An Approximation Algorithm for Bounded Degree Closest Phylogenetic 2nd Root Problem

Swakkhar Shatabda[*] Atif Rahman[*] Masud Hasan[*]

Abstract—**Given a graph $G = (V, E)$ and two positive integers k and Δ, the Δ-closest phylogenetic k-th root problem (ΔCPR_k) is to find a (phylogenetic) tree T such that the degree of each internal node in T is at least three and at most Δ, the external nodes of T are exactly the elements of V, and the number of "k-disagreements" is minimized. In this paper we give an approximation algorithm for ΔCPR_2 for any fixed $\Delta > 3$. The expected ratio of our algorithm is 3, if there is no vertex with degree more than $\Delta - 2$, and $\Delta + 2$, otherwise, which improves the best known previous ratio of 8 for $\Delta \leq 5$.**

Keywords: phylogenetic tree, clustering, approximation algorithm, randomized algorithm

1 Introduction

Phylogenetic trees are used by biologists to depict the evolutionary relationship among species, organisms or genes. The leaves of the tree represent the species and each branching corresponds to a speciation event. Such an event causes an ancestral species to give rise to two or more child species [5].

A number of phylogenetic tree construction methods have been proposed in the literature such as *parsimony methods, distance-based methods, maximum-likelihood methods,* etc. Recently, a *graph-theoretic* approach was suggested by Lin and others [9]. A *graph* $G = (V, E)$ consists of a set of vertices V and a set of edges E where an edge $(u, v) \in E$ connects two vertices $u, v \in V$. The graph is usually derived from similarity data. The vertices of graph G represent existing species and an edge connecting two vertices represents the evolutionary similarity between them.

The *degree* of a vertex $v \in V$ is the number of incident edges of v in G. A *path* in G is an alternating sequence of distinct vertices and edges of G, starting and ending with two vertices, in which each edge is incident to two vertices immediately preceding and following it. A *cycle* in G is a path with the exception that the first and last vertices in the sequence are the same. The *length* of a path or cycle is the number of edges in it. A graph G is *connected* if for any two vertices u and v in G there is a path between u and v. A *cluster* of G is a subgraph of G such that every pair of vertices in the subgraph is connected by an edge.

A *tree* is a connected graph with no cycle. In a tree, an *internal node* has degree at least two and an *external node*, or a *leaf*, has degree exactly one. The *distance* between two vertices u and v in a tree T is the length of the (unique) path between u and v in T and is denoted by $d_T(u, v)$.

In the graph-theoretic approach, a phylogeny is constructed from a graph in such a way that the leaves of the phylogeny are labeled by the vertices of the graph and two vertices are adjacent in the graph if and only if the corresponding leaves in the tree are connected by a path of length at most k, where k is a chosen proximity threshold.

Formally, given a connected graph G and an integer $k \geq 2$, the *k-th root phylogeny* of G, if it exists, is a tree T where the degree of each internal node of T is at least 3, the leaves of T are labeled by the vertices of V, and for each pair of vertices $u, v \in V$, $(u, v) \in E$ iff $d_T(u, v) \leq k$. Given G and an integer k, the *phylogenetic k-th root problem (PR_k)* is to find a *k-th root phylogeny* of G. For a given Δ, a restricted version of PR_k is *Δ-phylogenetic k-th root problem (ΔPR_k)*, where the maximum degree of the phylogenetic tree is Δ.

However, G is usually derived from some similarity data, which are inexact in practice and may contain some erroneous data. So, such data may result in a graph that has no phylogeny, which motivates to have an approximate phylogeny [6].

A *k-disagreement* between G and its approximate k-th root phylogeny T is a pair of vertices u and v of G such that either (i) $(u, v) \in E$ and $d_T(u, v) > k$, or (ii) $(u, v) \notin E$ and $d_T(u, v) \leq k$. Given G and an integer k, the *closest phylogenetic k-th root problem (CPR_k)* is to find an *approximate* k-th root phylogeny of G where the number of k-disagreements is minimum. For a given Δ, a restricted version of CPR_k is *Δ-approximate phylogenetic k-th root problem (ΔCPR_k)*, where the maximum

[*]Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka-1000, Bangladesh. Phone: +88 01556 634 106, Fax: +88 02 966 5612. *E-mail addresses:* swakkhar17@yahoo.com (S.Shatabda), atif.bd@gmail.com (A.Rahman), masudhasan@cse.buet.ac.bd (M.Hasan)

degree of the phylogenetic tree is Δ.

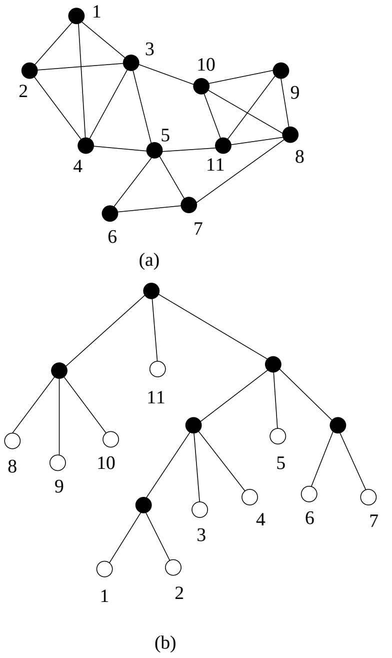

Figure 1: (a) A given graph, and (b) its 4-approximate phylogeny with fewest (two) 3-disagreements (which are (3,10) and (7,8).)

In an evolutionary process, a species usually gives rise to two child species. So the most common practice in phylogeny reconstruction is to compute phylogenetic trees of degree three. But phylogenetic trees with nodes of degree more than three are also required if enough biological data is not available to separate individual speciation events and several such events are collapsed into a super speciation event [6].

1.1 Related results

Phylogenetic root problems have been studied a lot for both exact phylogeny and approximate phylogeny [5, 6, 8, 9, 10, 11, 13]. For exact phylogeny there exist several algorithms to decide whether a graph has a phylogeny or not. Kearney and Corneil [8] have an $O(n^3)$-time algorithm to determine whether a given graph G has a k-th root phylogeny for a given k, where n is the number of vertices of G. For $k = 2$, Lin and Skiena [10] have an $O(n + m)$-time algorithm for this problem, where m is number of edges in G. Nishimura *et. al.* [11] have given an $O(n^3)$-time algorithm for $k \leq 4$ but the internal nodes of the phylogeny are allowed to have degree two.

For ΔPR_k, Chen *et. al.* [6] presented a linear time algorithm to decide whether G has a Δ-approximate k-th root phylogeny and to find one if it exists.

In the contrary, most of the problems of computing approximate phylogeny are intractable. Chen *et. al.* [6] showed that CPR_k is NP-hard for $k \geq 2$. For ΔCPR_k, Tsukiji and Chen [13] showed that it is NP-hard if $\Delta \geq 3$ and $k \geq 3$ or $\Delta \geq 4$ and $k = 2$. Exceptionally, $3CPR_2$ is identical to the *maximum matching problem* in graph theory [5] and thus efficiently solvable in polynomial time [14].

There exist several approximation algorithms for the ΔCPR_k (although all of them are due to Chen [5].) Chen [5] showed that for any given $\Delta \geq 3$ and $k \geq 2$, ΔCPR_k has a polynomial time approximation scheme for the maximization version of the problem. They have a polynomial time 8-approximation algorithm for ΔCPR_2, where $\Delta \geq 3$, and a quadratic time 12-approximation algorithm for $3CPR_3$.

There are many problems that are in one way or other related to the problems of finding phylogeny. CPR_2 is closely related to the *correlation clustering* problem [3]. Given a complete graph G with each edge labeled as '+' or '-', a correlation clustering problem is to partition G into clusters such that the number of '-' edges within clusters and the number of '+' edges crossing clusters is minimized. There exist variations of the correlation clustering problem and they are intractable in general [1]. Ailon *et. al.* [1] are the most recent who have given an approximation algorithm for the correlation clustering problem. Their algorithm has an expected approximation ratio of 3, which is the best known so far.

Another well studied clustering problem, which is also closely related to ΔCPR_2, is the *maximum clustering problem with given cluster sizes (MCPGCS)* [7]. Given a weighted complete graph G and a sequence of integers $c_1, \cdots, c_p$, MCPGCS requires to find a maximum-weight subgraph of G that has p clusters of weights $c_1, \cdots, c_p$ respectively. Clustering problems have many applications ranging from final exam scheduling to VLSI design [4, 7, 12].

1.2 Our contribution

In this paper we present a polynomial time randomized approximation algorithm for ΔCPR_2 for any fixed $\Delta \geq 3$. The algorithm has an expected approximation ratio of 3, if there is no vertex with degree higher than $\Delta - 2$, or has an upper bound of $3+(\Delta - 1)$, otherwise, where the first term 3 is expected value and a maximum of $\Delta - 1$ is added to it in the worst case.

2 Approximation algorithm

Let $G = (V, E)$ be the given graph of evolutionary relationship. G may not be complete. First we convert G to a complete graph with edges labeled as '+' and '-' as follows. For each pair of vertices u, v in G, if $(u, v) \in E$ we assign it the label '+', otherwise we add an edge (u, v) in G and assign it the label '-'.

For an integer $\Delta \geq 3$, a Δ-*clustering* G_C of G is a partitioning of the vertices of G into clusters of size at most $\Delta - 1$. A pair of vertices (u, v) in G_C is called a *disagreement* if (u, v) is an edge within a cluster but is labeled as '-' or it is an edge crossing a cluster and is labeled as '+'. (See Figure 2). Given G, the *approximate* Δ-*clustering problem* is to find G_C such that the number of disagreements in G_C is minimum.

We have two main steps. First, we find a Δ-clustering G_C for G having an approximation ratio $(\Delta + 2)$. Then we construct a Δ-approximate phylogeny from G_C without adding any new disagreement. For the first step we use the technique of Ailon *et. al.* which they used for solving correlation clustering problem [1]. For the second step our algorithm is same as that of Chen [5] and we only give a sketch.

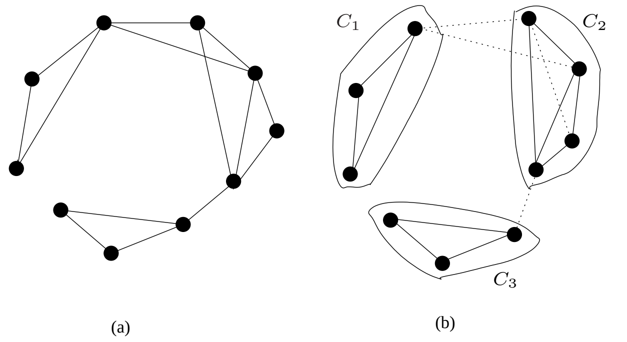

Figure 2: (a) A given graph G. Only (+) edges are shown here. (b) A 5-clustering of G. The disagreements are shown in dotted lines and the (-) edges that are not disagreement are not shown here.

2.1 Algorithm for Δ-clustering

We construct one cluster at a time for G_C. Let C be the cluster we are currently constructing. We will also maintain a set V' that will contain the vertices not assigned to any cluster so far. Initially both C and V' are empty. We pick a vertex u randomly from V and add it to C. For each remaining vertex $v \in V$, $u \neq v$, if (u, v) is a (+) edge of G, we move it to C, otherwise we move it to V'. If the size of C becomes Δ or more, then we remove the extra vertices from C which are chosen arbitrarily and are not u. Let G' be the subgraph of G induced by V'. We assign $G = G'$ and repeat the above procedure to form the next cluster until V becomes empty. See Algorithm 1 for the summary of our algorithm.

Running time for this algorithm is obviously $O(|V|^2)$.

Lemma 2.1. *If there is no vertex in G with degree higher than $\Delta - 2$, then the algorithm* Pivot Δ-Cluster *is an expected 3-approximation algorithm for Δ-clustering problem.*

Proof. Let C_O denote the minimum number of disagreement possible in a clustering of G. Let C_Δ denote the number of disagreement in G_C by our algorithm. We need to show that $C_\Delta \leq 3 \cdot C_O$.

Algorithm 1 Pivot Δ-Cluster(G)

```
1: Set $C = \emptyset$, $V' = \emptyset$
2: Pick a random vertex $u \in V$
3: Add $u$ to $C$
4: for all $v \in V$, $v \neq u$ do
5:    if $(u, v)$ is an (+) edge of $G$ then
6:       Add $v$ to $C$
7:    else
8:       Add $v$ to $V'$
9:    end if
10: end for
11: while $|C| > \Delta - 1$ do
12:    Choose an arbitrary vertex $x \neq u$ from $C$
13:    Remove $x$ from $C$
14:    Add $x$ to $V'$
15: end while
16: Output $C$
17: Let $G'$ be the subgraph induced by $V'$
18: Recursively call Pivot $\Delta$-Cluster($G'$)
```

When the vertices of G have degree no more than $\Delta - 2$, $|C| \leq \Delta - 1$. In that case Line 12 through 14 of Pivot Δ-Cluster are never executed and the expected ratio of disagreement is as follows.

Consider a particular recursive call of Pivot Δ-Cluster(G). Let C be the cluster that results from this recursive call. Consider a *triplet* (i.e., a set of three vertices) (u, v, w) in G. A pair of vertices, say (u, v), of this triplet incurs a disagreement for C if and only if in this recursive call there are two (+) and one (-) relations among u, v, w in C (irrespective of order) and w is chosen at random in Line 2. For example if (u, v) is a (-) edge of G and (u, w) and (v, w) are (+) edges of G, then the (-) edge (u, v) is included in the cluster C causing a disagreement. Again, if (v, w) is a (-) edge and (u, v) and (u, w) are (+) edges and w is selected at random, then v is not included in C resulting in the (+) edge (u, v) crossing boundary of C. We call the triplet (u, v, w) a *bad triplet.* If there is no bad triplets over all recursive calls then the algorithm have zero disagreement, which is optimum. Let T denote the set of bad triplets. For each $t = (u, v, w) \in T$ we define A_t as the event such that all three of u, v, w are in the same recursive call when one among them was chosen at random. Let p_t denote the probability of A_t. Now observe that a triplet t incurs a disagreement exactly when A_t occurs, and it can incur disagreement at most once. Therefore, the expected cost of the algorithm is

$$C_\Delta = \sum_{t \in T} p_t. \tag{1}$$

Conditioned on the event A_t, each of (u, v, w) can be chosen at random with probability 1/3, because all vertices of G in a recursive call are chosen at random with equal probabilities. Therefore (u, v) becomes a disagreement with probability 1/3, conditioned on A_t. Let $B_{(u,v)}$ denote the event that (u, v) becomes a disagreement. Now if A_t occurred then $B_{(u,v)}$ occurred if and only if w was chosen at random before u and v. Then we get for all $t \in B$ and $u, v \in t$,

$$\Pr[B_{(u,v)} \wedge A_t] = \Pr[B_{(u,v)}|A_t]Pr[A_t] = \frac{1}{3}p_t$$

Now for two different triplets $t, t' \in B$ that share a pair of vertices (u, v), the events $B_{(u,v)} \wedge A_t$ and $B_{(u,v)} \wedge A_{t'}$ are disjoint. If t incurs a disagreement due to (u, v), then u and v are not in G' and never picked at random. So the event $A_{t'}$ cannot occur. Therefore for all $(u, v) \in V$,

$$\sum_{t:u,v\in t} \frac{1}{3}p_t \leq 1 \tag{2}$$

Ailon *et. al.* proved that if Eq. 2 holds, then $C_O \geq \sum_{t:u,v\in t} \frac{1}{3}p_t$ [1, Proof of Theorem 1]. (They prove it by modeling it to the fractional packing problem.) Thus by Eq. 1,

$$C_O \geq \sum_{t:e\in t} \frac{1}{3}p_t = C_\Delta/3.$$

□

Next we have the following lemma for the case $|C| > \Delta - 1$.

Lemma 2.2. *The number of additional disagreement incurred by removing the extra vertices from C is no more than $\Delta - 1$ times of that in the optimum clustering.*

Proof. Consider the situation after deleting the $|C|-\Delta+1$ extra vertices in a particular recursive call of Pivot Δ-Cluster. Let C' be the cluster that results from C after removing the vertices. Let x be a vertex that was removed from C. How many disagreement does x incur after its removal? In worst case x has $(\Delta - 1)$-many (+) edges to all vertices in C', and so x can incur a maximum of $\Delta - 1$ disagreements. See Figure 3. Over all removed vertices, the maximum number of disagreement is $(|C| - \Delta + 1) \cdot (\Delta - 1)$.

Let u be the vertex that was chosen randomly in this recursive call (and has degree $|C| - 1$.) In any optimum Δ-clustering of G, u has at most $\Delta - 2$ neighbors in the same cluster and therefore has at least $|C|-\Delta+1$ adjacent edges crossing the boundary. So u incurs at least $|C| - \Delta + 1$ disagreements in that optimum clustering.

Observe that in both cases any edge causing a disagreement is removed. So an edge can incur a disagreement at most once. Therefore, the ratio of disagreement due to the removal of vertices over all recursive calls of Pivot Δ-Cluster and the disagreement in optimum clustering is

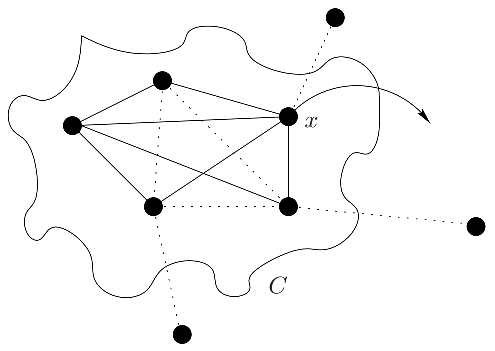

Figure 3: Removing x from C can incur at most $\Delta - 1$ new disagreements.

$$\begin{aligned} r &\leq \frac{\sum(|C| - \Delta + 1) \cdot (\Delta - 1)}{\sum(|C| - \Delta + 1)} \\ &= (\Delta - 1). \end{aligned}$$

□

The following theorem follows from Lemma 2.1 and Lemma 2.2.

Theorem 2.3. *The algorithm* Pivot Δ-Cluster *has an expected approximation ratio of* $\Delta+2$.

2.2 Tree construction

Let G_C be the $(\Delta + 2)$-approximation Δ-clustering of G constructed by the above algorithm. It suffices to show how to construct a 2-phylogeny from G_C without adding any new disagreement. We give only sketch here. Detail can be found in [5].

Let $C_1, \cdots, C_h$ be the clusters of G_C. A cluster C_i may be singleton or not. Observe that any two singleton clusters can not be connected by an edge, otherwise we could combine them together to a cluster of two. Since G has at least one edge, G_C has at least one non-singleton cluster. We have two cases: (i) G_C has at least two non-singleton clusters and (ii) G_C has only one non-singleton cluster. We sketch case (i) only.

Case (i). Suppose C_1 and C_2 are two non-singleton clusters. For each $i = 1, \ldots, h$, create an internal node x_i of T. For $i = 1, 2$, connect each vertex of C_i to x_i. Otherwise, create a subtree T_i for each cluster C_i and connect T_i to x_i by an edge. See Figures 4(a) and (b). Then connect $x_1, x_3, x_4, \ldots, x_h, x_2$ in a path as shown in Figure 4(c).

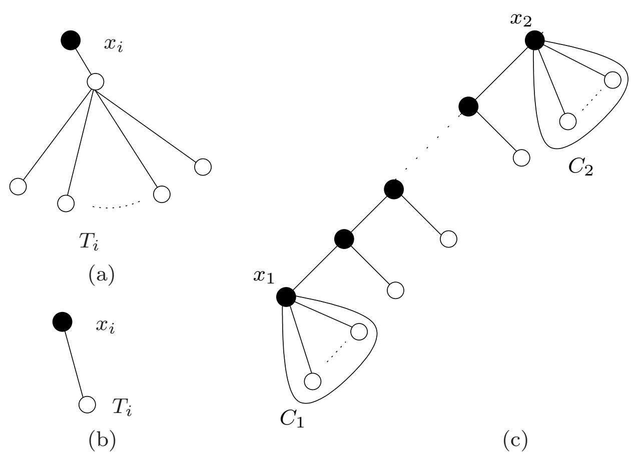

Figure 4: Forming a sub-tree for (a) non-singleton cluster, (b) singleton cluster. (c) Adding those sub-trees in a path to form T.

Within the premise of Case (i), let us justify T. Clearly the minimum degree of T is at least three. Since $|C_i| \leq \Delta - 1$, the maximum degree of T is at most Δ. For disagreements, observe that two vertices (u, v) of a (+) edge (similarly of a (-) edge) within a cluster C_i of G_C have a path length of two in the corresponding sub-tree of C_i and so they remain an agreement (similarly a disagreement) in T. On the other hand, since the distance between two vertices in two different clusters in T is at least three, a (-) edge (similarly a (+) edge) crossing clusters in G_C remains an agreement (similarly a disagreement) in T. Therefore, no new disagreement is added in T.

Theorem 2.4. *There is a polynomial time $(\Delta + 2)$-approximation algorithm for ΔCPR_2 problem.*

3 Conclusion

In our algorithm of constructing Δ-cluster graph we chose the very first vertex for C arbitrarily. It is possible to apply some heuristics here. One possible heuristic could be to choose the vertex with highest degree. This heuristic has been used by Altaf *et. al.* [2] for the clustering problem but they do not give any concrete approximation ratio for it. Another heuristic may be to select a vertex that is part of least number of bad triangles.

While deleting extra vertices from C, we deleted them arbitrarily. Again different heuristics might be applied here, for example, removing nodes with highest disagreement, fewest degree, etc. It would be interesting to see whether any of those heuristics improves the approximation ratio and/or makes the algorithm purely deterministic.

References

[1] Ailon, N., Charikar, M., Newman, A., "Aggregating Inconsistent Information: Ranking and Clustering," *ACM Symposium on Theory of computing (STOC'05)*, pp. 684–693 2005.

[2] Altaf-Ul-Amin, M., Tsuji, H., Kurokawa, K., Asahi, H., Shinbo, Y., Kanaya, S., "DPClus: A Density-periphery Based Graph Clustering Software Mainly Focused on Detection of Protein Complexes in Interaction Networks," *Journal of Computer Aided Chemistry*, V7, pp. 150–156, 2006.

[3] Bansal, N., Blum, A., Chawla, S., "Correlation Clustering," *IEEE Symposium on Foundations of Computer Science (FOCS'02)*, pp. 238–247 2002.

[4] Charikar, M., Guruswami, V., Wirth, A., "Clustering with Qualitative Information," *Journal of Computer and System Science*, V71, N3, pp. 360–383, 2005.

[5] Chen, Z.-Z., "Approximation Algorithms for Bounded Degree Phylogenetic Roots," *Algorithmica*, to appear.

[6] Chen, Z.-Z., Jiang, T., Lin, G.-H., "Computing Phylogenetic Roots with Bounded Degrees and Errors," *SIAM Journal on Computing*, V32, N4, pp. 864–879, 2003.

[7] Hassin, R., Rubinstein, S., "An Improved Approximation Algorithm for the Metric Maximum Clustering Problem with Given Cluster Sizes," *Information Processing Letters*, V98, pp. 92–95, 2006.

[8] Kearney, P. E., Corneil, D. G., "Tree Powers," *Journal of Algorithms*, V29, N1, pp. 111–131, 1998.

[9] Lin, G.-H., Jiang, T., Kearney, P. E., "Phylogenetic k-Root and Steiner k-Root," *International Conference on Algorithms and Computation (ISAAC'00)*, V 1969 of *LNCS*, pp. 539–551 Springer, 2000.

[10] Lin, Y.-L., Skiena, S. S., "Algorithms for Square Roots of Graphs," *SIAM Journal on Discrete Mathematics*, V8, N1, pp. 99–118, 1995.

[11] Nishimura, N., Ragde, P., Thilikos, D. M., "On Graph Powers for Leaf-labeled Trees," *Journal of Algorithms*, V42, N1, pp. 69–108, 2002.

[12] Shamir, R., Sharan, R., Tsur, D., "Cluster Graph Modification Problems," *International Workshop Graph-Theoretic Concepts in Computer Science (WG'02)*, V 2573 of *LNCS*, pp. 379–390 Springer, 2002.

[13] Tsukiji, T., Chen, Z.-Z., "Computing Phylogenetic Roots with Bounded Degrees and Errors is NP-complete," *Theor. Comput. Sci.*, V363, N1, pp. 43–59, 2006.

[14] West, D. B., *Introduction to Graph Theory*, Prentice Hall, 2000.