New Tools and Connections for Exponential-time Approximation

Nikhil Bansal[1], Parinya Chalermsook[2], Bundit Laekhanukit[3], Danupon Nanongkai[4], and Jesper Nederlof[5]

1 **Eindhoven University of Technology, The Netherlands.** n.bansal@tue.nl.
2 **Aalto University, Finland.** parinya.chalermsook@aalto.fi.
3 **Weizmann Institute of Science, Israel.** bundit.laekhanukit@weizmann.ac.il.
4 **KTH, Royal Institute of Technology, Sweden.** danupon@kth.se
5 **Eindhoven University of Technology, The Netherlands.** j.nederlof@tue.nl.

Abstract

In this paper, we develop new tools and connections for *exponential time approximation*. In this setting, we are given a problem instance and a parameter $\alpha > 1$, and the goal is to design an α-approximation algorithm with the fastest possible running time. We show the following results:

1. An r-approximation for maximum independent set in $O^*(\exp(\tilde{O}(n/r \log^2 r + r \log^2 r)))$ time,
2. An r-approximation for chromatic number in $O^*(\exp(\tilde{O}(n/r \log r + r \log^2 r)))$ time,
3. A $(2 - 1/r)$-approximation for minimum vertex cover in $O^*(\exp(n/r^{\Omega(r)}))$ time, and
4. A $(k - 1/r)$-approximation for minimum k-hypergraph vertex cover in $O^*(\exp(n/(kr)^{\Omega(kr)}))$ time.

(Throughout, $\tilde{O}$ and O^* omit polyloglog(r) and factors polynomial in the input size, respectively.) The best known time bounds for all problems were $O^*(2^{n/r})$ [Bourgeois et al. 2009, 2011 & Cygan et al. 2008]. For maximum independent set and chromatic number, these bounds were complemented by $\exp(n^{1-o(1)}/r^{1+o(1)})$ lower bounds (under the Exponential Time Hypothesis (ETH)) [Chalermsook et al., 2013 & Laekhanukit, 2014 (Ph.D. Thesis)]. Our results show that the naturally-looking $O^*(2^{n/r})$ bounds are not tight for all these problems. The key to these algorithmic results is a *sparsification* procedure that reduces a problem to its bounded-degree variant, allowing the use of better approximation algorithms for bounded degree graphs. For obtaining the first two results, we introduce a new *randomized branching rule*.

Finally, we show a connection between PCP parameters and exponential-time approximation algorithms. This connection together with our independent set algorithm refute the possibility to overly reduce the size of Chan's PCP [Chan, 2016]. It also implies that a (significant) improvement over our result will refute the gap-ETH conjecture [Dinur 2016 & Manurangsi and Raghavendra, 2016].

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Approximations Algorithms, PCP's, Exponential Time Algorithms

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

The Independent Set, Vertex Cover, and Coloring problems are central problems in combinatorial optimization and have been extensively studied. Most of the classical results concern either approximation algorithms that run in polynomial time or exact algorithms that run in (sub)exponential-time. While these algorithms are useful in most scenarios, they lack flexibility: Sometimes, we wish for a better approximation ratio with worse running time (e.g.

computationally powerful devices), or faster algorithms with less accuracy. In particular, the trade-off between the running time and approximation ratios are needed in these settings.

Algorithmic results on the trade-offs between approximation ratio have been studied already in the literature in several settings, most notably in the context of *Polynomial-time Approximation Schemes (PTAS)*. For instance, in planar graphs, Baker's celebrated approximation scheme for several NP-hard problems [1] gives an $(1+\varepsilon)$-approximation for e.g. Independent Set in time $O^*(\exp(O(1/\varepsilon)))$ time. In graphs of small treewidth, Czumaj et al. [14] give an $O^*(\exp(tw/r))$ time algorithm that given a graph along with a tree decomposition of it of width at most tw, find an r-approximation for Independent Set. For general graphs, approximation results for several problems have been studied in several works (see e.g. [5, 6, 7, 13, 12, 11]). A basic building block that lies behind many of these results is to partition the input instance in smaller parts in which the optimal (sub)solution can be computed quickly (or at least faster than fully exponential-time). For example, to obtain an r-approximation for Independent Set one may arbitrarily partition the vertex set in r blocks and restrict attention to independent sets that are subsets of these blocks to get a $O^*(\exp(n/r))$ time r-approximation algorithm.

While at first sight one might think that such a naïve algorithm should be easily improvable via more advanced techniques, it was shown in [9, 5] that almost linear-size PCPs [15, 30] imply that r-approximating Independent Set [9] and Coloring [27] requires at least $\exp(n^{1-o(1)}/r^{1+o(1)})$ time assuming the popular Exponential Time Hypothesis (ETH). In the setting of the more sophisticated Baker-style approximation schemes for planar graphs, Marx [29] showed that no $(1+\varepsilon)$-approximating algorithm for planar Independent Set can run in time $O^*(\exp((1/\varepsilon)^{1-\delta}))$ assuming ETH, which implies that the algorithm of Czumaj cannot be improved to run in time $O^*(\exp(tw/r^{1+\varepsilon}))$.

These lower bounds, despite being interesting, are far from tight and by no means answer the question whether the known approximation trade-offs can be improved significantly, and in fact in many settings we are far from understanding the full power of exponential time approximation. For example we cannot exclude algorithms that 2-approximate k-Independent Set[1] in time $O^*(f(k))$ for some function f (see e.g. [26]), nor do we know algorithms that run asymptotically faster than the fastest exact algorithm that runs in time $n^{0.792k}$ time [31].

In this paper we aim to advance this understanding and study the question of designing as fast as possible approximation algorithms for Independent Set, Coloring and Vertex Cover in general (hyper)graphs.

Our Results.

For Independent Set our result is the following. Here we use $\tilde{O}$ to omit log log factors in r.

▶ **Theorem 1.** *There is a randomized algorithm that given an n-vertex graph G and integer r outputs an independent set that, with constant probability, has size at least $\alpha(G)/r$, where $\alpha(G)$ denotes the maximum independent set size of G. The algorithm runs in time $O^*(\exp(\tilde{O}(n/(r\log^2 r) + r\log^2 r)))$.*

To prove this result we introduce a new *randomized branching rule* that we will now introduce and put in context towards previous results. This follows a *sparsification technique* that reduces the maximum degree to a given number. This technique was already studied

[1] That is, given a graph and integer k answer YES if it has an independent set of size at least $2k$ and NO if it has no independent set of size at least k.

before in the setting of exponential time approximation algorithms Independent Set by Cygan et al. (see [11, paragraph 'Search Tree Techniques']) and Bourgeois et al. (see [7, Section 2.1]), but the authors did not obtain running times sub-exponential in n/r. Specifically, the sparsification technique is to branch (e.g. select a vertex and try to both include v in an independent set or discard and recurse for both possibilities) on vertices of sufficiently high degree. The key property is that if we decide to include a vertex and the independent set, we may discard all neighbors of v. If we generate instances by keep branching on vertices of degree at least d until the maximum degree is smaller than d, then at most $\binom{n}{n/d} \lessapprox \exp(n \log(d)/d)$ instances are created. In each such instance, the maximum independent set can be easily d-approximated by a greedy argument. Cygan et al. [11] note that this gives worse than $O^*(2^{n/r})$ running times.

Our algorithm works along this line but incorporates two (simple) ideas. Our first observation is that instead of solving each leaf instance by greedy d-approximation algorithm, one can use a recent $\tilde{O}(\frac{d}{\log^2 d})$ approximation algorithm by Bansal et al. [2] for Independent Set on bounded degree graphs. If we choose $d \approx r \log^2 r$, this immediately gives an improvement, an r-approximation in time essentially $\mathsf{exp}(\frac{n}{r \log r})$. To improve this further we use present an additional (more innovative) idea introducing randomization. This idea relies on the fact that in the sparsification step we have (unexploited) slack as we aim for an approximation.[2] Specifically, whenever we branch, we only consider the 'include' branch with probability $1/r$. This will lower the expected number of produced leaf instances in the sparsification step to $2^{n/d} \approx \mathsf{exp}(\frac{n}{r \log^2 r})$ and preserves the approximation factor with good probability.

Via fairly standard methods (see e.g. [4]) we show this also gives a faster algorithm for coloring in the following sense:

▶ **Theorem 2.** *There is a randomized algorithm that, given an n-vertex graph G and an integer $r > 0$, outputs with constant probability a proper coloring of G using at most $r \cdot \chi(G)$ colors. The algorithm runs in time $O^*(\exp(\tilde{O}(n/(r \log r) + r \log^2 r)))$.*

As a final indication that sparsification is a very powerful tool to obtain fast exponential time approximation algorithms, we show that a combination of a result of Halperin [20] and the sparsification Lemma [22] gives the following result for the Vertex Cover problem in hypergraphs with edges of size at most k (or Set Cover problem with frequency at most k).

▶ **Theorem 3.** *For every k, there is an $r_0 := r(k)$ such that for every $r \geq r_0$ there is an $O^*(\exp(\frac{n}{(kr)^{\Omega(kr)}}))$ time $(k - \frac{1}{r})$-approximation algorithm for the Vertex Cover problem in hypergraphs with edges of size at most k.*

Note that for $k = 2$ (e.g. vertex cover in graphs), this gives an $O^*(\exp(\frac{n}{r^{\Omega(r)}}))$ running time, which gives an exponential improvement (in the denominator of the exponent) upon the $(2 - 1/r)$ approximation by Bonnet et al. [7] that runs in time $O^*(2^{n/r})$. It was recently brought to our attention that Williams and Yu [32] independently have unpublished results for (hypergraph) vertex cover and independent set using sparsification techniques similar to ours.

Connections to PCP parameters The question of approximating the maximum independent set problem in sub-exponential time has close connections to the trade-off between three important parameters of PCPs: *size*, *gap* and *free-bit*. We discuss the implications of our algorithmic results in terms of these PCP parameters.

[2] This observation was already made by Bourgeois et al. [7], but we exploit it in a new way.

Roughly speaking, the gap parameter is the ratio between completeness and soundness, while the *freeness* parameter is the number of distinct proofs that would cause the verifier to accept; the *free-bit* is simply a logarithm of freeness. For convenience, we will continue our discussions in terms of freeness, instead of freebit.

- **Freebit v.s. gap:** The dependency between freeness and gap has played important role in hardness of approximation. Most notably, the existence of PCPs with freeness $g^{o(1)}$ where g is a gap parameter is "equivalent" to $n^{1-o(1)}$ hardness of approximating maximum independent set [21, 3]; this result is a building block for proving other hardness of approximation for many other combinatorial problems, e.g., coloring [19], disjoint paths, induced matching, cycle packing, and pricing. So it is fair to say that this PCP parameter trade-off captures the approximability of many natural combinatorial problems.
  Better parameter trade-off implies stronger hardness results. The existence of a PCP with arbitrarily large gap and freeness 1 (lowest possible) is in fact equivalent to $(2-\epsilon)$ inapproximability for Vertex Cover. The best known trade-off is due to Chan [10]: For any $g > 0$, there is a polynomial-sized PCP with gap g and freeness $O(\log g)$, yielding the best known NP-hardness of approximating maximum independent set in sparse graphs, i.e. $\Omega(d/\log^4 d)$ NP-hardness of approximating maximum independent set in degree-d graphs. [3]
- **Size, freebit, and gap:** When a polynomial-time approximation algorithm is the main concern, polynomial size PCPs are the only thing that matter. But when it comes to exponential time approximability, another important parameter, *size* of the PCPs, has come into play. The trade-off between size, freebit, and gap tightly captures the (sub-)exponential time approximability of many combinatorial problems. For instance, for any $g > 0$, Moshkovitz and Raz [30] constructs a PCP of size $n^{1+o(1)}$ and freeness $2^{O(\sqrt{\log g})}$; this implies that r-approximating Independent Set requires time $2^{n^{1-o(1)}/r^{1+o(1)}}$ [9].

Our exponential-time approximation result for Independent Set implies the following tradeoff results.

▶ **Corollary 4.** *Unless ETH breaks, a freebit PCP with gap g, freeness F and size S must satisfy $F \cdot S = \Omega(n \log^2 g)$.*

In particular, this implies that (i) Chan's PCP cannot be made smaller size than $o(n \log g)$, unless ETH breaks, and (ii) in light of the equivalence between gap-amplifying freebit PCPs with freeness 1 and $(2-\epsilon)$ approximation for Vertex Cover, our result shows that such a PCP must have size at least $\Omega(n \log^2 g)$. We remark that no such trade-off results are known for polynomial-sized PCPs. To our knowledge, this is the first result of its kind.

Further related results The best known results for Independent Set in the polynomial-time regime are an $O(\frac{n(\log\log n)^2}{\log^3 n})$-approximation [17], and the hardness of $n/\mathsf{exp}(O(\log^{3/4+o(1)} n))$ (which also holds for Coloring) [24]. For Vertex Cover, the best known hardness of approximation is $(\sqrt{2}-o(1))$ NP-hardness [23] and $(2-\epsilon)$ hardness assuming the unique games conjecture [25]. All three problems (Independent Set, Coloring, and Vertex Cover) do not admit exact algorithms that run in time $2^{o(n)}$, unless ETH fails. Besides the aforementioned works [7, 11] sparsification techniques for exponential time approximation were studied by Bonnet and Paschos in [6], but mainly hardness results were obtained.

[3] Roughly speaking, the existence of a PCP with freeness $F(g)$ (where g is a gap) implies $\Omega(\frac{d}{F(d)\log^3 d})$ hardness of approximating independent set in degree-d graphs.

2 Preliminaries

We first formally define the three problems that we consider in this paper. Independent Set: Given a graph $G = (V, E)$, we say that $J \subseteq V$ is an independent set if there is no edge with both endpoints in J. The goal of Independent Set is to output an independent set J of maximum cardinality. Denote by $\alpha(G)$, the cardinality of the maximum independent set. Vertex Cover: Given a graph $G = (V, E)$, we say that $J \subseteq V$ is a vertex cover of G if every edge is incident to at least one vertex in J. The goal of Vertex Cover is to output a vertex cover of minimum size. A generalization of vertex cover, called k-Hypergraph Vertex Cover k-Vertex Cover, is defined as follows. Given a hypergraph $G = (V, \mathcal{E})$ where each hyperedge $h \in \mathcal{E}$ has cardinality at most k, the goal is to find a collection of vertices $J \subseteq V$ such that each hyperedge is incident to at least one vertex in J, while minimizing $|J|$. The *degree* $\Delta(H)$ of hypergraph H is the maximum frequency of an element. Coloring: Given a graph $G = (V, E)$, a proper k-coloring of G is a function $f : V \to [k]$ such that $f(u) \neq f(v)$ for all $uv \in E$. The goal of Coloring is to compute a minimum integer $k > 0$ such that G admits a (proper) k-coloring; this number is referred to as the *chromatic number*, denote $\chi(G)$.

For a graph $G = (V, E)$, $N_G(v)$ denotes the set of neighbors of v and $d_G(v)$ denotes $|N_G(v)|$. If $X \subseteq V$ we let $G[X]$ denote the graph $(X, E \cap (X \times X))$ i.e. the subgraph of G induced by X We use $\exp(x)$ to denote 2^x in order to avoid superscripts. We use the $O^*(\cdot)$-notation to suppress factors polynomial in the input size. We use $\tilde{O}$ and $\tilde{\Omega}$ to suppress factors polyloglog in r in respectively upper and lower bounds and write $\tilde{\Theta}$ for all functions that are in both $\tilde{O}$ and $\tilde{\Omega}$.

3 Faster Approximation via Randomized Branching and Sparsification

3.1 Maximum Independent Set

In this section we prove Theorem 1. Below is our key lemma.

▶ **Lemma 5.** *Suppose there is an approximation algorithm* $\texttt{dIS}(G, r)$ *that runs in time* $T(n, r)$ *and outputs an Independent Set of* G *of size* $\alpha(G)/r$ *if* G *has maximum degree* $d(r)$*, (where* $d(r) \geq 2r$*). Then there is an algorithm* $\texttt{IS}(G, r)$ *running in expected time* $O^*\left(\exp\left(\frac{n}{d(r)} \log(4d(r)/r)\right) T(n, r)\right)$ *that outputs an independent set of expected size* $\alpha(G)/r$*.*

Proof. Consider the following algorithm.

Algorithm $\texttt{IS}(G = (V, E), r)$
1: **if** $\exists v \in V : d_G(v) \geq d(r)$ **then**
2: Draw a random Boolean variable b such that $\Pr[b = \textbf{true}] = 1/r$.
3: **if** $b = \textbf{true}$ **then**
4: **return** the largest of $IS(G[V \setminus v])$ and $IS(G[V \setminus N(v)]) \cup \{v\}$.
5: **else**
6: **return** $IS(G[V \setminus v])$.
7: **else**
8: **return** $dIS(G)$.

Figure 1 Approximation algorithm for Independent Set using an approximation algorithm dIS that works in bounded degree graphs.

For convenience, let us fix r and $d := d(r)$. We start by analyzing the expected running time of this algorithm. Per recursive call the algorithm clearly uses $O^*(T(n,r))$ time. It remains to bound the number of recursive calls $R(n)$ made by $\texttt{IS}(G,r)$ when G has n vertices. We will bound $R(n) \leq 2^{\lambda n}$ for $\lambda = \log(4d/r)/d$ by induction on n. Note that here λ is chosen such that

$$\exp(-\lambda \cdot d) = r/(4d) \leq \frac{r\log(4d/r)}{2d}, \tag{1}$$

where we use $d/r \geq 2$ for the inequality. For the base case of the induction, note that if the condition at Line 1 does not hold, the algorithm does not use any recursive calls and the statement is trivial as λ is clearly positive. For the inductive step, we see that

$$\begin{aligned}
R(n) &\leq R(n-1) + \Pr[b = \textbf{true}] \cdot R(n-d) \\
&= R(n-1) + R(n-d)/r \\
&= \exp(\lambda(n-1)) + \exp(\lambda(n-d))/r \\
&= \exp(\lambda n)\left(\exp(-\lambda) + \exp(-\lambda d)/r\right) && \text{Using } \exp(-x) \leq 1 - x/2 \text{ for } x \in [0,1] \\
&\leq \exp(\lambda n)\left(1 - \lambda/2 + \exp(-\lambda d)/r\right) && \text{Using } \exp(-\lambda \cdot d(r)) \leq \lambda r/2 \text{ from (1)} \\
&\leq \exp(\lambda n).
\end{aligned}$$

We continue by analyzing the output of the algorithm. It clearly returns a valid independent set as all neighbors of v are discarded when v is included in Line 4 and an independent set is returned at Line 8. It remains to show $\mathbb{E}[|\texttt{IS}(G,r)|] \geq \alpha(G)/r$ which we do by induction on n. In the base case in which no recursive call is made, note that on Line 8 we indeed obtain an r-approximation as G has maximum degree $d(r)$. For the inductive case, let X be a maximum independent set of G and let v be the vertex as picked on Line 1. We distinguish two cases based on whether $v \in X$. If $v \notin X$, then $\alpha(G) = \alpha(G[V \setminus v])$ and the inductive step follows as $\mathbb{E}[|\texttt{IS}(G[V \setminus v], r)|] \geq \alpha(G)/r$ by the induction hypothesis. Otherwise, if $v \in X$, then $\mathbb{E}[|\texttt{IS}(G,r)|]$ is at least

$$\begin{aligned}
&\Pr[b = \textbf{false}] \cdot \mathbb{E}[|\texttt{IS}(G[V \setminus \{v\}], r)|] + \Pr[b = \textbf{true}] \cdot \mathbb{E}[|\texttt{IS}(G[N \setminus N(v)], r)| + 1] \\
\geq\ & \left(1 - \tfrac{1}{r}\right)\frac{\alpha(G)-1}{r} + \tfrac{1}{r}\left(\frac{\alpha(G)-1}{r} + 1\right) \\
=\ & \frac{\alpha(G)-1}{r} + \tfrac{1}{r} = \alpha(G)/r,
\end{aligned}$$

as required. Here the first inequality uses the induction hypothesis twice. ◀

We will invoke the above lemma by using the algorithm $dIS(G)$ by Bansal et al. [2] implied by the following theorem:

▶ **Theorem 6** ([2], Theorem 1.3)**.** *There is an $\tilde{O}(d/\log^2 d)$ approximation algorithm $dIS(G)$ for Independent Set on graphs of maximum degree d running in time $O^*(\exp(O(d)))$.*

Proof of Theorem 1. We may apply Lemma 5 with $r/3$ and, by virtue of Theorem 6, with $d(r/3) = \tilde{\Theta}(r\log^2 r)$, and $T(n,r) = O^*(\exp(\tilde{O}(r\log^2 r)))$. We obtain an $O^*(\exp(\tilde{O}(n/r\log^2 r + r\log^2 r)))$ expected time algorithm that outputs an independent set of expected size $2\alpha(G)/r$.

Since the size of the output is upper bounded by $\alpha(G)$ we obtain an independent set of size at least $\alpha(G)/r$ with probability at least $1/(3r)$, and we may boost this probability to $3/4$ by $O(r)$ repetitions.

By Markov's inequality these repetitions together run in $O^*(\exp(\tilde{O}(n/r \log^2 r + r \log^2 r)))$ time with probability 3/4. The theorem statement follows by a union bound as these $O(r)$ repetitions run in the claimed running time and simultaneously some repetition finds an independent set of size at least $\alpha(G)/r$, with probability at least 1/2. ◀

A deterministic algorithm: Additionally, we also show a deterministic r-approximation algorithm that runs in time $\exp(\tilde{O}(n/r \log r))$. The algorithm utilizes Feige's algorithm [17] as a blackbox, and is deferred to Appendix A.

3.2 Graph Coloring

Now we use the approximation algorithm for Independent Set as a subroutine for an approximation algorithm for Coloring to prove Theorem 2 as follows:

Proof of Theorem 2. The algorithm combines the approximation algorithm `IS` from Section 3.1 for Independent Set with an exact algorithm `optcol` for Coloring (see, e.g., [4]) as follows:

Algorithm `CHR`$(G = (V, E), r)$
1: Let $n = |V|$, $c = 0$.
2: **while** $|V| \geq n/(r \log r)$ **do**
3: $\quad c \leftarrow c + 1$.
4: $\quad C_c \leftarrow$ `IS`$(G[V], r/\ln(r \log r))$.
5: $\quad V \leftarrow V \setminus C_c$.
6: Let $(C_{c+1}, \ldots, C_\ell) \leftarrow$ `optcol`$(G[V])$ be some optimum coloring of the remaining graph $G(V)$.
7: **return** $(C_1, \ldots, C_\ell)$.

Figure 2 Approximation algorithm for the chromatic number.

We claim that `CHR`(G, r) returns with high probability a proper coloring of G using $\ell \leq (r+2) \cdot \chi(G)$ colors. To prove the theorem, we invoke `CHR`$(G, r-2)$ which has the same asymptotic running time. First, note that in each iteration of the while loop (Line 2 of Algorithm 2), $|V|$ is decreased by a multiplicative factor of at most $1 - \frac{\ln(r \log r)}{r \cdot \chi(G)}$ because $G[V]$ must have an independent set of size at least $n/\chi(G)$ and therefore $|C_c| \geq \ln(r \log r) n/(r \cdot \chi(G))$. Before the last iteration, we have $|V| \geq n/(r \ln r)$. Thus, the number ℓ of iterations must satisfy

$$1/(r \log r) \leq \left(1 - \frac{\ln(r \log r)}{r \cdot \chi(G)}\right)^{\ell-1} \leq \exp\left(-\frac{\ln(r \log r)(\ell-1)}{r \cdot \chi(G)}\right).$$

This implies that $(\ell - 1) \leq r \cdot \chi(G)$. Consequently, the number of colors used in the first phase of the algorithm (Line 1 to Line 5) is $c \leq r\chi(G) + 1$. The claimed upper bound on ℓ follows because the number of colors used for $G[V]$ in the second phase (Line 6) is clearly upper bounded by $\chi(G)$.

To upper bound the running time, note that Line 4 runs in time

$$\exp\left(\tilde{O}\left(\frac{n \ln(r \log r)}{r \log^2(r/\ln(r \log r))} + r \log^2 r\right)\right) = \exp\left(\tilde{O}\left(\frac{n}{r \log r}\right) + r \lg^2 r\right),$$

and implementing `optcol`$(G = (V, E))$ by using the $O^*(2^{|V|})$ time algorithm from [4], Line 6 also takes $O^*(2^{n/(r \log r)})$ time and the running time follows. ◀

3.3 Vertex Cover and Hypergraph Vertex Cover

In this section, we show an application of the sparsification technique to Vertex Cover to obtain Theorem 3. Here the sparsification step is not applied explicitly. Instead, we utilize the sparsification Lemma of Impagliazzo et al. [22] as a blackbox. Subsequently, we solve each low-degree instance by using an algorithm of Halperin [20]. The sparsification lemma due to Impagliazzo et al. [22], shows that an instance of the k-Hypergraph Vertex Cover problem can be reduced to a (sub-)exponential number of low-degree instances.[4]

▶ **Lemma 7** (Sparsification Lemma, [22, 8])**.** *There is an algorithm that, given a hypergraph $H = (V, \mathcal{E})$ with edges of size at most $k \geq 2$, a real number $\varepsilon > 0$, produces set systems $H_1 = (V, \mathcal{E}_1), \ldots, H_\ell = (V, \mathcal{E}_\ell)$ with edges of size at most k in $O^*(\ell)$ time such that*

1. *every subset $X \subseteq V$ is a vertex cover of H if and only if X is a vertex cover of H_i for some i,*
2. *for every $i = 1, \ldots, \ell$, the degree $\Delta(H_i)$ is at most $(k/\varepsilon)^{3k}$,*
3. *ℓ is at most $\exp(\varepsilon n)$.*

The next tool is an approximation algorithm for the k-Hypergraph Vertex Cover problem when the input graph has low degree due to Halperin [20].

▶ **Theorem 8** ([20])**.** *There is a polynomial time $k - (1 - o(1))\frac{k(k-1)\ln\ln\Delta}{\ln\Delta}$-approximation algorithm for the vertex cover problem in hypergraphs with edges of size at most k in which every element has degree at most Δ, for large enough $\Delta := \Delta(k)$.*

Now we complete the proof of the theorem by applying Lemma 7 with parameter $\varepsilon = k/(kr)^{kr}$. The number of low-degree instances H_i produced by Lemma 7 is at most $\exp(\varepsilon n) = \exp\left(O\left(\frac{k}{(kr)^{kr}}\right)\right)$. Each graph H_i has degree at most $\Delta(H_i) \leq (k/\varepsilon)^{3k} = (kr)^{3k^2 r}$. Note that

$$\frac{\ln\ln\Delta(H_i)}{\ln\Delta(H_i)} \geq \frac{\ln(3k^2 r \ln(kr))}{3k^2 r \ln(kr)} \geq \frac{1}{3k^2 r}.$$

Plugging this value of $\Delta(H_i)$, Halperin's algorithm gives the approximation factor of

$$k - \frac{k(k-1)\ln\ln\Delta}{\ln\Delta} \leq k - \frac{1}{6r}.$$

Thus this gives an $k - 1/(6r)$-approximation running in time $O^*(\exp(nk/(kr)^{kr}))$ which translates to an $k - 1/r$-approximation running in time $O^*(\exp(nk/(kr/6)^{kr/6}))$.

4 PCP Parameters and Exponential-time approximation hardness

Exponential-time approximation has connections to the trade-off questions between three parameters of PCPs: *size*, *freebit*, and *gap*. To formally quantify this connection, we define new terms, formally illustrating the ideas that have been already around in the literature.

[4] The original formulation is for the Set Cover problem and the most popular formulation is for CNF-SAT problem, but they are all equivalent by direct transformation.

We define a class of languages FGPCP which stands for *Freebit and Gap-amplifiable PCP*. Let g be a positive real, and S, F be non-decreasing functions. A language L is in $\mathsf{FGPCP}_c(S, F)$ if there is a constant $g_0 > 1$ such that, for all constants $g \geq g_0$, there is a verifier V_g that, on input $x \in \{0,1\}^n$, has access to a proof $\pi : |\pi| = O(S(n, g))$ and satisfies the properties:

- The verifier runs in $2^{o(n)}$ time.
- If $x \in L$, then there is a proof π such that $V_g^\pi(x)$ accepts with probability $\geq c$.
- If $x \notin L$, then for any proof π, $V_g^\pi(x)$ accepts with probability $\leq c/g$.
- For each x and each random string r, the verifier has $\leq F(g)$ accepting configurations.

The parameters g, S and $\log F$ are referred to as *gap*, *size* and *freebit* of the PCPs respectively. For convenience, we call $F(g)$ the *freeness* of the PCP. An intuitive way to view this PCP is as a class of PCPs parameterized by gap g. An interesting question in the PCPs and hardness of approximation literature has been to find the smallest functions S and F.

▶ **Theorem 9.** *If* $\mathsf{SAT} \in \mathsf{FGPCP}_\delta(S, F)$ *for some function* $S(n, g)$ *that is at least linearly growing in* n*, then for any constant* r*,* r*-approximating* Independent Set*, in input graph* G*, cannot be done in time* $2^{o(S^{-1}(|V(G)|,r)/rF(r))}$ *unless ETH fails. (we think of* r *as a fixed number, and therefore* $S(n, r)$ *should be seen as a function on a single variable* n*.)*

We prove the theorem later in this section.

▶ **Corollary 10.** *Assuming that* SAT *has no* $2^{o(n)}$*-time randomized algorithm and that* $\mathsf{SAT} \in \mathsf{FGPCP}_\delta(S, F)$*, then it must be the case that* $S(n, g) \cdot F(g) = \Omega(n \cdot \frac{\log^2 g}{\mathsf{poly}(\log \log g)})$.

Proof. Otherwise, $S^{-1}(|V(G)|, r) = o(|V(G)| \cdot \frac{F(r)\mathsf{poly}(\log \log r)}{\log^2 r})$, and the Theorem 9 would imply that there is no $2^{o(|V(G)| \cdot \frac{\mathsf{poly}(\log \log r)}{r \log^2 r})}$, contradicting the existence of our Independent Set approximation algorithm. ◀

Now let us phrase the known PCPs in our framework of FGPCP. Chan's PCPs [10] can be stated that $\mathsf{SAT} \in \mathsf{FGPCP}_{1-o(1)}(\mathrm{poly}, O(\log g))$. Applying our results, this means that if one wants to keep the same freebit parameters given by Chan's PCPs, then the size must be at least $\Omega(n \log g)$. Another interesting consequence is a connection between Vertex Cover and Freebit PCPs in the polynomial time setting [3].

▶ **Theorem 11** ([3])**.** Vertex Cover *is* $(2 - \epsilon)$ *hard to approximate if and only if* $\mathsf{SAT} \in \mathsf{FGPCP}_{1/2-\epsilon}(\mathsf{poly}, 1)$.

The intended PCPs in Theorem 11 have arbitrary small soundness while the freeness remains 1. Our Corollary 10 implies that such a PCP must have size at least $\Omega(n \log^2 g)$.

4.1 Proof of Theorem 9

Step 1: Creating a hard CSP We will need the following lemma that creates a "hard" CSP from FGPCP. This CSP will be used later to construct a hard instance of Independent Set.

▶ **Lemma 12.** *If* $\mathsf{SAT} \in \mathsf{FGPCP}_\delta(S, F)$*, then, for any* $g > 1$*, there is a randomized reduction from an* n*-variable* SAT ϕ *to a* CSP ϕ' *having the following properties (w.h.p.):*

- *The number of variables of* ϕ' *is* $\leq S(n)$.
- *The number of clauses of* ϕ' *is* $\leq 10S(n)g/\delta$.
- *The freeness of* ϕ' *is* $\leq F(g)$.
- *If* ϕ *is satisfiable, then* $\mathsf{val}(\phi') \geq \delta/2$*. Otherwise,* $\mathsf{val}(\phi') \leq 6\delta/g$.

Proof. Let g be any number and V_g be the corresponding verifier. On input ϕ, we create a CSP ϕ' as follows. For each proof bit Π_i, we have variable x_i. The set of variables is $X = \{x_1, \ldots, x_{S(n)}\}$. We perform $M = 10\lceil S(n)g/\delta \rceil$ iterations. In iteration j, the verifier picks a random string r_j and create a predicate $P_j(x_{b_1}, \ldots, x_{b_q})$, where $b_1, \ldots, b_q$ are the proof bits read by the verifier V_g^{Π} on random string r_j. This predicate is true on assignment γ if and only if the verifier accepts the local assignment where $\Pi_{b_i} = \gamma(x_i)$ for all $i \in [q]$.

First, assume that ϕ is satisfiable. Then there is a proof Π^* such that the verifier $V^{\Pi^*}(\phi)$ accepts with probability δ. Let $\gamma : X \to \{0,1\}$ be an assignment that agrees with the proof Π^*. So γ satisfies each predicate P_j with probability δ, and therefore, the expected number of satisfied predicates is δM. By Chernoff's bound, the probability that γ satisfies less than $\frac{\delta M}{2}$ predicates is at most $2^{-\delta M/8} \leq 2^{-n}$.

Next, assume that ϕ is not satisfiable. For each assignment $\gamma : X \to \{0,1\}$, the fraction of random strings satisfied by the corresponding proof Π_γ is at most δ/g. When we pick a random string r_j, the probability that $V^{\Pi_\gamma}(\phi, r_j)$ accepts is then at most δ/g. So, over all the choices of M strings, the expected number of satisfied predicates is $\delta M/g \geq 10S(n)$. By Chernoff's bound, the probability that γ satisfies more than $\delta M/g$ predicates is at most $2^{-10S(n)}$. By union bound over all possible proofs of length $S(n)$ (there are $2^{S(n)}$ such proofs), the probability that there is such a γ is at most $2^{S(n)}2^{-10S(n)} \leq 2^{-S(n)}$. ◀

Step 2: FGLSS reduction The FGLSS reduction is a standard reduction from CSP to Independent Set introduced by Feige et al. [18]. The reduction simply lists all possible configurations (partial assignment) for each clause as vertices and adding edges if there is a conflict between two configuration. In more detail, for each predicate P_i and each partial assignment γ such that $P_i(\gamma)$ is true, we have a vertex $v(i,\gamma)$. For each pair of vertices $v(i,\gamma)v(i',\gamma')$ such that there is a variable appearing in both P_i and $P_{i'}$ for which $\gamma(x_j) \neq \gamma'(x_j)$, we have an edge between $v(i,\gamma)$ and $v(i',\gamma')$.

▶ **Lemma 13** (FGLSS Reduction [18]). *There is an algorithm that, given an input* CSP ϕ *with m clauses, n variables, and freeness F, produces a graph $G = (V, E)$ such that (i) $|V(G)| \leq mF$ and (ii) $\alpha(G) = \mathsf{val}(\phi)m$, where $\mathsf{val}(\phi)$ denotes the maximum number of predicates of ϕ that can be satisfied by an assignment.*

Combining everything Assume that $\mathsf{SAT} \in \mathsf{FGPCP}_\delta(S, F)$. Let $g > 0$ be a constant and V_g be the verifier of SAT that gives the gap of g. By invoking Lemma 12, we have a CSP ϕ_1 with $S(n,g)$ variables and $100S(n,g)g/\delta$ clauses. Moreover, the freeness and gap of ϕ_1 are $F(g)$ and g respectively. Applying the FGLSS reduction, we have a graph G with $N = |V(G)| = 100S(n,g)F(g)g/\delta = O(S(n,g)F(g)g)$. Now assume that we have an algorithm $\mathcal{A}$ that gives a g approximation in time $2^{\frac{o(S^{-1}(N,g))}{gF(g)}}$. Notice that $S^{-1}(N,g) \leq O(ngF(g))$ and therefore algorithm $\mathcal{A}$ distinguishes between Yes- and No-instance in time $2^{o(n)}$, a contradiction.

Hardness under Gap-ETH: Dinur [16] and Manurangsi and Raghavendra [28] made a conjecture that SAT does not admit an approximation scheme that runs in $2^{o(n)}$ time. We observe a Gap-ETH hardness of r-approximating Independent Set in time $2^{n/r^c}$ for some constant c. The proof uses a standard amplification technique and is deferred to Appendix B.

5 Further Research

Our work leaves ample opportunity for exciting research. An obvious open question is to derandomize our branching, e.g. whether Theorem 1 can be proved without randomized

algorithms. While the probabilistic approximation guarantee can be easily derandomized by using a random partition of the vertex set in r parts or splitters, it seems harder to strengthen the expected running time bound to a worst-case running time bound.

Can we improve the running times of the other algorithms mentioned in the introduction that use the partition argument, possibly using the randomized branching strategy? Specifically, can we $(1+\varepsilon)$-approximate Independent Set on planar graphs in time $O^*(2^{(1/\varepsilon)/\log(1/\varepsilon)})$, or r-approximate Independent Set in time $O^*(2^{tw/r \log r})$? As mentioned in the introduction, a result of Marx [29] still leaves room for such lower order improvements. Another open question in this category is how fast we can r-approximate k-Independent Set, where the goal is to find an independent st of size of k. For example no $O(n^{k/f(r)})$ time algorithm is known, where $f(r)$ is a non-trivial function of r, that distinguishes graphs G with $\alpha(G) \geq 2k$ from graphs with $\alpha(G) \leq k$. The partition argument gives only a running time of $(n/r)^{0.792k}$, and no strong lower bounds are known for this problem. Finally, a big open question in the area is to find or exclude a $(2-\varepsilon)$-approximation for Vertex Cover in graphs in subexponential time for some fixed constant $\varepsilon > 0$.

Acknowledgment NB is supported by a NWO Vidi grant 639.022.211 and ERC consolidator grant 617951. BL is supported by ISF Grant No. 621/12 and I-CORE Grant No. 4/11. DN is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 715672 and the Swedish Research Council (Reg. No. 2015-04659). JN is supported by NWO Veni grant 639.021.438.

References

1 Brenda S. Baker. Approximation algorithms for np-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994.

2 Nikhil Bansal, Anupam Gupta, and Guru Guruganesh. On the Lovász Theta Function for Independent Sets in Sparse Graphs. In *Symposium on Theory of Computing, STOC*, pages 193–200, 2015.

3 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, pcps, and nonapproximability-towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998.

4 Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.

5 Édouard Bonnet, Michael Lampis, and Vangelis Th. Paschos. Time-approximation trade-offs for inapproximable problems. In *Symposium on Theoretical Aspects of Computer Science, STACS*, pages 22:1–22:14, 2016.

6 Édouard Bonnet and Vangelis Th. Paschos. Sparsification and subexponential approximation. *Acta Informatica*, pages 1–15, 2016.

7 Nicolas Bourgeois, Bruno Escoffier, and Vangelis Th. 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 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. A duality between clause width and clause density for SAT. In *Conference on Computational Complexity (CCC)*, pages 252–260, 2006.

9 Parinya Chalermsook, Bundit Laekhanukit, and Danupon Nanongkai. Independent set, induced matching, and pricing: Connections and tight (subexponential time) approximation hardnesses. In *Foundations of Computer Science, FOCS*, pages 370–379, 2013.

10 Siu On Chan. Approximation resistance from pairwise-independent subgroups. *J. ACM*, 63(3):27:1–27:32, 2016.

11 Marek Cygan, Lukasz Kowalik, Marcin Pilipczuk, and Mateusz Wykurz. Exponential-time approximation of hard problems. *CoRR*, abs/0810.4934, 2008.

12 Marek Cygan, Lukasz Kowalik, and Mateusz Wykurz. Exponential-time approximation of weighted set cover. *Inf. Process. Lett.*, 109(16):957–961, 2009.

13 Marek Cygan and Marcin Pilipczuk. Exact and approximate bandwidth. *Theor. Comput. Sci.*, 411(40-42):3701–3713, 2010.

14 Artur Czumaj, Magnús M. Halldórsson, Andrzej Lingas, and Johan Nilsson. Approximation algorithms for optimization problems in graphs with superlogarithmic treewidth. *Inf. Process. Lett.*, 94(2):49–53, 2005.

15 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007.

16 Irit Dinur. Mildly exponential reduction from gap 3sat to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:128, 2016.

17 Uriel Feige. Approximating maximum clique by removing subgraphs. *SIAM J. Discrete Math.*, 18(2):219–225, 2004.

18 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, 1996.

19 Uriel Feige and Joe Kilian. Zero knowledge and the chromatic number. *J. Comput. Syst. Sci.*, 57(2):187–199, 1998.

20 Eran Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. *SIAM J. Comput.*, 31(5):1608–1623, 2002.

21 Johan Håstad. Clique is hard to approximate within $n^{1\text{-epsilon}}$. In *37th Annual Symposium on Foundations of Computer Science, FOCS*, pages 627–636, 1996.

22 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

23 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games and grassmann graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:124, 2016.

24 Subhash Khot and Ashok Kumar Ponnuswami. Better inapproximability results for max-clique, chromatic number and min-3lin-deletion. In *Automata, Languages and Programming, International Colloquium, (ICALP)*, pages 226–237, 2006.

25 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008.

26 Subhash Khot and Igor Shinkar. On hardness of approximating the parameterized clique problem. In *Innovations in Theoretical Computer Science (ITCS)*, pages 37–45, New York, NY, USA, 2016. ACM. doi:10.1145/2840728.2840733.

27 Bundit Laekhanukit. *Inapproximability of Combinatorial Problems in Subexponential-Time*. PhD thesis, McGill University, 2014.

28 Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense csps. *CoRR*, abs/1607.02986, 2016.

29 Dániel Marx. On the optimality of planar and geometric approximation schemes. In *Foundations of Computer Science (FOCS)*, pages 338–348, 2007.

30 Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5):29:1–29:29, 2010.

31 Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 026(2):415–419, 1985.

32 Ryan Williams and Huacheng Yu. Personal communication.

A A Deterministic Algorithm for Independent Set

In this section, we give a deterministic r-approximation algorithm that runs in time $2^{O(n/r \log r)}$. This algorithm is a simple consequence of Feige's algorithm [17], that we restate below in a slightly different form.

▶ **Theorem 14** ([17]). *Let G be a graph with independence ratio $\frac{\alpha(G)}{|V(G)|} = 1/k$. Then, for any parameter t, one can find an independent set of size $\Omega(t \cdot \log_k(\frac{n}{kt}))$ in time $\mathrm{poly}(n)k^{O(t)}$.*

Now, our algorithm proceeds as follows.

- If $\alpha(G) < n/\log^2 r$, we can enumerate all independent sets of size $n/(r\log^2 r)$ (this is an r-approximation) in time $\binom{n}{n/(r\log^2 r)} \leq (er\log^2 r)^{\frac{n}{r\log^2 r}} \leq 2^{O(n/(r\log r))}$.
- Otherwise, the independence ratio is at least $1/k$ where $k = \log^2 r$. We choose $t = n/(r\log r)$, so Feige's algorithm finds an independent set of size at least

$$\Omega\left(t \cdot \log_k(\frac{n}{kt})\right) = \Omega\left(\frac{n}{r\log r} \cdot \log_k(r\log r)\right) = \Omega(n/(r\log\log r))$$

The running time is

$$k^{O(t)} = 2^{O(\frac{n(\log\log r)}{r\log r})}$$

If we redefine $r' = r\log\log r$, then the algorithm is an r'-approximation algorithm that runs in time $2^{O(n(\log\log r')^2/r'\log r')}$.

B Gap-ETH hardness of Independent Set (sketch)

We now sketch the proof. We are given an n-variable 3-CNF-SAT formula ϕ with perfect completeness and soundness $1-\epsilon$ for some $\epsilon > 0$. We first perform standard amplification and sparsification to get ϕ' with gap parameter g, the number of clauses is ng, and freeness is $g^{O(1/\epsilon)}$. Now, we perform FGLSS reduction to get a graph G such that $|V(G)| = ng^{O(1/\epsilon)}$. Therefore, g-approximation in time $2^{o(|V(G)|/g^{O(1/\epsilon)})}$ would lead to an algorithm that satisfies more than $(1-\epsilon)$ fraction of clauses in 3-CNF-SAT formula in time $2^{o(n)}$. In other words, any $2^{n/r^c}$-time algorithm that r-approximates Independent Set can be turned into a $(1+O(1/c))$-approximation algorithm for approximating 3-CNF-SAT in sub-exponential time.