On Subexponential Running Times for Approximating Directed Steiner Tree and Related Problems

Guy Kortsarz ∗ M. Cygan † Bundit Laekhanukit ‡

January 27, 2020

Abstract

Consider the Set Cover problem with \( n \) elements and \( m \) sets and the following problem: given some constant \( 0 < \alpha < 1 \), what is the time required to give a \( (1-\alpha) \ln n \) approximation for Set Cover? Cygan et al. [IPL, 2009] give an \( (1-\alpha) \cdot \ln n \) approximation for Set Cover with running time \( 2^{O(n^\alpha)} \) [12]. Moshkovitz [26], (indirectly) proves that under the Exponential Time Conjecture conjecture, a \( (1-\alpha) \ln n \) approximation for Set Cover requires \( 2^{n^{\alpha}} \) time, for some small constant \( c < 1 \). Our first result improves [26] and settles the best time required for this problem. Assuming the the Exponential Time Conjecture, we prove that a \( (1-\alpha) \ln n \)-approximation algorithm for Set Cover requires time \( 2^{O(n^\alpha)} \) with \( n \) the number of elements. Hence this is the best time up to constants in the exponent and the improves [26]. The Set Cover problem is a special case of both the Directed Steiner Tree (DST) problem and the Connected Polymatroid problem. Hence the lower bound holds for those problems as well. We complement this result by giving a \( (1-\alpha) \cdot \ln n \) approximation for the DST and the Connected Polymatroid problems for \( \alpha \geq 1/2 \), with \( 2^{O(n^{\alpha} \cdot \log n)} \cdot \text{pol}(m) \). Thus we have an almost tight time for approximating the DST and Connected Polymatroid problems (and many other in-between problem), within \( (1-\alpha) \ln n \) for \( \alpha \geq 1/2 \) as the difference is a small \( \log n \) factor in the exponent of the running time. We explain why we are unlikely to get such an upper bound for any \( \alpha < 1/2 \).

We further study the approximation ratio in the regime of \( \log^{2-\delta} n \) for Group Steiner tree and Covering Steiner tree. Chekuri and Pal [FOCS, 2005] showed that Group Steiner tree admits \( \log^{2-\delta} n \)-approximation in time \( \exp(2^{\log^\alpha n \cdot (1) \cdot n}) \), for any parameter \( 0 < \alpha < 1 \). We prove an almost matching lower bound on the running time for achieving the \( \log^{2-\delta} n \)-approximation.

1 Introduction

The traditional study of approximation algorithms concerns designing algorithms that run in polynomial time while producing a solution whose cost is within some factor \( \alpha \) away from the optimal solution. The Directed Steiner Tree DST problem is a seminal problem in approximation algorithm. One of the most important open questions in approximation algorithm is: does the DST problem admit a poly-logarithmic approximation, that runs in polynomial time? The failure in addressing this question, resulted in papers which ask what is the best ratio for the DST problem for running times that are super-polynomial in \( n \)? Designing hardness results which are super polynomial time in \( n \), require complexity assumptions that are much stronger than \( P \neq NP \). The most common conjecture used is the Exponential Time Conjecture,

‡ Shanghai University, School of Engineering E-mail: lbundit@gmail.com
which states that 3-SAT admits no $2^{o(n)}$-time algorithm. See [23, 7]. Recently, the question of what is the best ratio for DST under quasi-poly time was settled in [19]. The authors show that under the Exponential Time Conjecture and the Projection Game Conjecture (this conjecture is defined later) the best ratio possible for a quasi-polynomial time is algorithm for DST is $\Theta(\log^2 n / \log \log n)$. This is a recent trend in designing approximation algorithms that allows ones to break through the hardness barrier with stronger assumptions. See, e.g., [1, 3, 17, 4, 5, 12].

The Exponential Time Conjecture together with the almost linear size PCP theorems [13, 27] yields many running time lower bounds for approximation algorithms [22, 3, 6]. Let us give an example of the results of this type:

Example: Consider the Max Clique problem, in which the goal is to find a clique of maximum size in a graph $G = (V, E)$ on $n$ vertices. This problem is known to admit no $n^{1-c}$-approximation, for any $c > 0$, unless $P = \text{NP}$ [21, 30]. Now, let us ask for an $\alpha$-approximation algorithm, for $\alpha$ ranging from constant to $\sqrt{n}$. There is a trivial $2^{\alpha/\log(n)}$-time approximation algorithm, which is obtained by partitioning vertices of $G$ into to $\alpha$ parts and finding a maximum clique from each part separately. Clearly, the maximum clique amongst these solutions is an $\alpha$-approximate solution, and the running time is $2^{\alpha/\log(n)} \cdot n$. The question is whether this is the best possible running-time. Chalermsook et al. [22] showed that such a trivial algorithm is almost tight\(^1\). To be precise, under the Exponential Time Conjecture, there is no $\alpha$-approximation algorithm that runs in time $2^{n^{1-\epsilon}/\alpha^{3+\epsilon}}$, for any constant $\epsilon > 0$, unless the Exponential Time Conjecture is false.

1.1 The problems studied in this paper

1.1.1 The Set Cover problem

In the weighted SET COVER problem, the input is a universe $U$ of size $n$ and a collection $\mathcal{S}$ of $m$ subsets of $U$. Each set $s \in \mathcal{S}$ has a cost $c(s)$. The goal is to select a minimum-cost sub-collection $\mathcal{S}' \subseteq \mathcal{S}$ such that the union of the sets in $\mathcal{S}'$ spans the entire universe $U$.

The Submodular Cover problem admits as input a universe $U$ with cost $c(x)$ on every $x \in U$. A function is submodular if for every $S \subseteq T \subseteq V$ and for every $x \in U \setminus T$, $f(S \cup x) - f(S) \geq f(T \cup x) - f(T)$. Let $f : 2^U \to R$ be a submodular non-decreasing function. The goal in the submodular cover problem is to minimize $c(S)$ subject to $f(S) = f(U)$.

The Connected Polymatroid problem is the case that the elements in $U$ are leaves of a tree, rooted at some vertex $r$, and and both the elements and tree edges have costs. The goal is to select a set $S$ so that $f(S) = f(U)$ and that $c(S) + c(T(S))$ is minimized, where $T(S)$ is the unique tree rooted at $r$ spanning $S$.

Given two problems $A$ and $B$, we say that $A \rightarrow B$, if problem $A$ is a special case of problem $B$. Given this notation, we get that SET COVER $\rightarrow$ Submodular Cover $\rightarrow$ Connected Polymatroid.

1.1.2 The Group Steiner tree problem

In the Group Steiner tree problem, the input consists of an undirected graph $G(V, E)$ with cost $c(e)$ on each edge $e \in E$, and a collection of subsets of $V$ $g_1, g_2, \ldots, g_k \subseteq V$ (called group) and a special vertex $r \in V$. The goal is is to find a minimum-cost tree rooted at $r$ that contains at least one vertex from every group $g_i$. The Covering Steiner tree problem, has the same input as the Group Steiner tree problem in addition of a demand $d_i$ for every $g_i$, and $d_i$ vertices of $g_i$ must be spanned in the tree rooted by $r$.

\(^1\)Recently, Bansal et al. [1] showed that Max Clique admits $\alpha$-approximation in time $2^{n/\tilde{O}(\alpha \log^2 \alpha \log \log n)}$.\footnote{Recently, Bansal et al. [1] showed that Max Clique admits $\alpha$-approximation in time $2^{n/\tilde{O}(\alpha \log^2 \alpha \log \log n)}$.}
1.1.3 The DST problem

In the DST problem, the input consists of a directed graph with costs $c(e)$ on edges, a collection $S \subseteq V$ of terminals and a designated root $r \in V$. The goal is to find a minimum-cost directed graph rooted at $r$ that spans $S$. The following relation between the problems holds: Set Cover $\rightarrow$ Group Steiner tree $\rightarrow$ Covering Steiner tree $\rightarrow$ DST.

In [16] it is shown that an $\alpha$ approximation for Group Steiner tree implies the same approximation for the Covering Steiner tree problem, using a simple reduction.

We are not aware of any proven relation between the Connected Polymatroid and the DST problems. In fact such a relation may not exist.

1.2 Our Results

Theorem 1.1. The time required to get a $(1 - \alpha) \ln n$ for Set Cover, with $n$ the number of elements, is $2^{O(n^\epsilon)}$. This matched the running time of [12] in their $(1 - \alpha) \ln n$ approximation for Set Cover (the approximation of [12] is also in the number of elements). This settles the time, up to constants in the exponent, and improves [26]

Another natural question is what is the best time to design an $(1 - \alpha) \ln n$ ratio for Set Cover, if $n$ is the number of sets plus the number of elements. Using an additional assumption, we get the same result.

Theorem 1.2. Under the Exponential Time Conjecture and Projection Game Conjecture, the time required to give ratio ratio $(1 - \alpha) \ln n$ for $n$ the number of sets plus the number of elements is $2^{\Theta(n^{\alpha})}$.

Theorem 1.3. For any $\alpha \geq 1/2$, the Directed Steiner Tree problem admits an $2^{O(\log n n^{\alpha})}$ time, $(1 - \alpha) \ln n$ approximation. For $\alpha < 1/2$ we explain that such a theorem is unlikely.

1.3 Related work

The uniform weights Set Cover admits $\ln n + 1$ ratio that is usually attributed to Johnson [24] and Lovsz -[25]. This result was known many years before 1974, and should be considered folklore. The Submodular Cover problem was also shown to admit $\ln n + 1$-approximation by Wolsey [29]. In [14] it is shown that unless $P = NP$, the Set Cover problem admits no $(1 - \alpha) \ln n$ approximation.

The Group Steiner tree problem was studied by Garg, Konjevod and Ravi [18], who presented an elegant LP rounding algorithm to approximate Group Steiner tree on trees to within a factor of $O(\log^2 n)$. Using the probabilistic metric-tree embedding [2, 28], this implies an $O(\log^2 n)$-approximation algorithm for Group Steiner tree in general graphs. On the negative side, Halperin and Krauthgamer [20] showed the lower bound of $\log^{2-\epsilon} n$ for any $\epsilon > 0$ for approximating Group Steiner tree on trees under the assumption that $NP \not\subseteq ZTIME(n^{\text{polylog}(n)})$. This (almost) matches the approximation by Garg et al. For the related problem, the Connected Polymatroid problem was given a polylogarithmic approximation algorithm by Calinescu and Zelikovsky [8]; their algorithm is based on the work of Chekuri, Even and Kortsarz [11], who gave a combinatorial $\log^{2+\epsilon} n$ approximation for Group Steiner tree on trees.

The best known polynomial time approximation ratio algorithm for the DST problem is $n^\epsilon$ for any constant $\epsilon > 0$ [9].
2 Formal definition of our two complexity assumptions

Definition 2.1. In the Label Cover problem with the projection property (a.k.a., the Projection game), we are given a bipartite graph $G(A, B, E)$, two alphabet sets (also called labels) $\Sigma_A$ and $\Sigma_B$, and for any edge (also called query) $e \in E$, there is a function $\phi_e : \Sigma_A \rightarrow \Sigma_B$. A labeling $(\sigma_A, \sigma_B)$ is a pair of functions $\sigma_A : A \rightarrow \Sigma_A$ and $\sigma_B : B \rightarrow \Sigma_B$ assigning labels to each vertices of $A$ and $B$, respectively. An edge $e = (a, b)$ is covered by $(\sigma_A, \sigma_B)$ if $\phi_e(\sigma_A(a)) = \sigma_B(b)$.

The goal in Label Cover is to find a labeling $(\sigma_A, \sigma_B)$ that covers as many edges as possible.

In the context of the Two-Provers One-Round game (2P1R), every label is an answer to some ”question” $a$ sent to the Player $A$ and some question $b$ sent to the Player $B$, for a query $(a, b) \in E$. The two answers make the verifier accept if a label $x \in \Sigma_A$ assigned to $a$ and a label $y \in \Sigma_B$ assigned to $b$ satisfy $\phi(x) = y$. Since any label $x \in \Sigma_A$ has a unique label in $\Sigma_B$ that causes the verifier to accept, $y$ is called the projection of $x$ into $b$.

The Exponential Time Conjecture, asserts that an instance of the 3-SAT problem on $n$ variables and $m$ clauses cannot be solved in $2^{o(n)}$-time. It was later showed by Impagliazzo, Paturi and Zane [23] that any 3-SAT instance can be sparsified in $2^{o(n)}$-time to an instance with $m = O(n)$ clauses. Thus, ETH together with the sparsification lemma [7] implies the following:

Exponential-Time Hypothesis combined with the Sparsification Lemma: Given a boolean 3-CNF formula $\phi$ on $n$ variables and $m$ clauses, there is no $2^{o(n+m)}$-time algorithm that decides whether $\phi$ is satisfiable. In particular, 3-SAT admits no subexponential-time algorithm.

The following was proven by Moshkovitz and Raz [27].

Theorem 2.1. [27] There exists $c > 0$, such that for every $\epsilon \geq 1/n^c$, 3-SAT on inputs of size $n$ can be efficiently reduced to Label Cover of size $N = n^{1+o(1)}\log(1/\epsilon)$ over an alphabet of size $\exp(1/\epsilon)$ that has perfect completeness and soundness error $\epsilon$. The graph is bi-regular (namely, every two questions on the same side participate in the same number of queries).

For our theorems, we set $\epsilon = 1/polylog\ n$. This implies that the degree of $A$ and of $B$ is bounded by $polylog\ n$.

There does not seem to be an inherent reason that the alphabet would be so large. This leads to the following conjecture posed by Moshkovitz [26].

Conjecture 2.2 (The Projection Games Conjecture [26]). There exists $c > 0$, such that for every $\epsilon \geq 1/n^c$, 3-SAT on inputs of size $n$ can be efficiently reduced to Label Cover of size $N = n^{1+o(1)}\log(1/\epsilon)$ over an alphabet of size $\log(1/\epsilon)$ that has soundness error $\epsilon$. Moreover, the graph is bi-regular (namely, every two questions on the same side participate in the same number of queries).

The difference between Theorem 2.1 and Conjecture 2.2 is in the size of the alphabet.

3 Approximating the Directed Steiner Tree and the Connected Polymatroid problems

Let $T^*$ denote the optimum DST tree and opt denote its cost. The set of terminals is denoted by $S$ and the root by $r$.

If a tree rooted at $r$ contains a terminal, we say that the tree covers the terminal. Given the DST problem we may assume that every terminals is a leaf, and every leaf is a terminal. For any terminal that is not a leaf attach a leaf terminal to this vertex. Non terminal leaves can be removed. Thus we denote the number of terminals (hence leaves) by $\ell$. The number of uncovered leaves is denoted by $\ell'$. 


The density of a tree $T$ is defined as $c(T)/\ell'(T)$, with $\ell'(T)$ the number of uncovered terminals that belong to $T$.

We now present a $(1 - \alpha) \cdot \ln n$-approximation algorithm for the DST and the Connected Polymatroid problems, running in time $2^{O(\log n \cdot n^\alpha)}$.

**Lemma 3.1.** For any rooted tree $T$ with $\ell' >$ leaves (terminals), there exists a set $X \subseteq V(T)$ of $O(n^\alpha)$ vertices together with a family of edge disjoint trees $T_1, \ldots, T_q$, such that:

- the trees are edge (but not vertex) disjoint
- each $T_i$ is a subtree of $T$,
- the root of each $T_i$ belongs to $X$,
- each leaf of $T$ is a leaf of exactly one $T_i$,
- each $T_i$ has more than $n^\alpha$ but less than $2n^\alpha$ leaves.
- One of the trees has density at most $\frac{\text{opt}}{\ell'}$ with $\ell'$ the number of uncovered leaves.

**Proof.** As long as the tree has more than $e \cdot n^\alpha$ leaves do the following: pick the lowest vertex $v$ in the tree, whose subtree $T_v$ has more than $n^\alpha$ leaves. This implies that all its children contain strictly less than $n^\alpha$ leaves, by removing leaves. Accumulation subtrees gives at most $2n^\alpha$ leaves since before the last iteration there were less that $n^\alpha$ leaves, and the last iteration adds a tree of at most $n^\alpha$ leaves. Remove the collected tree, but do not remove their root (namely this root may later participate in other trees). Note that after the accumulated trees are removed, the tree rooted by our chosen root may still have more than $e \cdot n^\alpha$ leaves. This gives at most $\Theta(n^{1-\alpha})$ edge disjoint trees with at least $n^\alpha$ leaves each. By an averaging argument, there is a tree with at least $n^\alpha$ leaves, and density at most $\frac{\text{opt}}{\ell'}$, with $\ell'$ the number of uncovered terminals. \qed

**Lemma 3.2.** We can find in time $2^{O(\log n \cdot n^\alpha)}$ a tree rooted at $r$ with $n^\alpha$ leaves and density at most $4 \cdot \frac{\text{opt}}{\ell'}$

**Proof.** For simplicity assume that each one of the trees $T_i$ above has exactly $n^\alpha$ leaves by removing leaves. This increases the density by at most a factor of 2. Say that $T_i$ is the best density tree. Thus its density is at most $2 \cdot \frac{\text{opt}}{\ell'}$. Then we take the transitive closure of the graph and then add the edge from $r$ to the root of $T_i$. Note that the cost of this edge is at most opt, since otherwise the root of $T_i$ can be discarded. This makes the cost of the tree $2 \cdot \text{opt}$ but makes sure the tree is rooted at $r$. We find the “correct” leaves, namely the $n^\alpha$ leaves of $T_i$ via exhaustive search in $\binom{n}{n^\alpha} = 2^{O(\log n \cdot n^\alpha)}$ time. Given the set of leaves $L_i$ of $T_i$, so that $|L_i| = n^\alpha$, we now guess the Steiner vertices $S_i$ of the tree. As the graph went via transitive closure, the number of Steiner vertices in the tree containing $r + X$ is no larger than $|X| = n^\alpha$ as well. This follows because the tree does not have vertices of indegree and outdegree 1. We can guess both the leaves of $T_i$ and its set of Steiner vertices $S_i$ in time $2^{O(\log n \cdot n^\alpha)}$, (given the $O()$ notation). Given “correct” Leaves $L_i$ of $T_i$ and the “correct” Steiner vertices $S_i$ of $T_i$, we use an algorithm of Dreyfuss and Wagner [15], that finds the minimum cost Directed tree rooted at $r$, with $L_i$ as leaves and $S_i$ as Steiner vertices. The algorithm of [15] runs in time $O(3^{n^\alpha})$ which is negligible in our case. By the above discussion the density of the tree found is at most $4 \cdot \frac{\text{opt}}{\ell'}$. \qed
4 The algorithm and analysis

The algorithm:

The algorithm iterates, using Lemma 3.2. Each time we find a tree with \( n^\alpha \) leaves, and density at most \( 4\text{opt}/\ell' \) until there are at most \( e \cdot n^\alpha \) leaves. Each tree requires \( 2^{O(n^\alpha \log n)} \) time to find. The \( O() \) times we run the algorithm, does not change the running time (given the \( O() \) notation in the exponent). Thus the total running time is \( 2^{O(log n \cdot n^\alpha)} \). Finally, when there are less than \( e \cdot n^\alpha \) unconnected terminals left, we find an optimum directed Steiner tree for those vertices, using [15].

Lemma 4.1. The approximation ratio of the above algorithm is at most \((1 - \alpha) \ln n\).

Proof. Recall that \( T^* \) is the optimum Steiner tree spanning \( S \), and recall that its cost is \( \text{opt} \). Since all trees are rooted by \( r \) and the density of any tree found is at most \( 4\text{opt}/\ell' \), where \( \ell' \) is the number of not yet connected terminals, and since we stop when the number of uncovered terminals is at most \( e \cdot n^\alpha \), by the standard set-cover type analysis we can bound the cost of the tree up to a constant 4 by:

\[
\sum_{i=n}^{e \cdot n^\alpha} \frac{\text{opt}}{i} = \text{opt} \cdot (H_n - H_{e \cdot n^\alpha}) \leq (1 - \alpha) \ln n \cdot \text{opt}.
\]

The constant \( e \) in the term takes care of the fact that \( H_n - \ln n < 1 \). Thus \( \ln n + 1 \geq H_n \). Thus \( H_n - H_{e \cdot n^\alpha} \leq \ln n + 1 - \alpha \ln n - 1 = (1 - \alpha) \ln n \).

Now we observe that the same theorem applies for the Connected Polymatroid problem. Since the function is both submodular and \textit{increasing} for every collection of pairwise disjoint sets \( \{S_i\}_{i=1}^k \), \( \sum_{i=1}^k f(S_i) \geq f(\bigcup_{i=1}^k S_i) \). Thus for a given \( \alpha \), at iteration \( i \) there exists a collection of leaves \( S_i \) so that \( f(S_i)/c(S_i) \geq f(U)/c(U) \). We can guess \( S_i \) in time \( \exp(n^\alpha \cdot \log n) \) and its set of Steiner vertices \( X_i \) in time \( O(3^{n^\alpha}) \). Using the algorithm of [15], we can find a tree of density at most \( 2 \cdot \text{opt}/n^\alpha \). The rest of the proof is identical.

5 Hardness for Group Steiner Tree under the Exponential Time Hypothesis

In this section, we show that the approximation hardness of the group Steiner problem under the Exponential Time Conjecture, which implies that the subexponential-time algorithm for Group Steiner tree of Chekuri and Pal [10] is nearly tight. The following corollary of Theorem [20].

Theorem 5.1 (Corollary of Theorem 1.1 in [20]). Unless the Exponential-Time-Hypothesis is false, for any parameter \( 0 < \delta < 1 \), there is no exp(\( 2^{\log(\delta - \epsilon) N} \)) time \( \log^{2-\delta-\epsilon} k \)-approximation algorithm for Group Steiner tree, for any \( 0 < \epsilon < \delta \).

Proof Sketch of Theorem 5.1. We provide here the parameter translation of the reduction in [20], which will prove Theorem 5.1.

The Reduction of Halperin and Krauthgamer. We shall briefly describe the reduction of Halperin and Krauthgamer. The starting point of their reduction is the Label Cover instance obtained from \( \ell \) rounds of parallel repetition. In the first step, given a \( d \)-regular Label Cover instance \( G = (G = (A, B, E), \Sigma_A, \Sigma_B, \phi) \) completeness 1 and soundness \( \gamma \), they apply \( \ell \) rounds of parallel repetition to get a \( d^\ell \)-regular instance of Label Cover \( G' = (G^\ell, A^\ell, B^\ell, E^\ell), \Sigma_A^\ell, \Sigma_B^\ell, \phi^\ell) \). To simplify the notation, we
Subexponential-Time Approximation-Hardness. Now we derive the subexponential-time approximation hardness for Group Steiner tree. We start by the nearly linear-size PCP theorem of Dinur [13], which gives a reduction from 3-SAT of size $n$ (the number of variables plus the number of clauses) to a label cover instance $G = (G = (A, B, E), \Sigma_A, \Sigma_B, \phi)$ with completeness 1, soundness $\gamma$ for some $0 < \gamma < 1$, $|A|, |B| \leq n \cdot \text{polylog}(n)$, degree $d = O(1)$ and alphabet sets $\Sigma_A, \Sigma_B$, $|\Sigma_A|, |\Sigma_B| = O(1)$.

For every parameter $0 < \delta < 1$, we choose $H = \log^{1/\delta-1} n$, which then forces us to choose $\ell = \Theta((1/\delta - 1) \log \log n)$. Note that we may assume that $\delta < n$ since it is a fixed parameter. Plugging in these parameter settings, we have an instance of Group Steiner tree on a tree with $N$ vertices and $k$ groups such that

$$N = \left(O(1) \cdot n^{1+o(1)}\right)^{\Theta((1/\delta-1) \log \log n) - \log^{1/\delta-1} n} = \exp\left(\log^{1/\delta+o(1)} n\right)$$

and

$$k = O(1)^{\Theta((1/\delta-1) \log \log n) - \log^{1/\delta} n} = \exp\left(\log^{1/\delta+o(1)} n\right)$$

Observe that $H \geq \log^{1-\delta-o(1)} k$. Thus, the hardness gap is $\Omega(H \log k) = \Omega(\log^{2-\delta-o(1)} k)$. This means that any algorithm for Group Steiner tree on this family of instances with approximation ratio $\log^{2-\delta-\epsilon} k$, for any constant $\epsilon > 0$, would be able to solve 3-SAT.

Now suppose there is an $\exp(2^{\log^{\delta-\epsilon} N})$-time $\log^{2-\delta-\epsilon} k$-approximation algorithm for Group Steiner tree, for some $0 < \epsilon < \delta$. We apply such an algorithm to solve an instance of Group Steiner tree derived from 3-SAT as above. Then we have an algorithm that runs in time

$$\exp(2^{\log^{\delta-\epsilon} N}) = \exp\left(2^{(\log^{1/\delta+o(1)} n)^{\delta-\epsilon}}\right) = \exp\left(2^{(\log^{(1+o(1))(\delta-\epsilon)} n)}\right) = \exp\left(2^{o(\log n)}\right) = 2^{o(n)}$$

This implies a subexponential-time algorithm for 3-SAT, which contradicts the Exponential Time Conjecture. Therefore, unless the Exponential Time Conjecture is false, there is no $\exp(2^{\log^{\delta-\epsilon} N})$-time $\log^{2-\delta-\epsilon} k$-approximation algorithm for Group Steiner tree, thus proving Theorem 5.1.

Since we take log from the expression, the above is also true if we replace $k$ by $N$. Combined with [10] and [16], we have the following corollary which shows almost tight running-time lower and upper bounds for approximating Group Steiner tree and Covering Steiner tree to a factor $\log^{2-\delta} N$.

**Corollary 5.2.** The Group Steiner tree and Covering Steiner tree problems on graphs with $n$ vertices admit $\log^{2-\delta} n$-approximation algorithms for any constant $\delta < 1$ that runs in time $\exp(2^{(1+o(1)) \log^{\delta} n})$. In addition, for any constant $\epsilon > 1$ there is no $\log^{2-\delta-\epsilon} n$ approximation algorithm for Group Steiner tree and Covering Steiner tree that runs in time $\exp(2^{(1+o(1)) \log^{\delta-\epsilon} n})$. 


References


