(G)|$ such that $A(G, T^k_{\mathrm{opt}}) \leq (1+\epsilon)A(G, T^k)$, where T_{opt} is an approximate Δ-phylogeny of G such that $A(G, T^k_{\mathrm{opt}})$ is maximized over

all approximate Δ-phylogenies of G.

We start by presenting a PTAS for the simplest problem (namely, the maximization version of 3CPR$_3$) because it is simple and very efficient.

Lemma 3.1 *Let $n \geq 6$ be an integer, and let T be a 3-densest 3-phylogeny with n leaves. Then, $|E(T^3)| = n + 1$.*

PROOF. First, suppose that T^3 is connected. Then, the subtree of T induced by the set of its internal nodes must be a path P. Moreover, each endpoint of P is adjacent to exactly two leaves in T while each internal node of P is adjacent to exactly one leaf in T. Thus, T^3 is a double-ended cane because $n \geq 6$. Consequently, $|E(T^3)| = n + 1$.

Next, suppose that T^3 is disconnected. Consider a connected component C of T^3. Let T_C be the subtree of T whose leaves are exactly the vertices in $V(C)$. Note that $E(C) = E(T_C^3)$. We claim that $|E(C)| \leq |V(C)|$. The claim is clearly true when $|V(C)| \leq 3$. So, assume $|V(C)| \geq 4$. Then, by the connectivity of C, the subtree of T_C induced by the set of its internal nodes is a path Q and each internal node of Q is adjacent to exactly one leaf in T_C. Each endpoint of Q is adjacent to one or two leaves in T_C. Moreover, since T_C is a proper subtree of T, at least one endpoint of Q is adjacent to exactly one leaf in T_C. Thus, T_C^3 is a proper subgraph of a double-ended cane. Hence, $|E(T_C^3)| \leq |V(C)|$. This completes the proof of the claim. By the claim, it is clear that $|E(T^3)| \leq n$. □

The following corollary is immediate from the proof of Lemma 3.1 and will be very useful in Section 5.

Corollary 3.2 *A connected graph with at least six vertices has a 3rd root 3-phylogeny if and only if it is a double-ended cane. Moreover, a disconnected graph G has a 3rd root 3-phylogeny only if every connected component of G is a proper subgraph of a double-ended cane.*

Theorem 3.3 *The maximization version of* 3CPR$_3$ *admits a PTAS which runs in almost linear time.*

PROOF. Given a graph $G = (V, E)$ and an error parameter $\epsilon > 0$, the PTAS works as follows.

1. Let $n = |V|$. If n is small enough (say, $n < 15.5 + \frac{16}{\epsilon}$), then compute an approximate 3-phylogeny T of G by brute force such that $A(G, T^3)$ is maximized over all approximate 3-phylogenies of G, output it, and halt.

2. Let $v_1, \ldots, v_\ell$ be the vertices of G whose degrees in G are smaller than $\lceil \frac{n}{2} \rceil$.

3. If $\ell > 0$, then for each $v_i \in \{v_1, \ldots, v_\ell\}$, add enough edges from v_i to other vertices so that the degree of v_i in G becomes $\lceil \frac{n}{2} \rceil$.

4. Find a Hamiltonian path $P = u_1, \ldots, u_n$ in G. (*Comment*: Dirac's classic theorem asserts that if a graph has no vertex adjacent to less than half its vertices, then the graph is Hamiltonian.)

5. Output an approximate 3-phylogeny T of G such that T^3 contains P as a subgraph and $|E(T^3)| = n+1$. (*Comment*: By the proof of Lemma 3.1, it is easy to construct T.)

Let $\alpha = \frac{0.5n(n-1)-m}{n+1}$, where m is the number of edges in G. Let $T_{\rm opt}$ be an approximate 3-phylogeny of G such that $A(G, T^3_{\rm opt})$ is maximized over all approximate 3-phylogenies of G. In the best case, all edges in $T^3_{\rm opt}$ are contained in G. So, by Lemma 3.1, $A(G, T^3_{\rm opt}) \leq \alpha(n+1) + (n+1)$. For the output T of the above algorithm, we claim that $A(G, T^3_{\rm opt}) \leq (1+\epsilon)A(G, T^3)$. The claim is clearly true if $n < 15.5 + \frac{16}{\epsilon}$ (cf. Step 1). So, we hereafter assume that $n \geq 15.5 + \frac{16}{\epsilon}$.

We claim that $\ell \leq 2n - \frac{4m}{n-1}$. To see this, first note that $2m = \sum_{v \in V} deg_G(v) \leq \ell(\lceil \frac{n}{2} \rceil - 1) + (n-\ell)(n-1)$ because G has exactly ℓ vertices of degree smaller than $\frac{n}{2}$. So, $2m \leq \ell(\frac{n+1}{2} - 1) + (n-\ell)(n-1)$, or equivalently $\ell \leq 2n - \frac{4m}{n-1}$. This establishes the claim.

The above claim together with the definition of α implies that $\ell \leq \frac{4\alpha(n+1)}{n-1}$. Thus, at most $\frac{8\alpha(n+1)}{n-1}$ edges of P are not edges of G. Hence, at most $2 + \frac{8\alpha(n+1)}{n-1}$ edges of T^3 are not edges of G. Therefore, $A(G, T^3) \geq (\alpha(n+1) - 2 - \frac{8\alpha(n+1)}{n-1}) + ((n+1) - 2 - \frac{8\alpha(n+1)}{n-1}) = \alpha(n+1) - 4 - \frac{16\alpha(n+1)}{n-1}$. Now, since $A(G, T^3_{\rm opt}) \leq \alpha(n+1) + (n+1)$, $A(G, T^3_{\rm opt}) \leq (1+\epsilon)A(G, T^3)$ if and only if $\frac{(\alpha+1)\epsilon}{1+\epsilon} - \frac{16\alpha}{n-1} - \frac{4}{n+1} \geq 0$. Thus, it remains to show the last inequality. To this end, we denote the left side of the inequality by $f(\alpha)$, i.e., we view the left side as a linear function of α. The minimum value of α is 0 at which we have $f(\alpha) \geq 0$ because $n \geq 15.5 + \frac{16}{\epsilon}$. Moreover, the maximum value of α is $\frac{n(n-1)}{2(n+1)}$ at which we also have $f(\alpha) \geq 0$ because $n \geq 15.5 + \frac{16}{\epsilon}$. Hence, by the linearity of $f(\alpha)$, we always have $f(\alpha) \geq 0$.

Finally, we note that the above algorithm runs in almost linear time because Step 4 is the most time-consuming and it can be done in almost linear time [5]. □

We next present a PTAS for the maximization version of ΔCPR$_k$ for any fixed $\Delta \geq 3$ and $k \geq 2$. The PTAS needs a polynomial-time subroutine to construct a k-densest Δ-phylogeny. Such a subroutine exists as shown in the following lemma:

Lemma 3.4 *Let $\Delta \geq 3$ and $k \geq 2$ be constant integers. Then, given a positive integer $n \geq 3$ in unary, we can construct a k-densest Δ-phylogeny with n leaves in $O(n^{\Delta+1})$ time.*

Proof. By a dynamic programming method. We define a *Δ-quasi-phylogeny* to be a tree T with maximum degree $\leq \Delta$ such that T has a distinguished internal node α, all internal nodes of T except α are of degree at least 3 in T, and the degree of α in T is at most $\Delta - 1$ (and at least 2). As before, we use T^k to denote the graph whose vertices are the leaves of T and whose edges are the unordered pairs $\{u, v\}$ with $d_T(u, v) \leq k$.

We say that a k-tuple $(p, \ell_1, \ldots, \ell_{k-1})$ is *proper* if $p \leq n-1$, $\sum_{i=1}^{k-1} \ell_i \leq p$, and $0 \leq \ell_i \leq (\Delta-1)^i$ for each $i \in \{1, \ldots, k-1\}$. For each proper k-tuple $(p, \ell_1, \ldots, \ell_{k-1})$, let $M_p(\ell_1, \ldots, \ell_{k-1})$ denote the maximum size of $E(T^k)$ where T ranges over all Δ-quasi-phylogenies satisfying the following conditions:

1. T has exactly p leaves.
2. For each $i \in \{1, 2, \ldots, k-1\}$, ℓ_i is the number of leaves of T whose distance from the distinguished internal node of T is exactly i.

If there is no Δ-quasi-phylogeny T satisfying the above conditions, $M_p(\ell_1, \ldots, \ell_{k-1}) = -\infty$. Otherwise, we define a *witness Δ-quasi-phylogeny* for the k-tuple $(p, \ell_1, \ldots, \ell_{k-1})$ to be a Δ-quasi-phylogeny T that satisfies the above conditions and $|E(T^k)|$ is maximized.

Obviously, $M_2(\ell_1, \ldots, \ell_{k-1}) = 1$ only when $\ell_1 = 2$ and $\ell_2 = \cdots = \ell_{k-1} = 0$; for other values of $\ell_1, \ldots, \ell_{k-1}$, $M_2(\ell_1, \ldots, \ell_{k-1}) = -\infty$. Moreover, it is easy to construct a witness Δ-quasi-phylogeny for $(2, 2, 0, \ldots, 0)$.

For each proper k-tuple $t = (p, \ell_1, \ldots, \ell_{k-1})$, we define a *proper decomposition* of t to be a set of proper k-tuples $(p_1, \ell_{1,1}, \ldots, \ell_{1,k-1})$, $\ldots$, $(p_h, \ell_{h,1}, \ldots, \ell_{h,k-1})$ with $0 \leq h \leq \Delta - 1 - \ell_1$ that satisfy the following conditions:

1. $h + \ell_1 \geq 2$, $\sum_{i=1}^{h} p_i = p - \ell_1$, and $p_i \geq 2$ for each $1 \leq i \leq h$.
2. For each $1 \leq j \leq k-2$, $\sum_{i=1}^{h} \ell_{i,j} = \ell_{j+1}$.

The *value* of this proper decomposition is the sum of $\sum_{i=1}^{h} M_{p_i}(\ell_{i,1}, \ldots, \ell_{i,k-1})$, $\frac{\ell_1(\ell_1-1)}{2}$, $\ell_1 \sum_{j=2}^{k-1} \ell_j$, and $\sum_{i_1=1}^{h-1} \sum_{i_2=i_1+1}^{h} \sum_{j_1=1}^{k-3} (\ell_{i_1,j_1} \sum_{j_2=1}^{k-2-j_1} \ell_{i_2,j_2})$. This value is the size of $E(T^k)$, where T is the Δ-quasi-phylogeny obtained from h given witness Δ-quasi-phylogenies $T_1, \ldots, T_h$ for $(p_1, \ell_{1,1}, \ldots, \ell_{1,k-1})$, $\ldots$, $(p_h, \ell_{h,1}, \ldots, \ell_{h,k-1})$ as follows:

1. Introduce a new (internal) node α and connect it to ℓ_1 other new (leaf) nodes.
2. For each $1 \leq i \leq h$, connect α to the distinguished internal node of T_i.
3. Specify α as the distinguished internal node of T.

It should be clear that $M_p(\ell_1, \ldots, \ell_{k-1})$ is the maximum value of a proper decomposition of $(p, \ell_1, \ldots, \ell_{k-1})$.

We compute $M_p(\ell_1, \ldots, \ell_{k-1})$ and a witness Δ-quasi-phylogeny $T_p(\ell_1, \ldots, \ell_{k-1})$ for all proper k-tuples $(p, \ell_1, \ldots, \ell_{k-1})$, in the increasing order of p. Note that for each proper k-tuple $(n-1, \ell_1, \ldots, \ell_{k-1})$, we can use $T_{n-1}(\ell_1, \ldots, \ell_{k-1})$ to construct a Δ-phylogeny T with n leaves by connecting a new leaf to the distinguished internal node of $T_{n-1}(\ell_1, \ldots, \ell_{k-1})$; the size of $E(T^k)$ is $M_{n-1}(\ell_1, \ldots, \ell_{k-1}) + \sum_{j=1}^{k-1} \ell_j$. So, we find a $(k-1)$-tuple $(\ell_1, \ldots, \ell_{k-1})$ that maximizes $M_{n-1}(\ell_1, \ldots, \ell_{k-1}) + \sum_{j=1}^{k-1} \ell_j$. Then, by connecting a new leaf to the distinguished internal node of $T_{n-1}(\ell_1, \ldots, \ell_{k-1})$, we obtain a k-densest Δ-phylogeny with n leaves.

Finally, we point out that the above dynamic programming can be done in $O(n^{\Delta+1})$ time. □

Theorem 3.5 *For every constant $\Delta \geq 3$ and $k \geq 2$, the maximization version of ΔCPR$_k$ admits a randomized PTAS. Moreover, the randomized PTAS runs in $T_{k,\Delta}(n)+O(n)$ time, where $T_{k,\Delta}(n)$ is the time needed to construct a k-densest Δ-phylogeny with n leaves.*

PROOF. Given a graph $G = (V, E)$ and an error parameter ϵ, the randomized PTAS works as follows.

1. Use Lemma 3.4 to construct a k-densest Δ-phylogeny T with leaf set $V(G)$.
2. Let $n = |V|$, $m = |E|$, and $m' = |E(T^k)|$.
3. If n is so small that $\epsilon n(n-1) < (4+2\epsilon)m'$, then compute an approximate Δ-phylogeny $T_{\rm opt}$ of G by brute force such that $A(G, T^k_{\rm opt})$ is maximized over all approximate Δ-phylogenies of G, output $T^k_{\rm opt}$, and halt. (*Comment*: It is easy to see that $m' \leq (\Delta-1)^{k-1}n/2$. Indeed, we can claim that m' is at most $\Delta(\Delta-1)^{(k-2)/2}n$ (respectively, $(\Delta-1)^{(k+1)/2}n$) if k is even (respectively, odd). To see this claim, first observe that T^k is a chordal graph in which the maximum size of a clique is at most $\Delta(\Delta-1)^{(k-2)/2}$ (respectively, $(\Delta-1)^{(k+1)/2}$) if k is even (respectively, odd) [11]. The claim follows from this observation and the well-known fact that a chordal graph H without cliques of size $s+1$ can have at most $s|V(H)| - \frac{s(s+1)}{2}$ edges. By the claim, $m' = O(n)$ and hence n must be a small constant in order to satisfy $\epsilon n(n-1) < (4+2\epsilon)m'$.)
4. Generate a permutation σ of V uniformly at random.
5. Use σ to permute the labels of the leaves of T.
6. Output T.

Let $T_{\rm opt}$ be an approximate Δ-phylogeny of G such that $A(G, T^k_{\rm opt})$ is maximized over all approximate Δ-phylogenies of G. In the best case, all edges in $T^k_{\rm opt}$ are contained in G. So, $A(G, T^k_{\rm opt}) \leq \frac{n(n-1)}{2} - m + m'$.

For the output T of the above algorithm, we claim that the expected value of $A(G, T^k)$ is not smaller than $A(G, T^k_{\rm opt})/(1+\epsilon)$. The claim is clearly true if n is small enough (cf. Step 3). So, we assume that n is not small. Consider two arbitrary unordered pairs $\{u, v\}$ and $\{u', v'\}$. Since σ is a random permutation of V, the probability that $\{u, v\}$ is an edge of T^k is equal to the probability that $\{u', v'\}$ is an edge of T^k no matter whether $\{u, v\} \cap \{u', v'\} = \emptyset$ or not. Hence, the probability that $\{u, v\}$ is an edge of T^k is $\frac{2m'}{n(n-1)}$. Thus, the expected number of edges that are in both G and T^k is $\frac{2mm'}{n(n-1)}$. Moreover,

the expected number of edges that are in neither G nor T^k is $(\frac{n(n-1)}{2} - m)(1 - \frac{2m'}{n(n-1)})$. Obviously, the sum of these two expected numbers is equal to the expected value of $A(G, T^k)$. Now, we can use elementary calculus to show that the ratio of $A(G, T^k_{\rm opt})$ to the expected value of $A(G, T^k)$ is not larger than $1 + \epsilon$.

The time complexity of the randomized PTAS is clearly as stated in the theorem. □

Corollary 3.6 *For every constant $\Delta > 3$ and $k \geq 2$, the maximization version of ΔCPR$_k$ admits a PTAS. Moreover, the PTAS runs in $T_{k,\Delta}(n) + O(n^2(n+m))$ time.*

Proof. It suffices to derandomize the randomized PTAS in Theorem 3.5. This is done by the method of conditional expectations. Let T be as in Step 1 of the randomized PTAS. By the proof of Theorem 3.5, after the labels of the leaves of T are randomly permuted, the expected value $\mathcal{E}$ of $A(G, T^k)$ will be at least $A(G, T^k_{\rm opt})/(1 + \epsilon)$.

It suffices to (deterministically) fix the labels of the leaves of T so that $A(G, T^k) \geq \mathcal{E}$. To this end, we remove the labels of the leaves of T, and then (deterministically) assign the vertices of G (as labels) to the leaves of T one by one as follows. Let $v_1, \ldots, v_n$ be an arbitrary ordering of the vertices of G. Suppose that we have already assigned $v_1, \ldots, v_{i-1}$ $(1 \leq i \leq n)$ to some nodes $\alpha_1, \ldots, \alpha_{i-1}$ of T respectively so that if $v_i, \ldots, v_n$ are randomly one-to-one assigned to the remaining unlabeled leaves of T, then the expected value of $A(G, T^k)$ will be at least $\mathcal{E}$. This is trivially true when $i = 1$. We want to assign v_i so that if $v_{i+1}, \ldots, v_n$ are randomly one-to-one assigned to the remaining unlabeled leaves of T, then the expected value of $A(G, T^k)$ will be at least $\mathcal{E}$. To this end, for each unlabeled leaf α_j of T, we compute the conditional expectation $\mathcal{E}_{\alpha_j}$ of $A(G, T^k)$ given that $v_1, \ldots, v_i$ are assigned to $\alpha_1, \ldots, \alpha_i$, respectively. Before describing how to compute $\mathcal{E}_{\alpha_j}$, we first note that $\max_{\alpha_j} \mathcal{E}_{\alpha_j} \geq \mathcal{E}$ (where the maximization is taken over all unlabeled leaves of T) and that we can assign v_i to the α_j such that $\mathcal{E}_{\alpha_j}$ is maximized.

The computation of $\mathcal{E}_{\alpha_j}$ is based on linearity of expectation. For each unordered pair $\{u, v\}$ of vertices of G, let $p_{u,v}$ be the conditional probability that $\{u, v\}$ is an agreement between G and T^k, given that $v_1, \ldots, v_i$ are assigned to $\alpha_1, \ldots, \alpha_i$, respectively. Then, $\mathcal{E}_{\alpha_j} = \sum_{\{u,v\}} p_{u,v}$, where the summation is taken over all unordered pairs. So, we need to consider how to compute $p_{u,v}$. There are three cases:

Case 1: $\{u, v\} \subseteq \{v_1, \ldots, v_i\}$. In this case, either $p_{u,v} = 0$ or $p_{u,v} = 1$. If either $\{u, v\} \in E(G)$ and the two leaves of T labeled u and v are at most distance k apart, or $\{u, v\} \notin E(G)$ and the two leaves of T labeled u and v are at least distance $k + 1$ apart, then $p_{u,v} = 1$; otherwise, $p_{u,v} = 0$. Obviously, we can compute the sum of all $p_{u,v}$ such that $\{u, v\} \subseteq \{v_1, \ldots, v_i\}$, in linear total time.

Case 2: $|\{u, v\} \cap \{v_1, \ldots, v_i\}| = 1$. We assume that $u \in \{v_1, \ldots, v_i\}$ but $v \notin \{v_1, \ldots, v_i\}$; the other case is similar. Let a_u be the number of unlabeled leaves of T that are within a distance of at most k from u. Obviously, if $\{u, v\}$ is an edge of G, then $p_{u,v} = a_u/(n - i)$; otherwise, $p_{u,v} = 1 - a_u/(n - i)$. So, $p_{u,v}$ can be computed in $O(1)$ time. Since $p_{u,v}$ is

independent of v, we can compute the sum of all $p_{u,v}$ such that $|\{u,v\} \cap \{v_1, \ldots, v_i\}| = 1$, in linear total time.

Case 3: $\{u,v\} \cap \{v_1, \ldots, v_i\} = \emptyset$. Let b be the number of unordered pairs $\{\beta, \gamma\}$ of unlabeled leaves of T that are at most distance k apart in T. Obviously, if $\{u,v\}$ is an edge of G, then $p_{u,v} = \frac{2b}{(n-i)(n-i-1)}$; otherwise, $p_{u,v} = 1 - \frac{2b}{(n-i)(n-i-1)}$. Since $p_{u,v}$ is independent of u and v, we can compute the sum of all $p_{u,v}$ such that $\{u,v\} \cap \{v_1, \ldots, v_i\} = \emptyset$, in linear total time.

In summary, we can compute $\mathcal{E}_{\alpha_j}$ in linear time. Thus, we can assign v_i to the best α_j in $O((n-i)(n+m))$ time. So, the total running time is $T_{k,\Delta}(n) + O(n^2(n+m))$. □

Although the above PTAS runs in polynomial time, it is not very efficient because $T_{k,\Delta}(n)$ can be very large. We can make the PTAS very efficient if we can find out the structure of a k-densest Δ-phylogeny with n leaves. Obviously, we can do this when $k = 2$. We can also do this when $k = 3$, by the proofs of Lemma 3.1 and the following lemma:

Lemma 3.7 *Let $\Delta > 3$ and $n > 2\Delta - 2$ be integers. Let T be a 3-densest Δ-phylogeny with n leaves. Then, there are constants a, b, c such that $|E(T^3)| = (1.5\Delta - 3.5)n + a\Delta^2 + b\Delta + c$.*

PROOF. We may assume that there is no node of degree 2 in T, because we can delete such a node and connect its original neighbors by a new edge without decreasing $|E(T^3)|$.

Let v be an internal node of T. Node v is *unsaturated* if its degree in T is smaller than Δ, and is *saturated* otherwise. Node v is *extreme* if all but one of its neighbors in T are leaves of T. Node v is *branching* if at least three of its neighbors in T are internal nodes of T.

We will define two types of operations on T. Given two distinct unsaturated internal nodes x and y of T such that both x and y are adjacent to at least one leaf in T, the *first-type operation* on T modifies T as follows:

1. Let n_x (respectively, n_y) be the number of leaves adjacent to x (respectively, y) in T.

2. If the degree of each leaf adjacent to y in T^3 is not smaller than the degree of each leaf adjacent to x in T^3, then select $\min\{n_x, \Delta - deg_T(y)\}$ leaves adjacent to x in T, delete the edges from them to x, and add edges from them to y; otherwise, select $\min\{n_y, \Delta - deg_T(x)\}$ leaves adjacent to y in T, delete the edges from them to y, and add edges from them to x.

3. If x (respectively, y) becomes of degree 1, then delete x (respectively, y).

4. If x (respectively, y) becomes of degree 2, then delete x (respectively, y) and connect its two original neighbors by a new edge.

A simple inspection shows that the first-type operation on T does not decrease $|E(T^3)|$. Note that each nonbranching node of T is adjacent to at least one leaf in T because the degree of each internal node in T is at least 3. So, if the first-type operation is not applicable to T, then T has at most one unsaturated nonbranching node.

The *second-type operation* on T can be used only when T has at least one branching node and the first-type operation is not applicable to T. The operation works on T as follows:

1. If T has an unsaturated nonbranching internal node, then root T at such a node; otherwise, root T at an arbitrary extreme internal node.
2. Find a branching node x of T such that no descendant of x in T is a branching node of T.
3. Let x_1 and x_2 be two internal children of x in T. (*Comment*: Both x_1 and x_2 are saturated, because otherwise we would be able to apply the first-type operation on T.)
4. If x_1 is extreme, then let $y_1 = x_1$; otherwise, let y_1 be the extreme descendant of x_1 in T.
5. Let u_1 be a child of y_1 in T.
6. Delete edges $\{y_1, u_1\}$ and $\{x, x_2\}$, and add edges $\{x, u_1\}$ and $\{y_1, x_2\}$.
7. Unroot T.

A simple calculation shows that the second-type operation on T does not decrease $|E(T^3)|$.

If neither the first-type nor the second-type operation is applicable to T, then the subtree of T induced by the set of its internal nodes is a path and there is at most one unsaturated internal node x in T. If x exists and is not extreme in T, then we further modify T as follows:

1. Let $y \neq x$ be an extreme internal node of T. (*Comment*: y is saturated.)
2. Let n_x be the number of leaves adjacent to x in T. (*Comment*: $n_x \geq 1$.)
3. Select $\Delta - 2 - n_x$ leaves adjacent to y in T, delete the edges from them to y, and add edges from them to x.

A simple calculation shows that the above modification of T does not decrease $|E(T^3)|$. Now, T has the following properties:

- The subtree of T induced by the set of its internal nodes is a path P. (*Comment*: We call P the *backbone* of T.)

- Each internal node of P is adjacent to exactly $\Delta - 2$ leaves in T.
- At least one endpoint of P is adjacent to exactly $\Delta - 1$ leaves in T. (*Comment*: We call the other endpoint of P the *tail* of P.)

When T has the above properties, we say that T is in the *canonical form*.

If T is in the canonical form, a simple calculation shows that $|E(T^3)| = (1.5\Delta - 3.5)n - \Delta^2 + 4.5\Delta - 4.5 + 0.5\ell^2 - (0.5\Delta - 1)\ell$, where ℓ is the number of leaves of T adjacent to the tail of the backbone of T. Note that $\ell = \Delta - 2$ if $n - (\Delta - 1) \equiv 0 \pmod{\Delta - 2}$, $\ell = \Delta - 1$ if $n - (\Delta - 1) \equiv 1 \pmod{\Delta - 2}$, and ℓ equals $(n - \Delta + 1) \bmod (\Delta - 2)$ otherwise. □

Corollary 3.8 *For every constant $\Delta \geq 3$, both the maximization version of $\Delta\mathrm{CPR}_2$ and that of $\Delta\mathrm{CPR}_3$ admit a linear-time randomized PTAS and also admit an $O(n^2(n+m))$-time PTAS.*

4 Approximation Algorithm for $\Delta\mathrm{CPR}_2$

In this section, we present a polynomial-time 8-approximation algorithm for $\Delta\mathrm{CPR}_2$ for any constant $\Delta \geq 4$. Throughout the rest of this section, fix a graph $G = (V, E)$. We assume that $|V| \geq \Delta + 1$ and $|E| \geq 1$; otherwise, we can trivially solve the problem for G in linear time.

Our algorithm is a nontrivial modification of a 4-approximation algorithm for the correlation clustering problem due to Charikar *et al.* [2]. Recall that in the correlation clustering problem, we are required to modify G into a *cluster graph* by deleting and/or adding the fewest edges, where a cluster graph is a graph in which each connected component is a clique. For convenience, we call each connected component of a cluster graph a *cluster*.

The 4-approximation algorithm in [2] is based on the following IP formulation of the correlation clustering problem (for G):

$$\begin{aligned} \text{minimize} \quad & \sum_{\{u,v\} \in E} x_{u,v} + \sum_{\{u,v\} \notin E} (1 - x_{u,v}) \\ \text{such that} \quad & x_{u,w} \leq x_{u,v} + x_{v,w} \qquad \text{for all distinct vertices } u, v, w \\ & x_{u,v} \in \{0, 1\} \qquad \text{for all distinct vertices } u, v. \end{aligned} \tag{4.1}$$

In IP (4.1), two vertices u and v are in the same cluster if and only if $x_{u,v} = 0$. The LP relaxation is obtained by replacing the integer requirements in IP (4.1) with $0 \leq x_{u,v} \leq 1$ for all vertices u, v.

For an integer $\Delta \geq 3$, a Δ-*cluster graph* is a graph in which each connected component is a clique of size at most $\Delta - 1$. Roughly speaking, in our problem $\Delta\mathrm{CPR}_2$, we want a

Δ-cluster graph. So, we modify IP (4.1) by adding more constraints to ensure that the size of each cluster is at most $\Delta - 1$ as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{\{u,v\}\in E} x_{u,v} + \sum_{\{u,v\}\notin E} (1 - x_{u,v}) && (4.2)\\
\text{such that} \quad & x_{u,w} \le x_{u,v} + x_{v,w} \qquad \text{for all distinct vertices } u, v, w\\
& \sum_{v\in V, v\neq u} x_{u,v} \ge |V| - (\Delta - 1) \qquad \text{for all vertices } u\\
& \sum_{\{u,v\}\subseteq S, u\neq v} x_{u,v} \ge \frac{|S|(|S|-1)}{2} - \frac{(\Delta-1)(\Delta-2)}{2} - \frac{(|S|-\Delta+1)(|S|-\Delta)}{2}\\
& \qquad\qquad \text{for all } S \subseteq V \text{ with } |S| \le 2(\Delta-1)\\
& x_{u,v} \in \{0,1\} \qquad \text{for all distinct vertices } u, v.
\end{aligned}$$

In IP (4.2), the second group of constraints ensures that each cluster in the output cluster graph contains at most $\Delta-1$ vertices. Note that a Δ-cluster graph with $\ell \le 2(\Delta-1)$ vertices can have at most $\frac{(\Delta-1)(\Delta-2)}{2} + \frac{(\ell-\Delta+1)(\ell-\Delta)}{2}$ edges. Thus, for every Δ-cluster graph H and for every set S of at most $2(\Delta-1)$ vertices of H, the subgraph of H induced by S can have at most $\frac{(\Delta-1)(\Delta-2)}{2} + \frac{(|S|-\Delta+1)(|S|-\Delta)}{2}$ edges. This property leads to the third group of constraints in IP (4.2). Now, since T^2 is a Δ-cluster graph for each approximate Δ-phylogeny T of G, we see that IP (4.2) is a relaxation of ΔCPR_2.

We obtain the LP relaxation of IP (4.2) by replacing the integer requirements in IP (4.2) with $0 \le x_{u,v} \le 1$ for all vertices u, v.

In order to describe our approximation algorithm for ΔCPR_2, we need to review the 4-approximation algorithm in [2] as follows:

1. Solve the LP relaxation of IP (4.1) to obtain an optimal vector $(\ldots, x_{u,v}, \ldots)$.
2. Initialize $U = V$.
3. While U is nonempty, repeat the following steps (in turn):
   (a) Select a vertex u arbitrarily from U.
   (b) Let W be the set of all $v \in U - \{u\}$ such that $x_{u,v} \le \epsilon$, where $0 < \epsilon < \frac{2}{3}$ is a fixed constant to be determined later.
   (c) If $\sum_{v\in W} x_{u,v}/|W| \ge \epsilon/2$, then construct a new singleton cluster $C = \{u\}$ and jump to Step 3e.
   (d) If $\sum_{v\in W} x_{u,v}/|W| < \epsilon/2$, then construct a new cluster $C = \{u\} \cup W$.
   (e) Remove all vertices of C from U.

Lemma 4.1 *Let H be the cluster graph output by the above algorithm. Then, $D(G,H) \leq \max\{\frac{2}{\epsilon}, \frac{1}{1-1.5\epsilon}\} \cdot \mathrm{OPT}_{LP}$, where OPT_{LP} is the optimal value of a solution to the LP relaxation of IP (4.1).*

PROOF. Chariker *et al.* [2] fix the constant ϵ to be 0.5 and show that $D(G,H) \leq 4 \cdot \mathrm{OPT}_{LP}$. A careful inspection of their proof reveals that $D(G,H) \leq \max\{\frac{2}{\epsilon}, \frac{1}{1-1.5\epsilon}\} \cdot \mathrm{OPT}_{LP}$ for any fixed constant $0 < \epsilon < \frac{2}{3}$. □

We modify the above approximation algorithm (so that it outputs a Δ-cluster graph) by first replacing IP (4.1) in Step 1 with IP (4.2) and then modifying Step 3d as follows:

(3d') If $\sum_{v \in W} x_{u,v}/|W| < \epsilon/2$ and $|W| \leq \Delta - 2$, then construct a new cluster $C = \{u\} \cup W$; otherwise, construct two new clusters $D_1 = \{u\} \cup W'$ and $D_2 = W - W'$, where $W' \subseteq W$, $|W'| = \Delta - 2$, and $x_{u,w_1} \leq x_{u,w_2}$ for all $w_1 \in W'$ and $w_2 \in W - W'$.

The following lemma shows the correctness of our algorithm when $\epsilon \leq \frac{1}{2}$:

Lemma 4.2 *At each repetition of the while-loop of our algorithm, $|W| < \frac{\Delta-1}{1-\epsilon}$.*

PROOF. For a contradiction, assume that $|W| \geq \frac{\Delta-1}{1-\epsilon}$. Then, $\sum_{v \in V, v \neq u} x_{u,v} = \sum_{v \in W} x_{u,v} + \sum_{v \in V-(\{u\} \cup W)} x_{u,v} \leq \epsilon|W| + (|V| - |W| - 1) \leq |V| - \Delta$, contradicting the second group of constraints in IP (4.2). □

Lemma 4.3 *Let H be the cluster graph output by our algorithm where we fix $\epsilon = \frac{1}{3}$. Then, $D(G,H) \leq 8 \cdot \mathrm{OPT}_{LP}$, where OPT_{LP} is the optimal value of a solution to the relaxation of IP (4.2).*

PROOF. Suppose that Steps 3a through 3e were repeated exactly h times in our algorithm. For each $1 \leq i \leq h$, if only one cluster is constructed at the ith repetition of Step 3d', then let C_i denote it; otherwise, let $D_{i,1}$ and $D_{i,2}$ denote the two clusters constructed then, and let $C_i = D_{i,1} \cup D_{i,2}$.

Consider the cluster graph H' with clusters $C_1, \ldots, C_h$. As in the proof of Lemma 4.1, we can observe that the proof in [2] indeed shows that $D(G,H') \leq \max\{\frac{2}{\epsilon}, \frac{1}{1-1.5\epsilon}\} \cdot \mathrm{OPT}_{LP}$, even though our LP has more constraints than theirs. So, it remains to show that splitting C_i into $D_{i,1}$ and $D_{i,2}$ does not introduce too many disagreements.

Suppose $\epsilon < \frac{1}{2}$ (this ensures the correctness of our algorithm). Fix a constant $\delta > 0$ such that $1 - 2\epsilon \geq \delta \cdot 2\epsilon$. Consider an $i \in \{1, \ldots, h\}$ such that $D_{i,1}$ and $D_{i,2}$ were constructed in our algorithm. Let $s_1 = |D_{i,1}|$ and $s_2 = |D_{i,2}|$. By Lemma 4.2 and our algorithm, $s_1 = \Delta - 1$ and $s_2 \leq \Delta - 1$. Since G can have at most $s_1 s_2$ edges between $D_{i,1}$ and $D_{i,2}$, splitting C_i into $D_{i,1}$ and $D_{i,2}$ introduces at most $s_1 s_2$ new disagreements. On the other hand, by the third group of constraints in the LP relaxation of IP (4.2), $\sum_{\{v,w\} \subseteq C_i, v \neq w} x_{v,w} \geq \frac{|C_i|(|C_i|-1)}{2} - \frac{(\Delta-1)(\Delta-2)}{2} - \frac{(|C_i|-\Delta+1)(|C_i|-\Delta)}{2} = s_1 s_2$. Thus, splitting

C_i into $D_{i,1}$ and $D_{i,2}$ introduces at most $\sum_{\{v,w\}\subseteq C_i, v\neq w} x_{v,w}$ new disagreements. Moreover, for each $\{v,w\} \subseteq C_i$, $x_{v,w} \leq x_{u,v} + x_{u,w} \leq 2\epsilon$ and so $1 - x_{v,w} \geq 1 - 2\epsilon \geq \delta \cdot 2\epsilon \geq \delta x_{v,w}$. Therefore, the number of new disagreements introduced by splitting C_i into $D_{i,1}$ and $D_{i,2}$ is at most $\sum_{\{v,w\}\subseteq C_i, v\neq w} x_{v,w} \leq \sum_{\{v,w\}\subseteq C_i, \{v,w\}\in E} x_{v,w} + \frac{1}{\delta}\sum_{\{v,w\}\subseteq C_i, \{v,w\}\notin E}(1 - x_{v,w}) \leq \frac{1}{\delta}(\sum_{\{v,w\}\subseteq C_i, \{v,w\}\in E} x_{v,w} + \sum_{\{v,w\}\subseteq C_i, v\neq w}(1 - x_{v,w}))$.

By the discussion in the last paragraph, the total number of new disagreements is at most $\frac{1}{\delta} \cdot \mathrm{OPT}_{LP}$. Thus, the total number of disagreements between G and the cluster graph output by our algorithm is at most $(\max\{\frac{2}{\epsilon}, \frac{1}{1-1.5\epsilon}\} + \frac{1}{\delta}) \cdot \mathrm{OPT}_{LP}$, which achieves the value of 8 when $\epsilon = \frac{1}{3}$ and $\delta = \frac{1}{2}$. □

Remark. By Lemma 4.2, we can modify the third group of constraints in IP (4.2) by changing the condition $|S| \leq 2(\Delta - 1)$ to $|S| \leq 1.5\Delta - 0.5$. This modification does not change the correctness and the approximation ratio of our algorithm, and makes our algorithm slightly more efficient.

Theorem 4.4 *There is a polynomial-time 8-approximation algorithm for* $\Delta\mathrm{CPR}_2$.

Proof. Let H be the Δ-cluster graph output by our above algorithm. Let T_{opt} be an approximate Δ-phylogeny of G such that $D(G, T^2_{\mathrm{opt}})$ is minimized over all approximate Δ-phylogenies of G. Clearly, T^2_{opt} is a Δ-cluster graph. So, $D(G, T^2_{\mathrm{opt}}) \geq \mathrm{OPT}_{LP}$, where OPT_{LP} is as in Lemma 4.3. Thus, it suffices to show how to construct an approximate Δ-phylogeny of G from H.

Let $C_1, \ldots, C_h$ be the clusters of H. Since $|V| \geq \Delta+1$, $h \geq 2$. We may assume that the subgraph of G induced by the set of vertices in singleton clusters of H contains no edges at all, because otherwise we can decrease $D(G, H)$ by adding one edge of G to H to connect two singleton clusters of H into one (nonsingleton) cluster. By this assumption, at least one cluster of H is nonsingleton because G has at least one edge. If at least two clusters of H, say C_1 and C_2, are nonsingleton, then we can construct a 2nd root Δ-phylogeny T of H as follows:

1. For each $i \in \{1, \ldots, h\}$, introduce an internal node x_i and connect each vertex of C_i to x_i by an edge.

2. Use $h - 1$ edges to connect $x_1, \ldots, x_h$ into a path with endpoints x_1 and x_2.

Hence, we may assume that exactly one cluster of H, say C_1, is nonsingleton and the others are singleton. For each $i \in \{2, \ldots, h\}$, let v_i be the unique vertex in C_i. Then, the subgraph of G induced by $\{v_2, \ldots, v_h\}$ has no edges at all. Now, $|C_1| \leq \Delta - 1$ and removing the vertices of C_1 from G yields a graph with no edges at all; this simple structure of G enables us to construct T_{opt} in polynomial time. We omit the tedious details here and only sketch the ideas in the next paragraph.

The first idea is to divide $\{v_2, ..., v_h\}$ into groups so that two vertices v_i and v_j with $2 \leq i \neq j \leq h$ are in the same group if and only if $N_G(v_i) = N_G(v_j)$. Note that there can be at most $2^{|C_1|}$ groups. Let $U_1, \ldots, U_q$ be the groups. The second idea is to try each partition $\pi = \{C_{1,1}, \ldots, C_{1,p}\}$ of C_1 (into nonempty subsets) and each pq-tuple $t = (s_{1,1}, \ldots, s_{1,q}, \ldots, s_{p,1}, \ldots, s_{p,q})$ of nonnegative integers, where $s_{j,1}+\cdots+s_{j,q}+|C_{1,j}| \leq \Delta-1$ for each $j \in \{1, \ldots, p\}$, and $s_{1,i} + \cdots + s_{p,i} \leq |U_i|$ for each $i \in \{1, \ldots, q\}$. Note that π and t together correspond to a Δ-cluster graph $G_{\pi,t}$ in a natrual way: Initially $C_{1,1}, \ldots, C_{1,p}$ are the clusters of $G_{\pi,t}$, then $s_{j,i}$ vertices of U_i are added to cluster $C_{1,j}$, and finally the remaining isolated vertices are added to $G_{\pi,t}$ as singleton clusters. If $G_{\pi,t}$ has at least two nonsingleton clusters, we define the *cost* of $G_{\pi,t}$ to be $D(G, G_{\pi,t})$; otherwise, we define the *cost* of $G_{\pi,t}$ to be $D(G, G_{\pi,t}) + (2 - b)$, where b is the number of nonsingleton clusters in $G_{\pi,t}$. Now, the last idea is to find the cheapest $G_{\pi,t}$. From this $G_{\pi,t}$, it is easy to construct an approximate Δ-phylogeny T of G such that $D(T^2, G)$ equals the cost of $G_{\pi,t}$. One can easily verify that $D(T^2, G) = D(T^2_{\text{opt}}, G)$. □

5 Approximation Algorithm for 3CPR_3

As mentioned before (in Section 1.2), ΔCPR_3 is much more difficult to approximate than ΔCPR_2. In this section, trying to give some insight into ΔCPR_3, we consider the simplest case of ΔCPR_3, namely, 3CPR_3. Indeed, except Lemma 5.4, all the lemmas in this section can be generalized to ΔCPR_3 with the constant factors being replaced by appropriate factors depending on Δ.

Throughout this section, G denotes a graph with at least six vertices and T_{opt} denotes an approximate 3-phylogeny of G such that $D(G, T^3_{\text{opt}})$ is minimized over all approximate 3-phylogenies of G. Our goal is to design a quadratic-time approximation algorithm that outputs an approximate 3-phylogeny T of G with $D(G, T^3) \leq 12 \cdot D(G, T^3_{\text{opt}}) + 3$.

First, several definitions are necessary. Two vertices u and v of G are *indistinguishable* if $\{u, v\} \in E(G)$ and $N_G(u) \cup \{u\} = N_G(v) \cup \{v\}$. Construct an auxiliary graph $A = (V, E_A)$, where E_A consists of all $\{u, v\}$ such that u and v are indistinguishable. Obviously, each connected component of A is a clique of both A and G, and is hence called a *critical clique* of G. Let M be a maximum matching of A. Since A is simply a collection of disjoint cliques, computing M is trivial. Further construct two auxiliary graphs B and H by performing the following steps in turn:

1. Initialize B to be a copy of G, and then assign a unit weight to each edge of B.
2. While M is nonempty, perform the following steps:
   (a) Select an arbitrary edge $\{u, v\} \in M$, and delete it from M.
   (b) Modify B by merging u and v into a supervertex $s(u, v)$. (*Comment*: Each edge incident to $s(u, v)$ can have at most one other edge parallel to it.)

(c) For each pair of parallel edges e_1 and e_2 incident to $s(u, v)$, delete e_2 and add the weight of e_2 to that of e_1.

3. Compute a maximum-weight spanning subgraph H of B such that the degree of each supervertex in H is at most 1 and the degree of each (other) vertex in H is at most 2. (*Comment*: This step takes $O(|V(B)|^2)$ time if we use Pulleyblank's algorithm for the b-matching problem [8].)

4. While H has a supervertex, perform the following steps:

   (a) Select an arbitrary supervertex $s(u, v)$.

   (b) Modify H by (1) splitting $s(u, v)$ back into the two original vertices u and v, (2) connecting u and v by an edge, and (3) replacing each edge $\{s(u, v), w\}$ originally incident to $s(u, v)$ in H with the two edges $\{u, w\}$ and $\{v, w\}$. (*Comment*: w may be a vertex or supervertex of H.)

Note that the weight of each edge of B is 1, 2, or 4. In more details, the weight of an edge of B is 4 if both its endpoints are supervertices, is 2 if exactly one of its endpoints is a supervertex, and is 1 if both its endpoints are not supervertices.

The following lemma shows that $D(G, H) = |E(G) - E(H)| + |E(H) - E(G)| = |E(G) - E(H)| \leq 4 \cdot D(G, T^3_{\mathrm{opt}})$. To see this, let B' be the graph obtained by modifying the above construction of B by replacing G with T^3, where T is as in the following lemma. Obviously, B' is a subgraph of B, the degree of each supervertex in B' is at most 1, and the degree of each (other) vertex in B' is at most 2. Thus, $D(G, H) = |E(G) - E(H)| \leq |E(G) - E(T^3)| = D(G, T^3) \leq 4 \cdot D(G, T^3_{\mathrm{opt}})$.

Lemma 5.1 *G has an approximate 3-semi-phylogeny T with the following properties:*

1. *T^3 is a subgraph of G,*

2. *Two vertices u and v of G are siblings in T if and only if $\{u, v\} \in M$, and*

3. *$D(G, T^3) \leq 4 \cdot D(G, T^3_{\mathrm{opt}})$.*

Proof. If two vertices of G are indistinguishable, then we can exchange their positions in T_{opt} without altering $D(G, T^3_{\mathrm{opt}})$. So, we can assume that if two vertices u and v of G are indistinguishable and are siblings in T_{opt}, then $\{u, v\} \in M$.

We initialize T to be a copy of T_{opt}, and start to modify T so that it satisfies the conditions (1) and (2) in the lemma. In the course of modifying T, we may disconnect T, but we will always maintain that T is an approximate 3-semi-phylogeny of G.

We next detail how to modify T. We say that an edge e of T is *deletable*, if deleting e from T does not increase $D(G, T^3)$. We repeat deleting deletable edges from T until it has no deletable edges. Then, we can claim that T^3 is a subgraph of G. For a contradiction,

assume that u and v are nonadjacent vertices of G with $d_T(u,v) \leq 3$. Obviously, one of the following two cases occurs:

Case 1: $d_T(u,v) = 2$. In this case, u and v are siblings in T. Thus, u can have at most one other neighbor than v in T^3 because $|V(G)| \geq 6$ and T is a subgraph of T_{opt}. Let e be the edge incident to u in T. If we delete e from T, then we get rid of the disagreement $\{u,v\}$ between G and T^3, and can get at most one new disagreement between G and T^3. So, deleting e from T does not increase $D(G,T^3)$, a contradiction against the nonexistence of deletable edges in T.

Case 2: $d_T(u,v) = 3$. In this case, u and v are not siblings in T. Let x (respectively, y) be the node of T adjacent to u (respectively, v). Since $d_T(u,v) = 3$, $\{x,y\}$ is an edge of T. If neither x nor y is adjacent to a vertex of G other than u and v, then deleting the edge $\{x,y\}$ from T removes the disagreement $\{u,v\}$ between G and T^3 without incurring a new disagreement between G and T^3, a contradiction against the nonexistence of deletable edges in T. Moreover, at most one of x and y is adjacent to a vertex of G other than u and v because $V(G) \geq 6$ and T is a subgraph of T_{opt}. So, we may assume that x is adjacent to a vertex w of G other than u but v is the unique vertex of G adjacent to y in T. Now, if we delete edge $\{x,y\}$ from T, then we get rid of the disagreement $\{u,v\}$ between G and T^3, and can get at most one new disagreement (namely, $\{v,w\}$) between G and T^3. So, deleting e from T does not increase $D(G,T^3)$, a contradiction against the nonexistence of deletable edges in T.

So, the above claim holds. By the claim, it remains to modify T so that u and v become siblings in T for every $\{u,v\} \in M$. Let Γ be the set of disagreements between G and T^3 *at this point of time*. Note that $|\Gamma| \leq D(G,T^3_{\text{opt}})$. When we modify T in the future, we may increase $D(G,T^3)$ but we will charge the increase to the pairs in Γ so that each pair in Γ gets a total charge of at most 3.

We modify T by performing two types of operations on T. Either type of operations may delete existing edges from T, add new nodes and edges to T, and mark some *good* connected components of T. Here, a connected component C of T is *good* if there is an edge $\{u,v\}$ in M such that u and v are siblings in C and C has exactly three nodes (including u and v). Before and after performing either type of operations, we will maintain the following five invariants:

- T is an approximate 3-semi-phylogeny of G.
- T^3 is a subgraph of G.
- Each unmarked connected component of T has at least one node x with $x \notin V(G)$ and $deg_T(x) \leq 2$.
- Each marked connected component of T is good.
- For each disagreement $\{u,v\}$ between G and T^3 that is not contained in Γ, u or v is contained in a marked connected component of T.

We next define the first type of operations on T. This type of operations can be applied whenever there is an edge $\{u, v\} \in M$ satisfying the following three conditions:

1. u and v are not siblings in T.
2. The set $S_{u,v} = (N_{T^3}(u) - \{v\}) \cup (N_{T^3}(v) - \{u\})$ contains at most one vertex of G.
3. If $S_{u,v}$ contains a (unique) vertex w, then $N_{T^3}(w) \subseteq \{u, v\}$.

Given an edge $\{u, v\} \in M$ satisfying the above conditions, the *first-type operation* works on T as follows:

1. For each $w \in \{u, v\} \cup S_{u,v}$, if T has an edge incident to w, then delete the edge from T.
2. Introduce a new internal node x and add edges $\{x, u\}$ and $\{x, v\}$ (so that u and v become siblings in T).
3. If $|S_{u,v}| = 1$, then introduce another internal node y and add edges $\{x, y\}$ and $\{y, w\}$, where w is the unique vertex in $S_{u,v}$.

Obviously, performing the first-type operation on T does not increase $D(G, T^3)$, and does not violate the above invariants.

We next define the second type of operations on T. Note that the second-type operation can be applied to T only when the first-type operation cannot be applied to T. Given an edge $\{u, v\} \in M$ such that u and v are not siblings in T, the *second-type operation* works on T as follows:

1. If T has an edge incident to u, delete the edge from T.
2. If T has an edge incident to v, delete the edge from T.
3. Introduce a new internal node x, and add edges $\{x, u\}$ and $\{x, v\}$ (so that u and v become siblings in T).
4. Mark the connected component of T containing u and v.

Obviously, performing the second-type operation on T does not violate the above invariants, but may increase $D(G, T^3)$. We next investigate the increase by a case analysis. For convenience, we use T_b (respectively, T_a) to denote the tree T before (respectively, after) applying the second-type operation for a given edge $\{u, v\} \in M$.

Case (a): $d_{T_b}(u, v) \geq 4$. Let $S = N_{T_b^3}(u) \cap N_{T_b^3}(v)$. Since the maximum degree of T is at most 3, $|S| \leq 1$. Moreover, if w is a vertex in $N_{T_b^3}(u) - S$, then $\{v, w\}$ is in $E(G) - E(T_b^3)$ and is hence in Γ by the last invariant above. Similarly, if w is a vertex in $N_{T_b^3}(v) - S$, then $\{u, w\} \in \Gamma$. Obviously, $D(G, T_a) - D(G, T_b)$ is $|N_{T^3}(u)| + |N_{T^3}(v)| - 1$; we evenly charge

$D(G, T_a) - D(G, T_b)$ to the pairs in $\{\{u, v\}\} \cup \{\{v, w\} \mid w \in N_{T_b^3}(u) - S\} \cup \{\{u, w\} \mid w \in N_{T_b^3}(v) - S\}$. Clearly, each of the pairs has not been charged before (by the last invariant above), and gets a charge of at most 1 here (because $|S| \leq 1$).

Case (b): $d_{T_b}(u, v) = 3$. Let x (respectively, y) be the internal node adjacent to u (respectively, v) in T. Note that $\{x, y\}$ is an edge in T. Let n_u (respectively, n_v) be the number of vertices of G other than u (respectively, v) adjacent to x (respectively, y) in T. By the third invariant above, $n_u + n_v \leq 1$. For convenience, let $W_u = N_{T_b^3}(u) - (\{v\} \cup N_{T_b^3}(v))$ and $W_v = N_{T_b^3}(v) - (\{u\} \cup N_{T_b^3}(u))$. Since the first-type operation is not applicable to T_b, $|W_u| + |W_v| \geq 1$. Moreover, if w is a vertex in W_u, then $\{v, w\}$ is in $E(G) - E(T_b^3)$ and is hence in Γ by the last invariant above. Similarly, if w is a vertex in W_v, then $\{u, w\} \in \Gamma$. Obviously, $D(G, T_a) - D(G, T_b)$ is $|W_u| + |W_v| + 2(n_u + n_v)$; we evenly charge $D(G, T_a) - D(G, T_b)$ to the pairs in $\{\{v, w\} \mid w \in W_u\} \cup \{\{u, w\} \mid w \in W_v\}$. Clearly, each of the pairs has not been charged before (by the last invariant above), and gets a charge of at most 3 here (because $n_u + n_v \leq 1$ and $|W_u| + |W_v| \geq 1$).

We repeat performing the above two types of operations on T until none of them is applicable. Then, it is clear that T has the first two properties in the lemma. Moreover, by the above invariants, each pair in Γ is charged at most once. So, T also satisfies the third property in the lemma. □

As mentioned before, Lemma 5.1 implies that $D(G, H) = |E(G) - E(H)| \leq 4 \cdot D(G, T_{\text{opt}}^3)$. It remains to construct an approximate 3-phylogeny T from H such that $D(H, T^3)$ is not so large compared to $D(G, T_{\text{opt}}^3)$. To this end, first note that our construction of H implies that each connected component C of H satisfies one of the following:

- C is a K_1 (i.e., a vertex).
- C is a K_2 (i.e., an edge).
- C is a K_3 (i.e., a triangle).
- C is a K_4.
- C is a *long* path (i.e., a path of length 2 or more).
- C is a *long* cycle (i.e., a cycle of length 4 or more).
- C is a cane.
- C is a double-ended cane.
- C is a *degenerate cane* (i.e., a graph with five vertices in which the degree of one vertex is 4 and the degree of each other vertex is 2).

Lemma 5.2 *Suppose that G is connected. Then, we can construct an approximate 3-phylogeny T of G with $D(G, T^3) \leq 9 \cdot D(G, T_{\text{opt}}^3) + 2$ in quadratic time.*

PROOF. First consider the case where H is connected. In this case, since $|V(H)| = |V(G)| \geq 6$, our construction of H implies that H is a long path, long cycle, cane, or double-ended cane. If H is a double-ended cane, then it has a 3rd root phylogeny T (cf. Corollary 3.2) and so we have $D(G, T^3) = D(G, H) \leq 4 \cdot D(G, T^3_{\text{opt}})$. So, we may assume that H is not a double-ended cane. Then, G is not a double-ended cane, either (by the construction of H). Thus, $D(G, T^3_{\text{opt}}) \geq 1$ by Corollary 3.2. Moreover, we can transform H to a double-ended cane by deleting at most one edge and adding at most two edges, and further construct a 3rd root phylogeny T of the double-ended cane. Obviously, $D(G, T^3) \leq D(G, H) + D(H, T^3) \leq 4 \cdot D(G, T^3_{\text{opt}}) + 3 \leq 7 \cdot D(G, T^3_{\text{opt}})$, because $D(G, T^3_{\text{opt}}) \geq 1$ and $D(G, H) \leq 4 \cdot D(G, T^3_{\text{opt}})$.

Next consider the case where H is disconnected. Let n_c be the number of connected components in H. Since G is connected, $|E(G) - E(H)| \geq n_c - 1$. So, $n_c \leq D(G, H) + 1 \leq 4 \cdot D(G, T^3_{\text{opt}}) + 1$. We construct an approximate 3-phylogeny of G in three steps as follows.

Step 1: For each connected component C of H that is a K_4, long cycle, double-ended cane, or degenerate cane, we delete the fewest edges from C so that it becomes a proper subgraph of a double-ended cane. Let n_e be the total number of edges deleted in this step. (*Comment*: Obviously, $n_e \leq D(G, T^3_{\text{opt}})$. Moreover, after this step, every connected component of H is a proper subgraph of a double-ended cane.)

Step 2: Whenever H has a connected component C that is a path, we use a new edge to connect C to another connected component C' so that they together form a K_2, long path, or cane. Let n_a be the total number of edges added here. (*Comment*: Obviously, $n_a \leq n_c - 1$. Moreover, after this step, H is a path, a cane, or a collection of triangles and canes.)

Step 3: If H has at most two connected components, we transform H into a double-ended cane by adding at most two more edges to H, and then construct a 3rd root phylogeny T of H. Otherwise, H has at least three connected components and we can construct a 3rd root phylogeny T of H in linear time (as shown in Lemma 6 in [4]).

In summary, the total number of edges deleted or added in Steps 1 through 3 is at most $n_e + n_a + 2 \leq 5 \cdot D(G, T^3_{\text{opt}}) + 2$. So, $D(G, T^3) \leq 9 \cdot D(G, T^3_{\text{opt}}) + 2$. Moreover, our algorithm clearly runs in quadratic time. □

Hereafter, we assume that G is disconnected. If a connected component C of G is a single vertex or a path of length at least 2, then we say that C is *troublesome*; otherwise, we say that C is *helpful*. Note that each troublesome connected component of G is also a connected component of H.

Lemma 5.3 *For each helpful connected component C of G, we can use H to construct an approximate connected 3-semi-phylogeny T_C of C in quadratic time such that T_C has exactly one node of degree 2. Moreover, the total number of edges deleted or added in all the constructions of the approximate 3-semi-phylogenies T_C is at most $5 \cdot D(G, T^3_{\text{opt}}) + 1$.*

PROOF. A simple modification of the proof of Lemma 5.2. □

We want to use the troublesome connected components of G and the approximate 3-semi-phylogenies T_C to construct an approximate 3-semi-phylogeny T of G. The following lemma shows that we can do this without incurring too many new disagreements.

Lemma 5.4 *Let n_t (respectively, n_h) be the number of troublesome (respectively, helpful) connected components of G. Then, $D(G, T^3_{\text{opt}}) \geq \frac{1}{3}(n_t - n_h + 1)$.*

PROOF. We first define several notations as follows:

- n_1: The number of troublesome connected components of G that are also connected components of T^3_{opt}.
- n_2: The number of troublesome connected components C of G such that T^3_{opt} has a connected component C' with $V(C) = V(C')$ and $E(C) \subset E(C')$.
- n_3: The number of troublesome connected components C of G such that T^3_{opt} has a connected component C' with $V(C) \subset V(C')$ and $E(C) \subset E(C')$.
- n_4: The number of troublesome connected components C of G such that at least one edge of C is not in T^3_{opt}.
- m_1: The number of edges $e \in E(G) - E(T^3_{\text{opt}})$ such that e appears in a troublesome connected component of G.
- m_2: The number of edges $e \in E(G) - E(T^3_{\text{opt}})$ such that e appears in a helpful connected component of G.

Obviously, $n_t = n_1 + n_2 + n_3 + n_4$, $m_1 \geq n_4$, and $D(G, T^3_{\text{opt}}) \geq n_2 + n_3/2 + m_1 + m_2$.

Let n_c be the number of connected components in T^3_{opt}. Obviously, $n_c \leq n_1 + n_2 + n_3/2 + (n_4 + m_1) + (n_h + m_2)$. We claim that $n_c \geq 2n_1 + 1$; the proof is given in the next paragraph. By the claim, $n_2 + n_3/2 + m_1 + m_2 \geq n_1 - n_4 - n_h + 1$. So, $D(G, T^3_{\text{opt}}) \geq n_1 - n_4 - n_h + 1$. Hence, $D(G, T^3_{\text{opt}}) \geq \frac{2}{3}(n_2 + \frac{n_3}{2} + m_1 + m_2) + \frac{1}{3}(n_1 - n_4 - n_h + 1) \geq \frac{1}{3}(n_t - n_h + 1)$.

It remains to prove the claim. Let $C_1, \ldots, C_{n_1}$ be the troublesome connected components of G that are also connected components of T^3_{opt}. For each $i \in \{1, \ldots, n_1\}$ such that C_i is a single vertex v_i, let T_i be the edge between v_i and its neighbor in T_{opt}. Moreover, for each $i \in \{1, \ldots, n_1\}$ such that C_i is a path of length 2 or more, let T_i be the smallest subtree of T_{opt} containing the vertices of C_i (note that the vertices of C_i are the leaves of T_i and removing them from T_i yields a path). Furthermore, for each $i \in \{1, \ldots, n_1\}$, we say that an edge e of T_{opt} is *associated with* C_i if e is not in T_i but is incident to a node of T_i. Obviously, for each $i \in \{1, \ldots, n_1\}$, exactly two edges are associated with C_i. Moreover, since each C_i is a connected component of T^3_{opt}, each edge of T_{opt} can be associated with at most one C_i. The crucial point is that removing an edge associated with a C_i increases

the number of connected components of T_{opt} by 1 and does not affect T^3_{opt}. Therefore, if we remove the $2n_1$ edges associated with $C_1, \ldots, C_{n_1}$ from T_{opt}, we obtain a forest with $2n_1 + 1$ connected components and T^3_{opt} remains the same as before. This completes the proof of the claim. □

For an approximate 3-semi-phylogeny F of G, we say that a connected component C of F is *helpful* if C has exactly one node of degree 2, and say that C is *troublesome* otherwise.

Theorem 5.5 *We can construct an approximate 3-phylogeny T of G with $D(G, T^3) \leq 12 \cdot D(G, T^3_{\mathrm{opt}}) + 3$ in quadratic time.*

Proof. By Lemma 5.2, we may assume that G is disconnected. For each connected component C of G, we construct an approximate connected 3-semi-phylogeny T_C as follows. If C is helpful, then we use H to construct T_C as in Lemma 5.3; we call the node of degree 2 in T_C the *port* of T_C. If C is troublesome and is a single vertex v, we construct T_C by simply letting $T_C = v$. If C is troublesome and is a path of length 2 or more, we can easily construct T_C with $D(C, T^3_C) = 0$ such that T_C contains exactly two nodes of degree 2; we call each node of degree 2 in T_C a *port* of T_C. Let T be the forest whose connected components are the trees T_C. Then, by Lemmas 5.1 and 5.3, $D(G, T^3) \leq 9 \cdot D(G, T^3_{\mathrm{opt}}) + 1$.

We next connect the connected components T_C of T into an approximate 3-phylogeny of G as follows:

1. While T has at least two helpful connected components and $\mathcal{C}_t$ has at least one troublesome connected component, modify T by performing the following steps:
   (a) Pick two arbitrary helpful connected components T_1 and T_2, and pick one arbitrary troublesome connected component T_3.
   (b) Connect T_1 and T_2 into a single tree T_4 by introducing a new node x and connecting it to the unique port of T_1 and that of T_2.
   (c) If T_3 is a single node, then we connect T_3 and T_4 into a single tree T_5 by introducing a new node y and connecting it to x and the node of T_3. (*Comment*: y is the only node of degree 2 in T_5, and we call it the *port* of T_5. Also note that $D(G, T^3)$ remains unchanged in this step.)
   (d) If T_3 is not a single node, then we connect T_3 and T_4 into a single tree T_5 by connecting x to one port of T_3. (*Comment*: Since T_3 is not a single node, it has exactly two ports. So, T_5 has exactly one port. Also note that $D(G, T^3)$ remains unchanged in this step.)
2. While T has at least two helpful connected components, modify T by performing the following steps:
   (a) Pick two arbitrary helpful connected components T_1 and T_2.

(b) Connect T_1 and T_2 into a single tree T_3 by introducing a new node x and connecting it to the unique port of T_1 and that of T_2. (*Comment*: $D(G, T^3)$ remains unchanged in this step.)

3. Let p (respectively, q) be the number of troublesome (respectively, helpful) connected components in T. (*Comment*: $p \leq n_t - \max\{0, n_h - 1\}$.)

4. If $p = 0$, then modify T by deleting the unique degree-2 node and connecting its two original neighbors, and halt. (*Comment*: This step increases $D(G, T^3)$ by at most 3.)

5. Let S_t be the subgraph of T^3 induced by the set of those vertices of G that appear in troublesome connected components of T. (*Comment*: S_t is a collection of p vertex-disjoint paths. Moreover, S_t cannot be a single edge.)

6. If $q = 0$, then transform S_t into a double-ended cane by adding $p + 1$ new edges, replace T by a 3rd root phylogeny of the double-ended cane, and halt. (*Comment*: This step increases $D(G, T^3)$ by $p + 1$.)

7. If S_t is a single vertex v, then modify T by using a new edge to connect v to the unique degree-2 node of the helpful connected component of T, and halt. (*Comment*: This step increases $D(G, T^3)$ by at most 3.)

8. Transform S_t into a cane by adding p new edges.

9. Construct a connected approximate 3-semi-phylogeny T_t of S_t with $D(S_t, T_t^3) = 0$ such that T_t contains exactly one node of degree 2.

10. Modify T by replacing the troublesome connected components of T with T_t and connecting the unique degree-2 node of T_t to that of the helpful connected component of T. (*Comment*: This step increases $D(G, T^3)$ by at most $p + 2$.)

The above steps increase $D(G, T^3)$ by at most $\max\{3, p + 2\} \leq n_t - n_h + 3$, which does not exceed $3 \cdot D(G, T^3_{\text{opt}}) + 2$ by Lemma 5.4. Thus, by the last inequality in the first paragraph of this proof, $D(G, T^3) \leq 12 \cdot D(G, T^3_{\text{opt}}) + 3$. Moreover, our algorithm clearly runs in quadratic time. □

Remark. As shown in [4], we can decide whether a given graph has a 3rd root 3-phylogeny in linear time. So, if $D(G, T^3_{\text{opt}})$ is a constant, then we can construct T_{opt} in polynomial time. Hence, Theorem 5.5 indeed implies a polynomial-time r-approximation algorithm for 3CPR$_3$, where r is asymptotically 12.

6 Open Problems

This work leaves quite a few open problems. The first asks for a nontrivial approximation algorithm for CPR_k or its maximization version for $k \geq 3$. A seemingly easier problem is to ask whether $\Delta\mathrm{CPR}_k$ admits a polynomial-time r-approximation algorithm, where r is a constant (preferably independent of Δ and k).

Theorem 3.5 says that if we can efficiently construct a k-densest Δ-phylogeny with a given number of leaves, then we have a very efficient PTAS for the maximization version of $\Delta\mathrm{CPR}_k$. So, it would be very interesting if we could find out the structure of a k-densest Δ-phylogeny with a given number of leaves for $k \geq 4$ and $\Delta \geq 3$.

Finally, we want to ask whether CPR_k and $\Delta\mathrm{CPR}_k$ are fixed-parameter tractable or not.

Acknowledgments

The author would like to thank the referees for helpful comments.

References

[1] N. Bansal, A. Blum, and S. Chawla. Correlation Clustering. *Machine Learning*, 56(1-3):89-113, 2004.

[2] M. Charikar, V. Guruswami, and A. Wirth. Clustering with Qualitative Information. *Proceedings of 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, pp. 524-533, 2003.

[3] Z.-Z. Chen, T. Jiang, and G.-H. Lin. Computing Phylogenetic Roots with Bounded Degrees and Errors. *SIAM Journal on Computing*, 32(4):864-879, 2003. A preliminary version appeared in *Proceedings of 7th International Workshop on Algorithms and Data Structures (WADS'01)*, Lecture Notes in Computer Science, 2125:377-388, 2001.

[4] Z.-Z. Chen and T. Tsukiji. Computing Bounded-Degree Phylogenetic Roots of Disconnected Graphs. *Journal of Algorithms*, 59(2):125-148, 2006. A preliminary version appeared in *Proceedings of 30th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'04)*, Lecture Notes in Computer Science, 3353:308-319, 2004.

[5] E. Dahlhaus, P. Hajnal, and M. Karpinski. On the Parallel Complexity of Hamiltonian Cycle and Matching Problem on Dense Graphs. *Journal of Algorithms*, 15:367-384, 1993.

[6] R. Hassin and S. Rubinstein. An Improved Approximation Algorithm for the Metric Maximum Clustering Problem with Given Cluster Sizes. *Information Processing Letters*, 98:92-95, 2006.

[7] G.-H. Lin, P. E. Kearney, and T. Jiang. Phylogenetic k-Root and Steiner k-Root. *Proceedings of the 11th Annual International Symposium on Algorithms and Computation (ISAAC 2000)*, Lecture Notes in Computer Science, 1969:539-551, 2000.

[8] W.R. Pulleyblank. *Faces of Matching Polyhedra*. Ph.D. Thesis, Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, 1973.

[9] R. Shamir, R. Sharan, and D. Tsur. Cluster Graph Modification Problems. *Discrete Applied Mathematics*, 14:173-182, 2004.

[10] D. L. Swofford, G. J. Olsen, P. J. Waddell, and D. M. Hillis. Phylogenetic Inference. In D. M. Hillis, C. Moritz, and B. K. Mable, editors, *Molecular Systematics (2nd Edition)*, pp. 407-514, Sinauer Associates, Sunderland, Massachusetts, 1996.

[11] T. Tsukiji and Z.-Z. Chen. Computing Phylogenetic Roots with Bounded Degrees and Errors Is Hard. To appear in *Theoretical Computer Science*. A preliminary version appeared in the *Proceedings of 10th Annual International Computing and Combinatorics Conference (COCOON'04)*, Lecture Notes in Computer Science, 3106:450-461, 2004.

[12] R.R. Weitz and S. Lakshminarayanan. An Empirical Comparison of Heuristic Methods for Creating Maximally Diverse Groups. *Journal of the Operational Research Society*, 49:635-646, 1998.