On subexponential running times for approximating directed Steiner tree and related problems

Marek Cygan* Guy Kortsarz† Bundit Laekhanukit‡

February 27, 2018

Abstract

This paper concerns proving almost tight (super-polynomial) running times, for achieving desired approximation ratios for various problems. To illustrate the question we study, let us consider the Set-Cover problem with n elements and m sets. Now we specify our goal to approximate Set-Cover to a factor of $(1-\alpha)\ln n$, for a given parameter $0 < \alpha < 1$. What is the best possible running time for achieving such approximation ratio? This question was answered implicitly in the work of Moshkovitz [Theory of Computing, 2015]: Assuming both the *Projection Games Conjecture* (PGC) and the *Exponential-Time Hypothesis* (ETH), any $((1-\alpha)\ln n)$-approximation algorithm for Set-Cover must run in time at least $2^{n^{c\cdot\alpha}}$, for some small constant $0 < c < 1$.

We study the questions along this line. Our first contribution is in strengthening the above result. We show that under ETH and PGC the running time requires for any $((1-\alpha)\ln n)$-approximation algorithm for Set-Cover is essentially $2^{n^{\alpha}}$. This (almost) settles the question since our lower bound matches the best known running time of $2^{O(n^{\alpha})}$ for approximating Set-Cover to within a factor $(1-\alpha)\ln n$ given by Cygan et al. [IPL, 2009]. Our result is tight up to the constant multiplying the n^{α} terms in the exponent.

The lower bound of Set-Cover applies to all of its generalization, e.g., Group-Steiner-Tree, Directed-Steiner-Tree, Covering-Steiner-Tree and Connected-Polymatroid. We show that, surprisingly, in almost exponential running time, these problems reduce to Set-Cover. Specifically, we complement our lower bound by presenting an $(1-\alpha)\ln n$ approximation algorithm for all aforementioned problems that runs in time $2^{n^{\alpha}\cdot\log n}\cdot\mathrm{poly}(m)$.

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 $(\log^{2-\alpha} n)$-approximation in time $\exp(2^{\log^{\alpha+o(1)} n})$, for any parameter $0 < \alpha < 1$. We show the running time lower bound of Group-Steiner-Tree: any $(\log^{2-\alpha} n)$-approximation algorithm for Group-Steiner-Tree must run in time at least $\exp((1+o(1))\log^{\alpha-\epsilon} n)$, for any constant $\epsilon > 0$, unless the ETH is false. Our result follows by analyzing the hardness construction of Group-Steiner-Tree due to the work of Halperin and Krauthgamer [STOC, 2003]. The same lower and upper bounds hold for Covering-Steiner-Tree.

*Department of Math and information, University of Warsaw, Warsaw, Porland. Email: `cygan@nimuw.edu.pl`

†Computer Science Department, Rutgers University – Camden, Camden NJ, USA. Email: `guyk@camden.rutgers.edu`

‡Max Planck Institute for Informatics, Saarbrücken, Germany & Institute for Theoretical Computer Science, Shanghai University of Finance and Economics, Shanghai, China. Email: `blaekhan@mpi-inf.mpg.de`

1 Introduction

The traditional study of *approximation algorithms* concerns designing *algorithms that run in polynomial time* while producing a solution whose cost is within a factor α away from the optimal solution. Once the approximation guarantees meet the barrier, a natural question is to ask whether the approximation ratio can be improved if the algorithms are given running time beyond polynomial. This has been a recent trend in designing approximation algorithms that allows ones to break through the hardness barrier; see, e.g., [1, 5, 22, 6, 6, 7, 7, 15, 14].

While ones ask for improving the approximation ratio, another interesting question is to ask the converse: Suppose the approximation ratio has been specified at the start, what is the smallest running time required to achieve such approximation ratio? This question has recently been an active subject of study; see, e.g., [9, 3, 4].

To answer the above question, ones need complexity assumptions stronger than P $\neq$ NP as this standard assumption does not precisely specify the running times besides polynomial versus super-polynomial. The most popular and widely believed assumption is the *Exponential-Time Hypothesis* (ETH), which states that 3-SAT admits no $2^{o(n)}$-time algorithm. This together with the *almost linear size PCP theorems* [16, 33] yields many running time lower bounds for approximation algorithms [9, 3, 4]. Let us give an example of the results of this type:

Example: Consider the Maximum 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 $n^{1-\epsilon}$-approximation, for any $\epsilon > 0$, unless P = NP [26, 35]. Now, let us ask for an α-approximation algorithm, for α ranging from constant to $\sqrt{n}$. There is a trivial $2^{n/\alpha}\text{poly}(n)$-time approximation algorithm, which is obtained by partitioning vertices of G into to α parts and finding a maximum clique from each part separately. Clearly, the maximum clique amongst these solutions is an α-approximate solution, and the running time is $2^{n/\alpha}\text{poly}(n)$. The question is whether this is the best possible running-time. Chalermsook et al. [9] showed that such a trivial algorithm is almost tight[1]. To be precise, under the ETH, there is no α-approximation algorithm that runs in time $2^{n^{1-\epsilon}/\alpha^{1+\epsilon}}$, for any constant $\epsilon > 0$, unless the ETH is false.

In this paper, we consider the question along this line. We wish to show the tight lower and upper bounds on the running times of polylogarithmic approximation algorithms for Set-Cover, Group-Steiner-Tree and Directed-Steiner-Tree (which we will define in the next section) and related problems.

- For any constant $0 < \alpha < 1$, what is the best possible running times for $(1-\alpha)$-approximation algorithms for Set-Cover and Directed-Steiner-Tree.
- For any constant $0 < \alpha < 1$, what is the best possible running time for $\log^{2-\alpha}$-approximation algorithms for Group-Steiner-Tree.

In fact, one of our ultimate goals is to find an evidence on which ranges of running-times that the Directed-Steiner-Tree problem admits poly-logarithmic approximations. To be precise, we would like to partially answer the question of whether Directed-Steiner-Tree admits polylogarithmic approximations in polynomial-time, which is a big open problem in the area. While we are far from solving the above question, we aim to prove possibly tight running-time for Directed-Steiner-Tree in the logarithmic range in a very fine-grained manner, albeit assuming two strong assumptions, the *Exponential-Time Hypothesis* (ETH) [27, 28] and the *Projection Game Conjectures* (PGC) [32], simultaneously.

1.1 The problems studied in this paper

1.1.1 The Set-Cover problem and its extensions

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 subcollection $\mathcal{S}' \subseteq \mathcal{S}$ such that the union of the sets in $\mathcal{S}'$ spans the entire universe U.

[1] Recently, Bansal et al. [1] showed that Maximum Clique admits α-approximation in time $2^{n/\tilde{O}(\alpha \log^2 \alpha)}\text{poly}(n)$.

The more general 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 $x \in U \setminus T$, $f(S+x) - f(S) \geq f(T+x) - f(T)$. Let $f : 2^U \mapsto R$ be a submodular non-decreasing function. The goal in the submodular cover problem is to minimize $c(S)$ subject to $f(S) = f(U)$. This problem strictly generalizes the weighted SET-COVER problem.

The CONNECTED-POLYMATROID problem is the case that the elements in U are leaves of a tree, 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.

1.1.2 The Group-Steiner-Tree problem

In the GROUP-STEINER-TREE problem, the input consists an undirected graph with cost $c(e)$ on each edge $e \in E$, a collection of subsets $g_1, g_2, \ldots, g_k \subseteq V$ (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. In the COVERING-STEINER-TREE problem, there is a demand d_i for every g_i and d_i vertices of g_i must be spanned in the tree rooted by r This GROUP-STEINER-TREE problem strictly contains the SET-COVER problem. Every result for GROUP-STEINER-TREE holds also for the COVERING-STEINER-TREE problem given that there is a reduction from COVERING-STEINER-TREE to GROUP-STEINER-TREE [19, 24].

1.1.3 The Directed-Steiner-Tree problem

In the DIRECTED-STEINER-TREE problem, the input consists of a directed graph with costs $c(e)$ on edges, a collection S 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. This problem has GROUP-STEINER-TREE as a special case.

1.2 Related work

The SET-COVER problem is a well-studied problem. The first logarithmic approximation, to the best of our knowledge, is traced back to the early work of Johnson [29]. Many different approaches have been proposed to approximate SET-COVER, e.g., the dual-fitting algorithm by Chvátal [13]; however, all algorithms yield roughly the same approximation ratio. The more general problem, namely, the SUBMODULAR-COVER problem was also shown to admit $O(\log n)$-approximation in the work of Wolsey [34]. The question of why all these algorithms yield the same approximation ratio was answered by Lund and Yannakakis [31] who showed that the approximation ratio $\Theta(\log n)$ is essentially the best possible unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\log\log n})$. Subsequently, Feige [21] showed the more precise lower bound that SET-COVER admits no $(1-\epsilon)$-approximation, for any $\epsilon > 0$, unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\mathrm{polylog}(n)})$; this assumption has been weaken to $\mathrm{P} \neq \mathrm{NP}$ by the recent work of Dinur and Steurer [17]. These lower bounds are, however, restricted to polynomial-time algorithms. In the regime of subexponential-time, Cygan, Kowalik and Wykurz [15] showed that SET-COVER admits an approximation ratio of $(1-\alpha)\ln n$ in $2^{O(n^{\alpha}+\mathrm{poly}(\log n))}$ time. On the negative side, Moshkovitz [32] introduced the Projection Games conjecture (PGC) to prove the approximation hardness of SET-COVER. Originally, the conjecture was introduced in an attempt to show the $(1-\epsilon)\log n$-hardness of SET-COVER under $\mathrm{P} \neq \mathrm{NP}$ (which is now proved by Dinur and Steurer [17]). It turns out that this implicitly implies that SET-COVER admits no $(1-\alpha)$-approximation algorithm in $2^{n^{O(\alpha)}}$ time under PGC and ETH.

The generalization of the SET-COVER problem is the GROUP-STEINER-TREE problem. Garg, Konjevod and Ravi [23] presented a novel 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, 20], this implies an $O(\log^3 n)$-approximation algorithm for GROUP-STEINER-TREE in general graphs. On the negative side, Halperin and Krauthgamer showed the lower bound of $\log^{2-\epsilon} n$ for any $\epsilon > 0$ for approximating GROUP-STEINER-TREE on trees under the assumption that $\mathrm{NP} \not\subseteq \mathrm{ZPTIME}(n^{\mathrm{polylog}(n)})$. This (almost) matches the upper bound given by the algorithm by Garg et al. For the related problem, the *Connected Polymatroid* problem was given a polylogarithmic approximation algorithm by Cálinescu and Zelikovsky [8]; their algorithm is based on the work of Chekuri, Even and Kortsarz [11], which gave a *combinatorial* polylog(n) approximation for Group-Steiner-Tree on trees.

The problem that generalizes all the above problems is the Directed-Steiner-Tree problem. The best known approximation ratio for this problem is n^ϵ for any constant $\epsilon > 0$ [10, 30] in polynomial-time. In quasi-polynomial-time, Directed-Steiner-Tree admits an $O(\log^3 n)$-approximation algorithm. The question of whether Directed-Steiner-Tree admits a polylogarithmic approximation in polynomial-time has been a long standing open problem.

2 Our results

We show that under the combination of ETH and PGC, the running time for approximating Set-Cover to within a factor of $(1-\alpha)\ln n$ must be at least 2^{n^α}, where $0 < \alpha < 1$ is a given parameter. This improves the work of Moshkovitz who (implicitly) showed the running time lower bound of $2^{n^{O(\alpha)}}$. We complement this by showing that Directed-Steiner-Tree admits a $(1-\alpha)\ln n$ approximation algorithm that runs in time $2^{n^\alpha \cdot \log n}$ time. Since Directed-Steiner-Tree is the generalization of Set-Cover, Group-Steiner-Tree and Directed-Steiner-Tree, the lower bounds apply to all the aforementioned problems. Hence, up to a small factor of $\log n$ in the exponent, we get tight running time lower bounds for approximating all these problems to within $(1-\alpha)\ln n$. Essentially, the same algorithm and proof gives the same result for the Connected-Polymatroid problem.

We also investigate the work of Chekuri and Pal [12] who showed that, for any constant $0 < \delta < 1$, Group-Steiner-Tree admits a $\log^{2-\delta} n$ approximation algorithm that runs in time $\exp(2^{(1+o(1))\log^\delta n})$. We show that, for any constant $\epsilon > 1$, there is no $\log^{2-\delta-\epsilon} n$ approximation algorithm for Group-Steiner-Tree (and thus Covering-Steiner-Tree) that runs in time $\exp(2^{(1+o(1))\log^{\delta-\epsilon} n})$. This lower bound is nearly tight. We note that a reduction from Covering-Steiner-Tree to Group-Steiner-Tree was given in [19]. Thus, any approximation algorithm for Group-Steiner-Tree also applies for Covering-Steiner-Tree.

3 Formal definition of our two complexity assumptions

Definition 3.1. *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*) Σ_A and Σ_B, and for any edge (also called* query*) $e \in E$, there is a function $\phi_e : \Sigma_A \mapsto \Sigma_B$. A labeling (σ_A, σ_B) is a pair of functions $\sigma_A : A \mapsto \Sigma_A$ and $\sigma_B : B \mapsto \Sigma_B$ assigning labels to each vertices of A and B, respectively. An edge $e = (a, b)$ is* covered *by (σ_A, σ_B) if $\phi_e(\sigma_A(a)) = \sigma_B(b)$. The goal in* Label-Cover *is to find a labeling (σ_A, σ_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 $x \in \Sigma_x$ 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 Σ_B that causes the verifier to accept, y is called the *projection of x into b.*

We use two conjectures in our paper. The first is the *Exponential Time Hypothesis* (ETH). Consider the 3-SAT problem with n literals and m clauses. Impagliazzo, Paturi and Zane [27] stated the hypothesis which together with the sparsification lemma [28] by Calabro, Impagliazzo and Paturi implies the following:

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

The following was proven by Moshkovitz and Raz [33].

Theorem 3.2 ([33])**.** *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 $N = n^{1+o(1)}\mathrm{poly}(1/\epsilon)$ over an alphabet of size* $\exp(1/\epsilon)$ *that has* *soundness error ϵ. The graph is bi-regular (namely, every two questions on the same side participate in the same number of queries).*

There does not seem to be an *inherent* reason that the alphabet would be so large. This lead to the following conjecture posed by Moshkovitz [32].

Conjecture 3.3 (The Projection Games Conjecture [32])**.** *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 $N = n^{1+o(1)}\text{poly}(1/\epsilon)$ over an alphabet of size* $\text{poly}(1/\epsilon)$ *that has soundness error ϵ. 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 3.2 and Conjecture 3.3 is in the size of the alphabet.

For our purposes, we only need soundness $\epsilon = 1/\text{polylog}(n)$, and we know that the degree and alphabet size of the graph in Conjecture 3.3 are always $\text{polylog}(n)$ (which are inverse of the soundness). Hence, we may assume the (slightly) weaker assumption (obtained by setting $\epsilon = 1/\text{polylog}(n)$ in Conjecture 3.3) as below.

Conjecture 3.4 (Projection Games Conjecture, a variant)**.** *There exists $c > 0$, such that for every $\epsilon = 1/\text{polylog}(n)$,* 3-SAT *on inputs of size n can be efficiently reduced to* Label-Cover *of size $N = n^{1+o(1)}\text{poly}(1/\epsilon)$ where* **the graph is bi-regular and all degrees are bounded by** $\text{polylog}(n)$*. The size of the alphabet is* $\text{polylog}(n)$ *and the soundness is* $1/\text{polylog}(n)$*. and the completeness is* 1.

We need to inspect very carefully and slightly change the proof of [32] since we do not want the Label-Cover instance to grow by a lot by the modification in [32]. Hence, in fact we have to go over all steps of [32] and bound the size more carefully in all steps that require that.

4 First part of the proof

We start with the same definition as in [32].

Definition 4.1 (Total disagreement)**.** *Let $(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$ be a* Label-Cover *instance. Let $\phi_A : A \to \Sigma_A$ be an assignment to the A-vertices. We say that the A-vertices* totally disagree *on a vertex $b \in B$, if there are no two neighbors $a_1, a_2 \in A$ of b, for which*

$$\pi_{e_1}(\phi_A(a_1)) = \pi_{e_2}(\phi_A(a_2)),$$

where $e_1 = (a_1, b), e_2 = (a_2, b) \in E$.

The above simply states that for a given assignment ϕ_A and a vertex $b \in B$, no matter which label we assign to the vertex b, we will satisfy only one edge incident to it.

Definition 4.2 (Agreement soundness)**.** *Let $\mathcal{G} = (G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$ be a* Label-Cover *for deciding whether a Boolean formula ϕ is satisfiable. We say that $\mathcal{G}$ has* agreement soundness error *ϵ, if for unsatisfiable ϕ, for any assignment $\phi_A : A \to \Sigma_A$, the A-vertices are in total disagreement on at least $1 - \epsilon$ fraction of the $b \in B$.*

For a Yes-Instance (of 3-SAT), a standard argument implies that you can label the vertices so that every edge is covered. The usual condition of soundness required is that the number of edges covered is a small fraction of the edges, for every label assignment. The total disagreement is stronger than that. It states that for any assignment ϕ_A, no matter how we set ϕ_B almost all of vertices of B will have at most one incident edge satisfied.

In the rest of this subsection the goal is to show (list) agreement soundness error of bounded degree Label-Cover instances. First, we use the following lemma (we do not alter its proof).

Lemma 4.3 (Combinatorial construction). *For $0 < \epsilon < 1$, for a prime power D, and Δ that is a power of D, there is an explicit construction of a regular graph $H = (U, V, E)$ with $|U| = n$, V-degree D, and $V \leq n^{O(1)}$ that satisfies the following. For every partition $U_1, \ldots, U_\ell$ of U into sets such that $|U_i| \leq \epsilon|U|$, for $i = 1, \ldots, \ell$, the fraction of vertices $v \in V$ with more than one neighbor in any single set U_i, is at most ϵD^2.*

It is rather trivial to show the above lemma by a probabilistic method. Moshkovitz showed in [32] that such graphs can be constructed deterministically via a simple and elegant construction.

In the next lemma, we show how to take a Label-Cover instance with standard soundness and convert it to a Label-Cover instance with total disagreement soundness, by combining it with the graph from Lemma 4.3. Here (as opposed to [32]) we have to bound the size of the created instance more carefully ([32] only states that the size is raised to a constant power).

Lemma 4.4. *Let $D \geq 2$ be a prime power and let Δ be a power of D. Let $\epsilon > 0$. From a* Label-Cover *instance with soundness error $\epsilon^2 D^2$ and B-degree n, we can construct a* Label-Cover *instance with agreement soundness error $2\epsilon D^2$ and B-degree D. The transformation preserves the alphabets, and the size of the created instance is increased by a factor* $\mathrm{poly}(\Delta)$, *namely by polynomial in the original B-degree.*

Proof. Let $\mathcal{G} = (G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$ be the original Label-Cover from the Projection Game Conjecture. Let $H = (U, V, E_H)$ be the graph from Lemma 4.3, where Δ, D and ϵ are as given in the current lemma. Let us use U to enumerate the neighbors of a B-vertex, i.e., there is a function $E^{\leftarrow} : B \times U \to A$ that, given a vertex $b \in B$ and $u \in U$, gives us the A-vertex which is the u neighbor of b.

We create a new Label-Cover $(G = (A, B \times V, E'), \Sigma_A, \Sigma_B, \Phi')$. The intended assignment to every vertex $a \in A$ is the same as its assignment in the original instance. The intended assignment to a vertex $\langle b, v\rangle \in B \times V$ is the same as the assignment to b in the original game. We put an edge $e' = (a, \langle b, v\rangle)$ if there exist $u \in U$ such that $E^{\leftarrow}(b, u) = a$ and $(u, v) \in E_H$. We define $\pi_{e'} = \pi_{(a,b)}$.

If there is an assignment to the original instance that satisfies c fraction of its edges, then the corresponding assignment to the new instance satisfies c fraction of its edges (this follows from the regularity of the graph H).

Suppose there is an assignment for the new instance $\phi_A : A \to \Sigma_A$ in which more than $2\epsilon D^2$ fraction of the vertices in $B \times V$ do not have total disagreement.

Let us say that $b \in B$ is good if for more than an ϵD^2 fraction of the vertices in $\{b\} \times V$ the A-vertices do not totally disagree. Note that the fraction of good $b \in B$ is at least ϵD^2.

Focus on a good $b \in B$. Consider the partition of U into $|\Sigma_B|$ sets, where the set corresponding to $\sigma \in \Sigma_B$ is:

$$U_\sigma = \{u \in U | a = E^{\leftarrow}(b, u) \wedge e = (a, b) \wedge \pi_e(\phi_A(a)) = \sigma\} .$$

By the goodness of b and the property of H, there must be $\sigma \in \Sigma_B$ such that $|U_\sigma| > \epsilon|U|$. We call σ the champion for b.

We define an assignment $\phi_B : B \to \Sigma_B$ that assigns good vertices b their *champions*, and other vertices b arbitrary values. The fraction of edges that ϕ_A, ϕ_B satisfy in the original instance is at least $\epsilon^2 D^2$.

The new instance is bigger by a factor $|V|$, which is $\mathrm{poly}(\Delta)$. ☐

Next we consider a variant of Label-Cover that is relevant for the reduction to Set-Cover. In this variant, the prover is allowed to assign each vertex ℓ values, and an agreement is interpreted as agreement on one of the assignments in the list.

Definition 4.5 (List total disagreement [32]). *Let $(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$ be a* Label-Cover *Let $\ell \geq 1$. Let $\hat{\phi}_A : A \to \binom{\Sigma_A}{\ell}$ be an assignment that assigns each A-vertex ℓ alphabet symbols. We say that the A-vertices* totally disagree *on a vertex $b \in B$ if there are no two neighbors $a_1, a_2 \in A$ of b, for which there exist $\sigma_1 \in \hat{\phi}_A(a_1), \sigma_2 \in \hat{\phi}_A(a_2)$ such that*

$$\pi_{e_1}(\sigma_1) = \pi_{e_2}(\sigma_2) ,$$

where $e_1 = (a_1, b), e_2 = (a_2, b) \in E$.

Definition 4.6 (List agreement soundness [32]). *Let $(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$ be a* Label-Cover *for deciding membership whether a Boolean formula ϕ is satisfiable. We say that G has* list-agreement soundness error (ℓ, ϵ), *if for unsatisfiable ϕ, for any assignment $\hat{\phi}_A : A \to \binom{\Sigma_A}{\ell}$, the A-vertices are in total disagreement on at least $1 - \epsilon$ fraction of the $b \in B$.*

If a PCP has low error ϵ, then even when the prover is allowed to assign each A-vertex ℓ values, the game is still sound. This is argued in the next corollary.

Lemma 4.7 (Lemma 4.7 of [32]). *Let $\ell \geq 1$, $0 < \epsilon' < 1$. Any instance of* Label-Cover *with agreement soundness error ϵ' has list-agreement soundness error $(\ell, \epsilon'\ell^2)$.*

The following corollary summarizes this subsection.

Corollary 4.8. *For any $\ell = \ell(n) = \text{polylog}(n)$, for any constant prime power D and constant $0 < \alpha < 1$, 3-*SAT *on input of size n can be reduced to a* Label-Cover *instance of size $N = n^{1+o(1)}$ with alphabet size $\text{polylog}(n)$, where the B-degree is D, and the list-agreement soundness error is (ℓ, α).*

Proof. Our starting point is the Label-Cover instance from 3.4 with soundness error ϵ, such that $2\sqrt{\epsilon} \cdot l^2 \leq \alpha$. Note that the B-degree of the instance is $\Delta = \text{polylog}(n)$. The corollary then follows by invoking Lemma 4.4 and Lemma 4.7. ☐

4.1 From Label-Cover to Set-Cover

Lemma 4.9 (Partition System [32]). *For natural numbers m, D, and $0 < \alpha < 1$, for all $u \geq (D^{\mathcal{O}(\log D)} \log m)^{1/\alpha}$, there is an explicit construction of a universe U of size u and partitions $\mathcal{P}_1, \ldots, \mathcal{P}_m$ of U into D sets that satisfy the following: there is no cover of U with $\ell = D \ln |U|(1 - \alpha)$ sets $S_{i_1}, \ldots, S_{i_\ell}$, $1 \leq i_1 < \ldots < i_\ell \leq m$, such that each set S_{i_j} belongs to the partition P_{i_j}.*

We will use the contrapositive of the lemma: if U has a cover of size at most ℓ, then this cover must contain at least two sets from the same partition. Next follows the reduction, which is almost the same as in [32], where the only difference is the parameter setting.

We take a Label-Cover instance $\mathcal{G}$ from Corollary 4.8 and transform it into an instance of Set-Cover. In order to do so, we invoke Lemma 4.9 with $m = |\Sigma_B|$ and D which is the B-degree of $\mathcal{G}$. The parameter u will be determined later. Let U be the universe, and $\mathcal{P}_{\sigma_1}, \ldots, \mathcal{P}_{\sigma_m}$ be the partitions of U, where the partitions are indexed by symbols of Σ_B. The elements of the Set-Cover instance are $B \times U$, i.e., for each vertex $b \in B$ there is a copy of U. Covering $\{b\} \times U$ corresponds to satisfying the edges that touch b. There are m ways to satisfy the edges that touch b – one for every possible assignment $\sigma \in \Sigma_B$ to b. The different partitions covering U correspond to those different assignments.

For every vertex $a \in A$ and an assignment $\sigma \in \Sigma_A$ to a, we have a set $S_{a,\sigma}$ in the Set-Cover instance. Taking $S_{a,\sigma}$ to the cover would correspond to assigning σ to a. Notice that a cover might consist of several sets of the form $S_{a,\cdot}$ for the same $a \in A$, which is the reason we consider list agreement. The set $S_{a,\sigma}$ is a union of subsets, one for every edge $e = (a, b)$ touching a. Suppose e is the i-th edge coming into b $(1 \leq i \leq D)$, then the subset associated with e is $\{b\} \times S$, where S is the i-th subset of the partition $\mathcal{P}_{\Phi_e(\sigma)}$.

If we have an assignment to the A-vertices such that all of the neighbors of b agree on one value for b, then the D subsets corresponding to those neighbors and their assignments form a partition that covers bs universe. On the other hand, if one uses only sets that correspond to totally disagreeing assignments to the neighbors, then by the definition of the partitions, covering U requires $\approx \ln |U|$ times more sets. The formal claim proved by Moshkovitz is as follows.

Claim 4.10 (Claim 4.10 of [32]). *The following holds*

- *Completeness: If all the edges in $\mathcal{G}$ can be satisfied, then the created instance admits a set cover of size $|A|$.*
- *Soundness: Let $\ell := |D| \ln |U|(1 - \alpha)$ be as in Lemma 4.9. If $\mathcal{G}$ has agreement soundness (ℓ, α), then every set cover of the created instance is of size more than $|A| \ln |U|(1 - 2\alpha)$.*

The following is our main theorem, where we fine-tune the parameters to get the best possible (and thus almost tight) running time lower bound.

Theorem 4.11. *Fix a constant $\gamma > 0$ and $\epsilon > 0$. Assuming* PGC *there is an algorithm that given an instance ϕ of* 3-SAT *of size n one can create an instance I of* Set-Cover *with universe of size $n^{1+o(1)} \cdot u$ such that if ϕ is satisfiable, then I has a set cover of size x, while if ϕ is not satisfiable, then I does not admit a set cover of size at most $x \ln |u|(1-\epsilon)$.*

Proof. Given a sparsified 3-CNF formula ϕ of size n we transform it into a Label-Cover instance $\mathcal{G}$, by Corollary 4.8, obtaining a list-agreement soundness error (ℓ, α), where we set $\alpha = \epsilon/2$ and $\ell = |D| \ln |U|(1-\alpha)$. Next, we perform the reduction from this section and by Claim 4.10 we have the following:

- If ϕ is satisfiable, then there exists a solution of size $|A|$ (where A is one side of $\mathcal{G}$).
- If ϕ is not satisfiable, then any set cover has size more than $|A| \ln |U|(1-2\alpha) = |A| \ln |U|(1-\epsilon)$.

□

By setting the value of $|U| = u$ appropriately we get a tradeoff between the approximation ratio and running time in the following lower bound obtained directly from Theorem 4.11.

Corollary 4.12. *Unless the* ETH *fails, for any $0 < \alpha < 1$ and $\epsilon > 0$ there is no $(1-\alpha)\ln n$ approximation for* Set-Cover *with universe of size n and m sets in time $2^{n^{\alpha-\epsilon}}\mathrm{poly}(m)$.*

Proof. Set $u = |\phi|^{1/\alpha - 1}$, then the created instance has at most $|\phi|^{1/\alpha + o(1)}$ elements, which fits the desired running time in the lower bound. It remains to analyze the approximation ratio. Note that $|A| \le u^{\alpha/(1-\alpha)}$, hence

$$(1-\alpha)\ln(|A| \cdot u) \le (1-\alpha)(\alpha/(1-\alpha)+1)\ln u = \ln u\,.$$

□

5 Approximating Directed Steiner Tree

In this section, we present a $(1-\epsilon) \cdot \ln n$-approximation for Directed-Steiner-Tree running in time $2^{\mathcal{O}(n^\alpha \log n)}$.

Lemma 5.1. *For any rooted tree T with ℓ leaves, there exists a set $X \subseteq V(T)$ of $\mathcal{O}(n^\alpha)$ 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^α but less than $2n^\alpha$ leaves.*

Proof. As long as the tree has more than n^α leaves do the following: pick the lowest vertex v in the tree, the subtree rooted at which has more than n^α leaves. This implies that all its children contain strictly less than n^α leaves. Accumulation subtrees gives at most $2n^\alpha$ leaves since before the last iteration there were less that n^α leaves, and the last iteration adds a tree of at most n^α 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 n^α leaves. This gives $\Theta(n^\alpha)$ edge disjoint trees with $\Theta(n^\alpha)$ leaves each. Thus, there is a tree with $\Theta(n^\alpha)$ leaves, and density (cost over the number of leaves) no larger than the optimum density. □

For simplicity, we make sure that the number of leaves in each tree is exactly n^α by discarding leaves. Since the trees are edge disjoint there must be a tree whose density: cost over the number of leaves is not worse (up to a factor of 2) than the optimum density $\textsc{opt}/\ell$.

Let (G, K, r) be an instance of Directed-Steiner-Tree. Our algorithm enumerates guesses the (roughly) n^α leaves L' in the tree whose density is no worse than the optimal density, and also guesses the subset $X_{L'} \subseteq V(G)$ that behaves as Steiner vertices. Assuming the graph went via a transitive closure, the size of $X_{L'}$ is at most the size of L' of size $\mathcal{O}(n^\alpha)$. For a fixed set X, the algorithm first finds an optimum directed Steiner tree T_0. We note that assuming that the graph went via transitive closure, we may assume that the number of Steiner vertices is less than the number of leaves, and so we may guess the Steiner vertices at time $n^{\sqrt{n}}$ as well. It is known that given the Steiner vertices and the leaves of the tree, we can, in polynomial time, find the best density tree with these leaves and these X vertices. The first such algorithm is due to Dreyfus and Wagner [18]. The algorithm is quite non trivial and that uses dynamic programming. The running time of the algorithm is $O(3^n)$ time which is negligible in our context.

We iterate adding more trees in this way. Each time we find the best density tree rooted at some vertex of X which covers n^α leaves, and add the edges to S. Each time it requires $2^{\mathcal{O}(n^\alpha \log n)}$ time. Finally, when there are less than $(e^2+1)n^\alpha$ unconnected terminals left, we find an optimum directed Steiner tree for those vertices.

Lemma 5.2. *The approximation ratio of the above algorithm is at most $(1-\alpha)\ln n$.*

Proof. Let T be an optimum Steiner tree spanning K, let $\textsc{opt}$ be its cost and let X be the set from Lemma 5.1 for the tree T. Let us analyze the algorithm in the iteration when it chooses the set X properly, that is picks the same set as Lemma 5.1. Note that since vertices of X belong to T the cost of the first tree T_0 found by the algorithm is at most $\textsc{opt}$. The last property of Lemma 5.1 guarantees, that our algorithm always finds a tree with at least as good density as $\textsc{opt}/r$, where r is the number of not yet connected terminals. By the standard set-cover type analysis we can bound the cost of all the best density trees found by the algorithm by

$$\sum_{i=n}^{e^2\cdot 2^{1-\alpha}} \frac{\text{OPT}}{i} = \textsc{opt} \cdot (H_n - H_{e^2\cdot 2^{1-\alpha}}) \tilde{=} (1-\alpha)\ln n \cdot \textsc{opt}.$$

Finally, the last tree is of cost at most $\textsc{opt}$, and it can be found in time $2^{\mathcal{O}(n^\alpha \log n)}$. ☐

Now we observe that the same theorem applies for the Connected-Polymatroid problem. Since the function is both submodular and *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 α, 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 [18], we can find a tree of density at most $2 \cdot \textsc{opt}/n^\alpha$. The rest of the proof is identical.

6 Hardness for Group Steiner Tree under the ETH

In this section, we show that the approximation hardness of the group Steiner problem under the ETH, which implies that the subexponential-time algorithm for Group-Steiner-Tree of Chekuri and Pal [12] is nearly tight. This hardness result is implicitly in the work Halperin and Krauthgamer [25]. More precisely, the following is a corollary of Theorem 1.1 in [25].

Theorem 6.1 (Corollary of Theorem 1.1 in [25])**.** *Unless the* ETH *is false, for any parameter* $0 < \delta < 1$*, there is no* $\exp(2^{\log^{\delta-\varepsilon} N})$*-time* $\log^{2-\delta-\varepsilon} k$*-approximation algorithm for* Group-Steiner-Tree*, for any* $0 < \varepsilon < \delta$*.*

Proof Sketch of Theorem 6.1. We provide here the parameter translation of the reduction in [25], which will prove Theorem 6.1.

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 ℓ rounds of parallel repetition. In the first step, given a d-regular LABEL-COVER instance $\mathcal{G} = (G = (A, B, E), \Sigma_A, \Sigma_B, \phi)$ completeness 1 and soundness γ, they apply ℓ rounds of parallel repetition to get a d^ℓ-regular instance of LABEL-COVER $\mathcal{G}' = (G^\ell = (A^\ell, B^\ell, E'), \Sigma_A^\ell, \Sigma_B^\ell, \phi')$. To simplify the notation, we let $m = |A| = |B|$, $\sigma = |\Sigma_A^\ell| = |\Sigma_B^\ell|$ be the number of vertices and the alphabet size of the LABEL-COVER instance $\mathcal{G}$, respectively. Then we have that the number of vertices and the alphabet size of $\mathcal{G}'$ is $2m^\ell$ and σ^ℓ, respectively. In the second step, they apply a recursive composition to produce an instance I of GROUP-STEINER-TREE on a complete $(2 \cdot m^\ell \cdot \sigma^\ell)$-ary tree $\widehat{T}$ of height H on $k = d^\ell m^{\ell \cdot H}$ groups. Moreover, if $\mathcal{G}$ is a Yes-Instance (i.e., there is a labeling that covers all the constraints), then there is a feasible solution to GROUP-STEINER-TREE with cost H with high probability, and if $\mathcal{G}$ is a No-Instance (i.e., every labeing covers at most γ fraction of the edges), then there is no solution with cost less than $\beta H^2 \log k$, for some sufficiently small constant $\beta > 0$.

In short, the above reduction gives an instance of GROUP-STEINER-TREE on a tree with $N = O((\sigma m)^{\ell H})$ vertices, $k = O(d^\ell m^{\ell H})$ groups, and with approximation hardness gap $\Omega(H \log k)$. Additionally, the reduction in [25] requires $\ell > c_0(\log H + \log\log m + \log\log d)$ for some sufficiently large constant $c_0 > 0$. □

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 [16], which gives a reduction from 3-SAT of size n (the number of variables plus the number of clauses) to a label cover instance $\mathcal{G} = (G = (A, B, E), \Sigma_A, \Sigma_B, \phi)$ with completeness 1, soundness γ for some $0 < \gamma < 1$, $|A|, |B| \le n \cdot \mathrm{polylog}(n)$, degree $d = O(1)$ and alphabet sets $\Sigma_A|, |\Sigma_B| = O(1)$.

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 $\delta \ll n$ 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

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

and

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

Observe that $H \ge \log^{1-\delta-o(1)} k$. 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 $\log^{2-\delta-\varepsilon} k$, for any constant $\varepsilon > 0$, would be able to solve 3-SAT.

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

$$\exp(2^{\log^{\delta-\varepsilon} N}) = \exp(2^{(\log^{1/\delta+o(1)} n)^{\delta-\varepsilon}}) = \exp(2^{(\log^{\frac{(1+o(1))(\delta-\varepsilon)}{\delta}} n)}) = \exp(2^{o(\log n)}) = 2^{o(n)}$$

This implies a subexponential-time algorithm for 3-SAT, which contradicts the ETH. Therefore, unless the ETH is false, there is no $\exp(2^{\log^{\delta-\varepsilon} N})$-time $\log^{2-\delta-\varepsilon} k$-approximation algorithm for GROUP-STEINER-TREE, thus proving Theorem 6.1

Since we take log from the expression, the above is also true if we replace k by N. Combined with [12] and [19], 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 6.2. *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 $\exp(2^{(1+o(1))\log^{\delta-\epsilon} n})$.*

Remark: We omit the algorithm for the Connected-Polymatroid problem, as its similar to the algorithm for Directed-Steiner-Tree. The lower bound holds because Connected-Polymatroid has Set-Cover as a special case.

Acknowledgments. The work of Marek Cygan is part of a project TOTAL that has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 677651). Guy Kortsarz was partially supported by NSF grant 1540547. Bundit Laekhanukit was partially supported by ISF grant no. 621/12, and I-CORE grant no. 4/11. Parts of the work were done while Guy Kortsarz and Bundit Laekhanukit were at the Weizmann Institute of Science, Israel.

References

[1] N. Bansal, P. Chalermsook, B. Laekhanukit, D. Nanongkai, and J. Nederlof. New tools and connections for exponential-time approximation. *CoRR*, abs/1708.03515, 2017.

[2] Y. Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996.

[3] E. Bonnet, B. Escoffier, E. J. Kim, and V. T. Paschos. On subexponential and fpt-time inapproximability. *Algorithmica*, 71(3):541–565, 2015. Preliminary version in IPEC'13.

[4] É. Bonnet, M. Lampis, and V. T. Paschos. Time-approximation trade-offs for inapproximable problems. *J. Comput. Syst. Sci.*, 92:171–180, 2018.

[5] É. Bonnet and V. T. Paschos. Sparsification and subexponential approximation. *Acta Inf.*, 55(1):1–15, 2018.

[6] N. Bourgeois, B. Escoffier, and V. T. Paschos. Efficient approximation of min set cover by moderately exponential algorithms. *Theor. Comput. Sci.*, 410(21-23):2184–2195, 2009.

[7] N. Bourgeois, B. Escoffier, and V. T. Paschos. Approximation of max independent set, min vertex cover and related problems by moderately exponential algorithms. *Discrete Applied Mathematics*, 159(17):1954–1970, 2011.

[8] G. Călinescu and A. Zelikovsky. The polymatroid steiner problems. *J. Comb. Optim.*, 9(3):281–294, 2005. Preliminary version in ISAAC'04.

[9] P. Chalermsook, B. Laekhanukit, and D. Nanongkai. Independent set, induced matching, and pricing: Connections and tight (subexponential time) approximation hardnesses. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 370–379, 2013.

[10] M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999. Preliminary version in SODA'98.

[11] C. Chekuri, G. Even, and G. Kortsarz. A greedy approximation algorithm for the group steiner problem. *Discrete Applied Mathematics*, 154(1):15–34, 2006.

[12] C. Chekuri and M. Pál. A recursive greedy algorithm for walks in directed graphs. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 245–253, 2005.

[13] V. Chvátal. A greedy heuristic for the set-covering problem. *Math. Oper. Res.*, 4(3):233–235, 1979.

[14] M. Cygan, L. Kowalik, M. Pilipczuk, and M. Wykurz. Exponential-time approximation of hard problems. *CoRR*, abs/0810.4934, 2008.

[15] M. Cygan, L. Kowalik, and M. Wykurz. Exponential-time approximation of weighted set cover. *Inf. Process. Lett.*, 109(16):957–961, 2009.

[16] I. Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007. Preliminary version in STOC'06.

[17] I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633, 2014.

[18] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

[19] G. Even, G. Kortsarz, and W. Slany. On network design problems: Fixed cost flows and the covering steiner problem. In *Algorithm Theory - SWAT 2002, 8th Scandinavian Workshop on Algorithm Theory, Turku, Finland, July 3-5, 2002 Proceedings*, pages 318–327, 2002.

[20] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. Preliminary version in STOC'03.

[21] U. Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998. Preliminary in STOC'96.

[22] D. Fotakis, M. Lampis, and V. T. Paschos. Sub-exponential approximation schemes for csps: From dense to almost sparse. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, pages 37:1–37:14, 2016.

[23] N. Garg, G. Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, 25-27 January 1998, San Francisco, California.*, pages 253–259, 1998.

[24] A. Gupta and A. Srinivasan. On the covering steiner problem. In *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings*, pages 244–251, 2003.

[25] E. Halperin and R. Krauthgamer. Polylogarithmic inapproximability. In *STOC*, pages 585–594, 2003.

[26] J. Håstad. Clique is hard to approximate within $\mathrm{n}^{\text{1-epsilon}}$. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 627–636, 1996.

[27] R. Impagliazzo and R. Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. Preliminary version in CCC'99.

[28] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. Preliminary version in FOCS'98.

[29] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. Syst. Sci.*, 9(3):256–278, 1974. Preliminary version in STOC'73.

[30] G. Kortsarz and D. Peleg. Approximating the weight of shallow steiner trees. *Discrete Applied Mathematics*, 93(2-3):265–285, 1999. Preliminary version in SODA'97.

[31] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41(5):960–981, 1994.

[32] D. Moshkovitz. The projection games conjecture and the np-hardness of ln n-approximating set-cover. *Theory of Computing*, 11:221–235, 2015. Preliminary version in APPROX'12.

[33] D. Moshkovitz and R. Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5):29:1–29:29, 2010. Preliminary version in FOCS'08.

[34] L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

[35] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007. Preliminary version in STOC'06.