Towards Optimal Lower Bounds For Clique and Chromatic Number*

Lars Engebretsen[1,**] and Jonas Holmerin[2]

[1] MIT Laboratory for Computer Science
200 Technology Square, NE43-369
Cambridge, Massachusetts 02139-3594
E-mail: enge@mit.edu
Phone: (617) 253-1499 Fax: (617) 258-8682

[2] Dept. of Numerical Analysis and Computer Science
Royal Institute of Technology
SE-100 44 Stockholm, SWEDEN
E-mail: joho@nada.kth.se
Phone: +46 8 790 68 09 Fax: +46 8 790 09 30

December 4, 2000

Abstract. It was previously known that neither Max Clique nor Min Chromatic Number can be approximated in polynomial time within $n^{1-\epsilon}$, for any constant $\epsilon > 0$, unless **NP** = **ZPP**. In this paper, we extend the reductions used to prove these results and combine the extended reductions with a recent result of Samorodnitsky and Trevisan to show that unless **NP** $\subseteq$ **ZPTIME**$(2^{O(\log n (\log \log n)^{3/2})})$, neither Max Clique nor Min Chromatic Number can be approximated within $n^{1-O(1/\sqrt{\log \log n})}$. Since there exists polynomial time algorithms approximating both problems within $n^{1-O(\log \log n / \log n)}$, our result shows that the best possible ratio we can hope for is of the form $n^{1-o(1)}$, for some—yet unknown—value of $o(1)$ between $O(1/\sqrt{\log \log n})$ and $\Omega(\log \log n / \log n)$.

1 Introduction

The Max Clique problem, i.e., the problem of finding in a graph $G = (V, E)$ the largest possible subset C of the vertices in V such that every vertex in C has edges to all other vertices in C, is a well-known combinatorial optimization problem. The decision version of Max Clique was one of the problems proven to be **NP**-complete in Karp's original paper on **NP**-completeness [16], which means that we cannot hope to solve Max Clique efficiently, at least not if we want an exact solution. Thus, attention has

*A preliminary version of this paper appeared in Proceedings of 27th ICALP [7].

**Part of the work performed at the Department of Numerical Analysis and Computer Science, Royal Institute of Technology, Stockholm, SWEDEN.

turned to algorithms producing solutions which are at most some factor from the optimum value. It is trivial to approximate Max Clique in a graph with n vertices within n—just pick any vertex as the clique—and Boppana and Halldórsson [6] have shown that Max Clique can be approximated within $O(n/\log^2 n)$ in polynomial time. It is an astonishing, and unfortunate, result that it is hard to do substantially better than this. In fact, the Max Clique problem cannot be approximated within $n^{1-\epsilon}$, for any constant $\epsilon > 0$, unless $\mathbf{NP} = \mathbf{ZPP}$. The first to explore the possibility of proving strong lower bounds on the approximability of Max Clique were Feige et al. [10], who proved a connection between Max Clique and probabilistic proof systems. Their reduction was then improved independently by Bellare, Goldreich, and Sudan [4] and Zuckerman [23]. As the final link in the chain, Håstad [15] constructed a probabilistic proof system with the properties needed to get a lower bound of $n^{1-\epsilon}$.

Since the hardness result holds for any arbitrarily small constant ϵ, the next logical step to improve the lower bound is to show inapproximability results for non-constant ϵ. However, Håstad's proof of the existence of a probabilistic proof system with the needed properties is very long and complicated. This has, until now, hindered any advance in this direction, but with the appearance of the new ingenious construction of Samorodnitsky and Trevisan [22] new results are within reach.

In this paper, we show that it is indeed impossible to approximate Max Clique in polynomial time within $n^{1-\epsilon}$ where $\epsilon \in O(1/\sqrt{\log\log n})$, given that **NP** does not admit randomized algorithms with slightly super-polynomial expected running time. To do this we first ascertain that the reductions from probabilistic proof systems to Max Clique [10, 4, 23] work also in the case of a non-constant ϵ. This has the additional bonus of collecting in one place the various parts of the reduction, which were previously scattered in the literature. We also extend the previously published reductions to be able to use the construction of Samorodnitsky and Trevisan [22], which characterizes **NP** in terms of a probabilistic proof system with so called *non-perfect completeness*. To our knowledge, such reductions have not appeared explicitly in the literature before.

When we combine the new reductions with the probabilistic proof system of Samorodnitsky and Trevisan [22], we obtain the following concrete result regarding the approximability of Max Clique:

Theorem 8.1. *Unless* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{O(\log n(\log\log n)^{3/2})})$, *Max Clique on a graph with n vertices cannot be approximated within $n^{1-O(1/\sqrt{\log\log n})}$ in polynomial time.*

As a comparison, the best known polynomial time approximation algorithm [6], approximates Max Clique within $n^{1-O(\log\log n/\log n)}$.

Another problem—akin to Max Clique—is Min Chromatic Number, i.e., the problem of finding the minimum number of colors needed to properly

vertex color a graph. In fact, results regarding the approximability of Min Chromatic Number are very similar to the above results regarding the approximability of Max Clique. If the graph has n vertices, it is always possible to properly vertex color the graph using n colors and at least one color is always needed. This immediately gives an algorithm approximating Min Chromatic Number within n and Halldórsson [13] has shown that the problem can be approximated within

$$O\big(n(\log\log n)^2/\log^3 n\big) = n^{1-O(\log\log n/\log n)}$$

in polynomial time. On the negative side, Feige and Kilian [8, 11] have shown that Min Chromatic Number cannot be approximated within $n^{1-\epsilon}$, for any constant $\epsilon > 0$, unless **NP** = **ZPP**. In this paper, we combine a slight development of the construction used by Feige and Kilian [8, 11] with an extended version of the probabilistic proof system of Samorodnitsky and Trevisan [22] and obtain the following concrete result regarding the approximability of Min Chromatic Number:

Theorem 11.2. *Unless* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{O(\log n(\log\log n)^{3/2})})$*, it is impossible to approximate Min Chromatic Number on a graph with n vertices within $n^{1-O(1/\sqrt{\log\log n})}$ in polynomial time.*

The paper is outlined as follows: After some basic definitions in Sec. 2, we give a description of the developments of connections between probabilistic proof systems and the approximability of Max Clique and Min Chromatic Number in Sec. 3. This description leads to a proposed modification of a certain parameter of proof systems, the *amortized free bit complexity*, in Sec. 4. In Sec. 5 we establish that the previously known reductions from probabilistic proof systems to Max Clique can be used also when the amortized free bit complexity and the other parameters involved are not constants. Our tools are the original construction of Feige et al. [10] and the gap amplification technique of Zuckerman [23], and our goal is to obtain results of the form "If **NP** has a probabilistic proof system with certain parameters, then Max Clique cannot be approximated within some factor in polynomial time unless **NP** is contained in some class." for various values of the parameters involved. The results we obtain are implicit in the works of Zuckerman [23] and Bellare et al. [4], we repeat them here for the sake of completeness. In Sec. 6, we generalize the reductions to the case of non-perfect completeness. After a brief description of the new result of Samorodnitsky and Trevisan [22] in Sec. 7, we combine our reductions with their result to prove our lower bound on the approximability of Max Clique, Theorem 8.1, in Sec. 8. Then we turn to Min Chromatic Number. We first describe the reductions used to prove our lower bound in Sec. 9 and then show how the construction of Samorodnitsky and Trevisan can be modified to be usable for proving lower

bounds on the approximability of Min Chromatic Number in Sec. 10. Finally, we combine the above two sections to prove our lower bound on the approximability of Min Chromatic Number in Sec. 11.

2 Optimization and Approximation

In this paper, we study polynomial time approximation algorithms for some **NP**-hard optimization problems. To measure the efficiency of such an algorithm, we prove guarantees of the form that the algorithm always outputs a feasible solution with weight at most some factor from the weight of the optimal solution.

Definition 2.1. *Let P be a maximization problem. For an instance x of P let* $\mathrm{opt}(x)$ *be the optimal value. A solution y, with weight $w(x, y)$, is* c-approximate *if it is feasible and $w(x, y) \geq \mathrm{opt}(x)/c$.*

Definition 2.2. *Let P be a minimization problem. For an instance x of P let* $\mathrm{opt}(x)$ *be the optimal value. A solution y, with weight $w(x, y)$, is* c-approximate *if it is feasible and $w(x, y) \leq c \cdot \mathrm{opt}(x)$.*

Definition 2.3. *A* c-approximation algorithm *for an optimization problem is a polynomial time algorithm that for any instance x of the problem and any input y outputs a c-approximate solution.*

We use the wording *to approximate within c* as a synonym for *to compute a c-approximate solution*. Let us now formally define the problems we study in this paper.

Definition 2.4. Max Clique *is the following problem: Given a graph $G = (V, E)$ find the largest possible $C \subseteq V$ such that if v_1 and v_2 are vertices in C, then (v_1, v_2) is an edge in E.*

Definition 2.5. Max Independent Set *is the following problem: Given a graph $G = (V, E)$ find the largest possible $I \subseteq V$ such that if v_1 and v_2 are vertices in I, then (v_1, v_2) is not an edge in E.*

Obviously, Max Clique and Max Independent Set are equivalent problems since every clique in a graph G is an independent set in the complement graph $\overline{G}$ and vice versa.

Definition 2.6. Min Chromatic Number *is the following problem: Given a graph $G = (V, E)$ find the smallest possible integer k such that it is possible to partition V into k disjoint subsets with the property that if (v_1, v_2) is an edge in V, then v_1 and v_2 belong to different subsets in this partition.*

3 Preliminaries

A language L is in the class **NP** if there exists a polynomial time Turing machine M, with the properties that

1. For $x \in L$, there exists a proof π, of size polynomial in $|x|$, such that M accepts (x, π).
2. For $x \notin L$, M does not accept (x, π) for any proof π of size polynomial in $|x|$.

Arora and Safra [3] used a generalization of the above definition of **NP** to define the class $\mathbf{PCP}[r, q]$, consisting of a probabilistically checkable proof system (PCP) where the verifier has oracle access to the membership proof, is allowed to use $r(n)$ random bits and query $q(n)$ bits from the oracle.

Definition 3.1. *A probabilistic polynomial time Turing machine V with oracle access to π is an* (r, q)-restricted verifier *if it, for every oracle π and every input of size n, uses at most $r(n)$ random bits and queries at most $q(n)$ bits from the oracle. We denote by V^π the verifier V with the oracle π fixed.*

Definition 3.2. *A language L belongs to the class $\mathbf{PCP}[r, q]$ if there exists an (r, q)-restricted verifier V with the properties that*

1. *For $x \in L$, $\Pr_\rho[V^\pi$ accepts $(x, \rho)] = 1$ for some oracle π.*
2. *For $x \notin L$, $\Pr_\rho[V^\pi$ accepts $(x, \rho)] \leq 1/2$ for all oracles π.*

where ρ is the random string of length r.

3.1 Connection Between PCPs and Max Clique

The connection between the approximability of Max Clique and PCPs was first explored by Feige et al. [10], who showed that

$$\mathbf{NP} \subseteq \mathbf{PCP}[O(\log n \log\log n), O(\log n \log\log n)] \tag{1}$$

and used this characterization of **NP** and a reduction to show that unless $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{O(\log\log n)})$, Max Clique cannot be approximated within any constant in polynomial time.

The assumption on **NP** needed to prove hardness result on the approximability of Max Clique is closely related to the connection between the classes **NP** and $\mathbf{PCP}[r, q]$ for various values of r and q. This connection was the subject of intensive investigations leading to the following result of Arora et al. [2]:

Theorem 3.3. $\mathbf{NP} = \mathbf{PCP}[O(\log n), O(1)]$.

A consequence of this result is that the abovementioned assumptions in the proof of Feige et al. [10] could be weakened to $\mathbf{P} = \mathbf{NP}$. In fact, the lower bound was improved as well, to a factor N^{β}, for some constant β.

A technical tool in the proof of Feige et al. [10] is the construction of a graph $G_{V,x}$, corresponding to a verifier in some proof system and some input x.

Definition 3.4. *A list* $(a_1, \ldots, a_k)$ *is an* accepting view *for a random string of a verifier in a PCP if it contains answers to the queries made by the verifier on this random string and those answers make the verifier accept.*

Note that it is enough to put the answers in the list, the queries can be inferred from the random string and, in the case of adaptive verifiers, previous answers.

Definition 3.5. *Two accepting views are* consistent *if, whenever some bit is queried from the oracle, the answers are the same for both views.*

Definition 3.6. *From a verifier V and some input x, the graph $G_{V,x}$, the* FGLSS graph corresponding to V and x*, is defined as follows: Every vertex in $G_{V,x}$ corresponds to an accepting computation of the verifier. Thus a vertex can be described by a random string and an accepting view for that random string. Two vertices in $G_{V,x}$ are connected if they correspond to consistent accepting views.*

In the original construction, the number of vertices in $G_{V,x}$ was bounded by $2^{r(n)+q(n)}$, where $r(n)$ is the number of random bits used by the verifier and $q(n)$ is the number of bits the verifier queries from the oracle. Feige et al. suggest in their paper that the bound on the number of vertices in $G_{V,x}$ could be improved, and it was later recognized that the number of vertices can be bounded by $2^{r(n)+f(n)}$, where $f(n)$ is the *free bit complexity*.

Definition 3.7. *A verifier has* free bit complexity f *if the number of accepting views is at most 2^f for any outcome of the random bits tossed by the verifier.*

The free bit complexity is enough when only an upper bound on the number of vertices in $G_{V,x}$ is necessary. To give a lower bound, one needs to refine the above definition.

Definition 3.8. *A verifier has* average free bit complexity f_{av} *if the sum, over all random strings, of the number of accepting views is $2^{r+f_{av}}$.*

Lemma 3.9. *Suppose that we construct the graph $G_{V,x}$ from a verifier V and an input x as described in Definition 3.6. Also suppose that this verifier has free bit complexity f and average free bit complexity f_{av}. Then the number of vertices in $G_{V,x}$ graph is $2^{r+f_{av}} \leq 2^{r+f}$.*

Proof. Since every vertex in $G_{V,x}$ corresponds to an accepting computation of the verifier, the number of vertices in $G_{V,x}$ is exactly $2^{r+f_{\text{av}}}$. The last inequality follows since $f_{\text{av}} \leq f$. □

In fact, the verifiers considered in this paper all have $f_{\text{av}} = f$, and thus we use f instead of f_{av} everywhere.

Definition 3.10. *A language L belongs to the class $\mathbf{FPCP}_{c,s}[r, f]$ if there exists a verifier V with free bit complexity f that given an input x and oracle access to π tosses r independent random bits ρ and has the following properties:*

1. *For $x \in L$, $\Pr_\rho[V^\pi$ accepts $(x, \rho)] \geq c$ for some oracle π.*
2. *For $x \notin L$, $\Pr_\rho[V^\pi$ accepts $(x, \rho)] \leq s$ for all oracles π.*

We say that V has completeness c *and* soundness s.

To understand the intuition behind the free bit complexity of a proof system, it is perhaps best to study the behavior of a typical verifier in a typical proof system. Such a verifier first reads a number of bits, the free bits, from the oracle. From the information obtained from those bits and the random string, the verifier determines a number of bits, the non-free bits, that it should read next from the oracle and the values these bits should have in order for the verifier to accept. Finally, the verifier reads these bits from the oracle and check if they have the expected values.

Theorem 3.11. *Suppose that $L \in \mathbf{FPCP}_{c,s}[r, f]$. Let x be some instance of L, and construct the graph $G_{V,x}$ as in Definition 3.6. Then, there is a clique of size at least $c2^r$ in $G_{V,x}$ if $x \in L$, and there is no clique of size greater than $s2^r$ if $x \notin L$.*

Proof. First suppose that $x \in L$. Then there exists an oracle such that a fraction c of all random strings make the verifier accept. The computations corresponding to the same oracle are always consistent, and thus there exists a clique of size at least $c2^r$ in $G_{V,x}$.

Now suppose that $x \notin L$ and that there is a clique of size greater than $s2^r$ in $G_{V,x}$. Since vertices corresponding to the same random string can never represent consistent computations, the vertices in the clique all correspond to different random strings. Thus, we can use the vertices to form an oracle making the verifier accept with probability larger than s. This contradicts the assumption that the PCP has soundness s. □

Corollary 3.12. *Suppose that $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[O(\log n), f]$ for some constants c, s, and f. Then it is impossible to approximate Max Clique within c/s in polynomial time unless $\mathbf{P} = \mathbf{NP}$.*

Proof. Let L be some **NP**-complete language and x be some instance of L. Let B be some polynomial time algorithm approximating Max Clique within c/s.

The following algorithm decides L: Construct the graph $G_{V,x}$ corresponding to the instance x. Now run B on $G_{V,x}$. If the output from B is at least $s2^r$ where $r \in O(\log(n))$ is the number of random bits used by the verifier, accept x, otherwise reject.

By Lemma 3.9, the number of vertices in $G_{V,x}$ is 2^{r+f}. Since, in our case, r is logarithmic and f is a constant, the graph $G_{V,x}$ has polynomial size. Since B is a polynomial time algorithm, the above algorithm also runs in polynomial time. □

It is possible to improve on the above result by *gap amplification*. The simplest form of gap amplification is to simply run a constant number of independent runs of the verifier. If the verifier accepts for all these runs, we accept, otherwise we reject. This shows that, for any constant k,

$$\mathbf{FPCP}_{c,s}[r, f] \subseteq \mathbf{FPCP}_{c^k,s^k}[kr, kf], \tag{2}$$

for any functions c, s, r, and f, which strengthens Corollary 3.12 as follows:

Corollary 3.13. *Suppose that* $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[O(\log n), f]$ *for some constants* c, s, *and* f. *Then it is impossible to approximate Max Clique within any constant in polynomial time unless* $\mathbf{P} = \mathbf{NP}$.

The above procedure can improve the inapproximability result from a specific constant c/s to any constant, but to improve the inapproximability result from n^α to $n^{\alpha'}$ for some constants α and α', we have to use a more sophisticated form of gap amplification. Also, the concept of free bit complexity needs to be refined. To see why the above procedure fails in this case, suppose that we have some proof system which gives a graph $G_{V,x}$ with $n = 2^{r+f}$ vertices such that we can deduce that it is impossible to approximate Max Clique within n^α in polynomial time. Put another way, this particular proof system has $c/s = n^\alpha$. Now we try to apply the above gap amplification technique. Then we get a new graph $G_{V',x}$ with $2^{kr+kf} = n^k$ vertices and a new inapproximability factor $c^k/s^k = n^{k\alpha}$. Thus, we have failed to improve the lower bound. Obviously, it is not only the free bit complexity of a proof system that is important when it comes to proving lower bounds for Max Clique, but also the *gap*, the quotient of the soundness and the completeness. We see above that an exponential increase in the gap does not give us anything if the free bit complexity and the number of random bits increase linearly. Bellare and Sudan [5] recognized that the interesting parameter is $f/\log s^{-1}$ in the case of perfect completeness. This parameter was later named the *amortized free bit complexity* and denoted by $\bar{f}$. Note that the above gap amplification does not change $\bar{f}$. Two methods which do improve the lower bound in the case above by keeping down

the number of random bits needed to amplify the gap have appeared in the literature [4, 23], and both prove the same result: If every language in **NP** can be decided by a proof system with logarithmic randomness, perfect completeness, and amortized free bit complexity $\bar{f}$, then Max Clique cannot be approximated within $n^{1/(1+\bar{f})-\epsilon}$ in polynomial time, unless **NP** = **ZPP**. The constructions are valid for any constant $\bar{f}$ and some arbitrarily small constant $\epsilon > 0$.

3.2 Connection Between PCPs and Min Chromatic Number

Lund and Yannakakis [19] reduced Min Chromatic Number to Max Clique. This reduction, which did not preserve the ratio of approximation, was improved by Khanna, Linial, and Safra [17] and Bellare and Sudan [5]. As a final improvement, Fürer [12] constructed a randomized reduction showing that if Max Clique cannot be approximated within $n^{1/(1+f)}$, then Min Chromatic Number cannot be approximated within $n^{\min\{1/2,1/(1+2f)\}-o(1)}$.

In the reduction used above to prove strong lower bounds on the approximability of Max Clique, we used the graph $G_{V,x}$ defined in Definition 3.6 as a technical tool. Instead of first prove a strong lower bound on Max Clique and then reduce Min Chromatic Number to Max Clique, Feige and Kilian [8, 11] used the graph $G_{V,x}$—or to be more precise, the complement of this graph—to more directly prove a lower bound on the approximability of Min Chromatic Number. To describe their proof we need to introduce some properties of graphs.

Definition 3.14. *Given any graph G, we define*

$V(G)$ = *the set of vertices in G,*
$\alpha(G)$ = *the size of G's maximum independent set,*
$\chi(G)$ = *the minimum number of colors needed to vertex-color G.*

Obviously, $\alpha(G) \cdot \chi(G) \geq |V(G)|$; The vertices in $V(G)$ can be covered by a union of $\chi(G)$ independent sets, each of size at most $\alpha(G)$. The *fractional chromatic number* can be defined by considered arbitrary distributions on G's independent sets.

Definition 3.15. *The* fractional chromatic number $\chi_f(G)$ *is the smallest real k such that for some distribution D on G's independent sets, choosing I according to D covers any $v \in V(G)$ with probability at least $1/k$.*

The fractional chromatic number is related to the independence number and the number of vertices in the same way as the chromatic number; It can be shown that $\alpha(G) \cdot \chi_f(G) \geq |V(G)|$.

Furthermore, $\chi_f(G) \leq \chi(G)$; We can pick the independent sets corresponding to the color classes obtained from a proper vertex-coloring of G

with $\chi(G)$ colors and use the uniform distribution on these independent sets as the distribution D above. Lovász [18] has shown that

$$\chi_f(G) \geq \frac{\chi(G)}{1 + \ln \alpha(G)}.$$

This implies that a hardness result of the form n^β for the fractional chromatic number translates into a hardness result of the form $n^{\beta - O(\log\log n / \log n)}$ for the chromatic number. Thus, if we can prove a result similar to Theorem 3.11 for the fractional chromatic number, we more or less automatically obtain a result for Min Chromatic Number. Since $\alpha(G) \cdot \chi_f(G) \geq |V(G)|$, Theorem 3.11 can directly be used to prove the following:

Lemma 3.16. *Suppose that $L \in \mathbf{FPCP}_{c,s}[r, f]$. Let x be some instance of L, and construct the graph $G_{V,x}$ as in Definition 3.6. Then, $\overline{G_{V,x}}$ has fractional chromatic number at least $2^f/s$ if $x \notin L$.*

Proof. By Theorem 3.11, there is no clique of size greater than $s2^r$ in $G_{V,x}$, i.e., $\alpha(\overline{G_{V,x}}) \leq s2^r$, if $x \notin L$. By the connection between the fractional chromatic number and the independence number, this implies that

$$\chi_f(\overline{G_{V,x}}) \geq |V(\overline{G_{V,x}})|/s2^r = 2^f/s,$$

where the last equality follows from Lemma 3.9. □

Unfortunately we cannot get a bound for the case $x \in L$ by an equally simple manipulation of Theorem 3.11. Feige and Kilian used an extension of the class **FPCP** to define a new parameter of probabilistic proof systems and then proved that this parameter can be used to get a bound on the fractional chromatic number for the case when $x \in L$.

Definition 3.17. *A language L belongs to the class $\mathbf{RPCP}_{\rho,s}[r, f]$ if it belongs to the class $\mathbf{FPCP}_{c,s}[r, f]$ for some $c > s$ and, in addition to this, there is a probability distribution on the valid proofs with the property that if a valid proof of an input in L is selected according to this distribution, for any random string, every accepting view occurs with probability at least ρ. The parameter ρ is called the* covering radius *of the proof system.*

Note that $\rho \leq 1/2^f$ since the best we can hope for is that each of the 2^f accepting views occur with equal probability. Below, we will not be able to achieve the optimal ρ, but come within a constant factor. Using the covering radius, it is possible to prove a result similar to Theorem 3.11 for the fractional chromatic number [11].

Theorem 3.18. *Suppose that a language $L \in \mathbf{RPCP}_{\rho,s}[r, f]$. For an input x, let $G_{V,x}$ be the graph from Definition 3.6 corresponding to such an RPCP for L. Then*

$$x \in L \implies \chi_f(\overline{G_{V,x}}) \leq 1/\rho,$$
$$x \notin L \implies \chi_f(\overline{G_{V,x}}) \geq 2^f/s.$$

Proof. By Theorem 3.11, there is no clique of size greater than $s2^r$ in $G_{V,x}$, i.e., $\alpha(\overline{G_{V,x}}) \leq s2^r$, if $x \notin L$. By Lemma 9.1, this implies that

$$\chi_f(\overline{G_{V,x}}) \geq |V(\overline{G_{V,x}})|/s2^r = 2^f/s,$$

where the last equality follows from Lemma 3.9.

For the case $x \in L$ we proceed as follows: Let π be a proof making the verifier accept. By the proof of Theorem 3.11, this proof corresponds to a clique in $G_{V,x}$. This implies that every proof making the verifier accept corresponds to a clique in $G_{V,x}$, i.e., to an independent set in $\overline{G_{V,x}}$.

By Definition 3.17, there exists a distribution on the valid proofs such that every accepting view occurs with probability at least ρ. Suppose that the prover selects a proof according to this distribution. This induces a distribution on the independent sets in $\overline{G_{V,x}}$ with the property that any given vertex in $\overline{G_{V,x}}$ is covered by the independent set with probability at least ρ. By the Definition 3.15, this implies that $\chi_f(\overline{G_{V,x}}) \leq 1/\rho$. □

Note that Theorem 3.18 gives a hardness result for the fractional chromatic number very similar to the hardness result for Max Clique obtained in Corollary 3.12.

Corollary 3.19. *Suppose that* $\mathbf{NP} \subseteq \mathbf{RPCP}_{\rho,s}[r, f]$ *where* $r \in O(\log n)$ *and the other parameters are constants. Then it is impossible to approximate the fractional chromatic number within* $\rho 2^f/s$ *in polynomial time unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Let L be some **NP**-complete language and x be some instance of L. Let B be some polynomial time algorithm approximating Min Chromatic Number within $\rho 2^f/s$.

The following algorithm decides L: Construct the graph $G_{V,x}$ corresponding to the instance x. Now run B on $\overline{G_{V,x}}$. If B determines that $\overline{G_{V,x}}$ has a proper vertex coloring using more than $2^f/s$ colors, accept x, otherwise reject.

Since the number of random bits used by the verifier is logarithmic and the average free bit complexity is a constant, the graph $G_{V,x}$ has polynomial size. Since B is a polynomial time algorithm, the above algorithm also runs in polynomial time. □

At first, it may seem strange that the completeness does not appear in the above result. However, the completeness does appear, although very implicitly, in the covering radius. Intuitively, the quantity $\rho 2^f$ has the same role in this result as the completeness in the lower bound on the approximability of Max Clique.

To get hardness result of the form n^β for Max Clique, gap amplification is used to boost a constant inapproximability threshold. In the case of Min Chromatic Number, Feige and Kilian [8, 11] used *randomized graph*

products. We describe the details of this construction in Sec. 9, for now we just mention that the construction gives the following hardness result: Suppose that $\mathbf{NP} \subseteq \mathbf{RPCP}_{\rho,s}[r, f]$. Let

$$\beta = 1 - \frac{\log \rho^{-1}}{f + \log s^{-1}} - \frac{\log D + r + f}{D(f + \log s^{-1})},$$

where D is a parameter that can be selected to boost the inapproximability result. Then it is impossible to approximate the fractional chromatic number in a graph with $N \leq 2^{D(f+\log s^{-1})}$ vertices within N^{β} in polynomial time unless $L \in \mathbf{ZPTIME}(2^{\Theta(r+D(f+\log s^{-1}))})$.

4 A New Amortized Free Bit Complexity

For the case of non-perfect completeness, Bellare et al. [4] define the amortized free bit complexity as $f/\log(c/s)$. In this paper, we propose that this definition should be modified.

Definition 4.1. *The* amortized free bit complexity *for a PCP with free bit complexity f, completeness c and soundness s is*

$$\bar{f} = \frac{f + \log c^{-1}}{\log(c/s)}. \tag{3}$$

Note that both this definition and the previous one reduce to $f/\log s^{-1}$ in the case of perfect completeness, i.e., when $c = 1$. Note also that the above gap amplification does not change the amortized free bit complexity, neither with the original definition nor with our proposed modification of the definition. However, our proposed definition is robust also with respect to the following: Suppose that we modify the verifier in such a way that it guesses the value of the first free bit. This lowers the free bit complexity by one, and halves the completeness and the soundness of the test. With our proposed definition, the amortized free bit complexity does not change, while it decreases with the definition of Bellare et al. [4]. In the case of perfect completeness, the lower bound on the approximability increases as the amortized free bit complexity decreases. This makes it dubious to have a definition in the general case that allows the free bit complexity to be lowered by a process as the above. Using our proposed definition of the free bit complexity, we can prove the following results:

Theorem 5.7. *If $\mathbf{NP} \subseteq \mathbf{FPCP}_{1,s}[r, f]$, then, for any $R > r$, Max Clique in a graph with N vertices cannot be approximated within $N^{1/(1+\bar{f})-r/R}$ in polynomial time unless $\mathbf{NP} \subseteq \mathbf{coRTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$.*

Corollary 5.9. *If $\mathbf{NP} \subseteq \mathbf{FPCP}_{1,s}[r, f]$ for some $r \in \Omega(\log n)$, then, for any $R > r$, Max Clique in a graph with $N = 2^{R+(R+2)\bar{f}}$ vertices cannot be approximated within $N^{1/(1+\bar{f})-r/R}$ in polynomial time unless $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$.*

Theorem 6.4. *If* $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r,f]$ *for some* $r \in \Omega(\log n)$*, then, for any* $R > r$ *such that* $c^d 2^R/2 > 2^r$*, where* $d = (R+2)/\log s^{-1}$*, Max Clique in a graph with* N *vertices cannot be approximated within* $N^{1/(1+\bar{f})-(r+3)/(R+2)}$ *in polynomial time unless* $\mathbf{NP} \subseteq \mathbf{BPTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$.

Note that we in our applications choose R such that r/R is small, thus essentially reducing the above lower bounds to $N^{1/(1+\bar{f})}$. If we want to get a reduction without two-sided error in the case of non-perfect completeness, we do not quite achieve the above. Instead, the interesting parameter becomes

$$F_\nu = \frac{1+\nu f}{\nu \log(\nu/s) + (1-\nu)\log(1-\nu)}. \tag{4}$$

where ν is a parameter which is arbitrarily close to c. We can then prove the following theorem

Theorem 6.16. *Suppose that* $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r,f]$*. Let* ν *be any constant such that* $s < \nu < c$*. Let* $h = (c-\nu)/(1-\nu)$*. Then, for any* $R > r$*, it is impossible to approximate Max Clique in a graph with* $N = 2^{R+(R+2)F_\nu}$ *vertices within* $N^{1/(1+F_\nu)-r/R-(\log h^{-1})/R}$ *in polynomial time unless*

$$\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{\Theta(R+F_\nu+RF_\nu)}).$$

5 Zuckerman's Construction

In this section we show how to obtain strong inapproximability results for Max Clique, given that we have at hand a theorem saying that $\mathbf{NP} \subseteq \mathbf{FPCP}_{1,s}[r,f]$. One way to lower the soundness in a PCP is to run several independent runs of the verifier. As discussed above, this is not enough in our case. Instead, Zuckerman [23] used a probabilistic construction to in some sense recycle randomness. His construction uses R random bits to simulate a number of runs each using r random bits. In his paper, the parameters s and f were constants and r logarithmic, but we make no such assumptions in this section. Instead we establish that Zuckerman's construction works even in the general case, and we prove a general theorem where the parameters involved appear.

The idea is to pick a random bipartite graph with 2^R vertices on the left side, 2^r vertices on the right side, and where the degree of the vertices on the left side is d. From this graph construct a new probabilistic machine, which uses the R random bits to select a vertex v on the left side, and then runs the verifier in the proof system d times, letting the verifier's random bits be determined by the vertices adjacent to v. If we obtain only accepting runs, we accept our input, otherwise we reject. Obviously, it may happen that we incorrectly accept an input which is not in the language. Since our original proof system for **NP** has soundness s, an s-fraction of all r-bit random strings may cause the verifier to accept. Thus, we must bound the

probability that a large subset of the vertices on the left side has only such neighbors.

Definition 5.1. *A bipartite graph $H = (E, V_1 \cup V_2)$ is a* (d, n_1, n_2)-disperser *if all vertices in V_1 have degree d, all vertices in V_2 have expected degree $d|V_1|/|V_2|$, and no sets $S_1 \subseteq V_1$ and $S_2 \subseteq V_2$ of cardinality n_1 and n_2, respectively, have the property that every vertex in S_1 is connected only with vertices in S_2.*

Lemma 5.2. *Let H be a random bipartite graph $H = (E, V_1 \cup V_2)$ where $|V_1| = 2^R$, $|V_2| = 2^r < 2^R$, and the edge set $E \subseteq V_1 \times V_2$ is chosen as follows: For each $v \in V_1$, pick d neighbors independently and uniformly among the vertices in V_2. We allow multiple edges in H. If $d = (R+2)/\log s^{-1}$ for some $s < 1$, the graph H is a $(d, 2^r, s2^r)$-disperser with probability at least $1 - 2^{-2^r}$.*

Proof. Let S_1 be a subset of cardinality 2^r of the vertices in V_1 and S_2 be a subset of cardinality $s2^r$ of the vertices in V_2. Call a vertex $v \in V_1$ S_2-*bad* if all the neighbors of v are in S_2. Let A_{S_1,S_2} be the event that all $v \in S_1$ are S_2-bad. Since the neighbors of every $v \in V_1$ are chosen independently and uniformly,

$$\Pr[A_{S_1,S_2}] = s^{d|S_1|} = s^{d2^r}$$

for fixed S_1 and S_2. Thus,

$$\Pr\left[\bigcup_{S_1}\bigcup_{S_2} A_{S_1,S_2}\right] \leq \sum_{S_1}\sum_{S_2} \Pr[A_{S_1,S_2}] \leq \binom{2^r}{s2^r}\binom{2^R}{2^r} s^{d2^r}.$$

This is at most $2^{(R+1-d\log s^{-1})2^r} = 2^{-2^r}$ since $d = (R+2)/\log s^{-1}$. □

The above lemma means that a graph H constructed as in the lemma with high probability has the property that for any PCP which uses r random bits and has soundness s, few of the vertices in V_1 has only accepting neighbors. Let us now formalize this property.

Definition 5.3. *Given a verifier V from some proof system and an input x we first pick a random bipartite graph H as in Lemma 5.2 with $V_1 = \{0,1\}^R$ and $V_2 = \{0,1\}^r$. We then construct an* acceptance tester*, which is a probabilistic machine $A(H, V, x)$ behaving as follows on oracle π: It picks a random $v \in V_1$, and determines the $d = (R+2)/\log s^{-1}$ neighbors $u_1, \ldots, u_d$ of v in V_2. It simulates V on x and π with random strings $u_1, \ldots, u_d$, and accepts if all simulations accept. We write $A^\pi(H, V, x)$ for $A(H, V, x)$ with the oracle π fixed. We say that a random string σ is an* accepting string *for the tester $A^\pi(H, V, x)$ if $A^\pi(H, V, x)$ accepts on oracle when the vertex $v \in V_1$ corresponding to σ is the random vertex picked.*

Lemma 5.4. *Suppose that $L \in \mathbf{FPCP}_{1,s}[r, f]$, and that V is the verifier in that proof system. The acceptance tester $A(H, V, x)$ as described in Definition 5.3 has the following properties given some input x:*

1. *If $x \in L$, then there is an oracle π such that all R-bit strings are accepting strings for $A^\pi(H, V, x)$.*
2. *If $x \notin L$, then the following holds with probability $1 - 2^{-2^r}$ over the choice of H: For all oracles π at most 2^r of all R-bit strings are accepting strings for $A^\pi(H, V, x)$.*

Proof. We first assume that $x \in L$. Then there is an oracle π such that all r-bit string make the verifier V accept on x and π. This implies, by Definition 5.3, that all R-bit strings are accepting strings for $A^\pi(H, V, x)$.

Now we study the case $x \notin L$. Lemma 5.2 implies that with probability $1 - 2^{-2^r}$, the graph H used to construct the acceptance tester has the property that there are no sets U and S of cardinality 2^r and $s2^r$, respectively, such that all vertices in U are connected only to vertices in S. Let us now suppose that this is the case. For any oracle π, let S^π be the set of all r-bit strings that make the verifier V accept on π. Since the proof system has soundness s, the set S^π can contain at most $s2^r$ strings. Thus, at most 2^r of all R-bit strings can be accepting strings for $A^\pi(H, V, x)$. □

Note that once H is fixed, we get a new PCP with a new verifier V_H that uses R random bits and runs d repetitions of the verifier V from the original PCP. We now construct a graph $G_{V_H,x}$ as in Definition 3.6. Then we prove that there is a connection between the size of the maximum clique in this graph and the acceptance probabilities of the acceptance tester. Finally, we use this connection to show that an algorithm approximating Max Clique within $n^{1/(1+\bar{f})-r/R}$ can probabilistically decide a language in $\mathbf{FPCP}_{1,s}[r, f]$ with one-sided error.

Lemma 5.5. *Suppose that $L \in \mathbf{FPCP}_{1,s}[r, f]$ and that V is the verifier in the above proof system. Given some input x, pick an acceptance tester $A(H, V, x)$ as in Definition 5.3 and construct the graph $G_{V_H,x}$ as in Definition 3.6. Then the following holds:*

1. *If $x \in L$, there is a clique of size 2^R in $G_{V_H,x}$.*
2. *If $x \notin L$, the following holds with probability $1 - 2^{-2^r}$ over the choice of H: There is no clique larger than 2^r in $G_{V_H,x}$.*

Proof. This follows from Lemma 5.4 together with a proof very similar to that of Theorem 3.11. □

Lemma 5.6. *Suppose that $L \in \mathbf{FPCP}_{1,s}[r, f]$, where $r \in \Omega(\log n)$, and that there is an algorithm B which approximates Max Clique within 2^{R-r},*

in a graph G with $N = 2^{R+(R+2)\bar{f}}$ vertices. Then $L \in \mathbf{coRTIME}(T(N) + p(N))$, where $T(N)$ is the running time of the algorithm B on a graph with N vertices and p is some polynomial.

Proof. Given the verifier V from the proof system and an input x, we pick an acceptance tester $A(H, V, x)$ and construct the graph $G_{V_H,x}$ as described in Definition 3.6. Since the acceptance tester uses R random bits and has free bit complexity $df = (R+2)\bar{f}$, the number of vertices in $G_{V_H,x}$ is at most $2^{R+(R+2)\bar{f}}$. Run B on $G_{V_H,x}$. If the output from B is at least 2^r we accept, otherwise we reject.

By Lemma 5.5, the above algorithm never rejects if $x \in L$, and if $x \notin L$ the algorithm accepts with probability at most 2^{-2^r}. The running time of the algorithm is $T(N)+p(N)$ for some polynomial p, so $L \in \mathbf{coRTIME}(T(N)+p(N))$. □

Now we are ready to prove a lower bound on the approximability of Max Clique.

Theorem 5.7. *If $\mathbf{NP} \subseteq \mathbf{FPCP}_{1,s}[r, f]$, then, for any $R > r$, Max Clique in a graph with $N = 2^{R+(R+2)\bar{f}}$ vertices cannot be approximated within $N^{1/(1+\bar{f})-r/R}$ in polynomial time unless $\mathbf{NP} \subseteq \mathbf{coRTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$, where $\bar{f}$ is the amortized free bit complexity as per Definition 4.1.*

Proof. In Lemma 5.6, let the language L be some **NP**-complete language and the algorithm B be some algorithm with polynomial running time. Then, the lemma implies that clique in a graph with $N = 2^{R+(R+2)\bar{f}}$ vertices cannot be approximated within 2^{R-r} in polynomial time, unless

$$\mathbf{NP} \subseteq \mathbf{coRTIME}\big(2^{\Theta(R+\bar{f}+R\bar{f})}\big).$$

To express the approximation ratio in terms of N, we note that

$$2^{R-r} = N^{(R-r)/(R+(R+2)\bar{f})}.$$

The exponent is at least

$$\frac{(R+2)-(r+2)}{(R+2)(1+\bar{f})} > \frac{1}{1+\bar{f}} - \frac{r+2}{R+2},$$

since $\bar{f} > 0$. Since the last term above is at most r/R when $R > r$. $2^{R-r} > N^{1/(1+\bar{f})-r/R}$. □

To show a hardness result under the weaker assumption that

$$\mathbf{NP} \not\subseteq \mathbf{ZPTIME}\big(2^{\Theta(R+\bar{f}+R\bar{f})}\big),$$

we make use of the following relation between the classes **NP**, **coRTIME**$(\cdot)$, and **ZPTIME**$(\cdot)$.

Lemma 5.8. *If* $\mathbf{NP} \subseteq \mathbf{coRTIME}(T(n))$ *for some function* $T(n)$*, then* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(O(n \log n)T(n))$.

Proof. We show that

$$\text{Sat} \in \mathbf{coRTIME}(T(n)) \implies \text{Sat} \in \mathbf{ZPTIME}(O(n \log n)T(n)). \quad (5)$$

If Sat is in $\mathbf{coRTIME}(T(n))$, there exists a randomized algorithm A that probabilistically decides Sat in time $T(n)$. For an instance in the language, the algorithm never rejects, but for an instance not in the language, the algorithms accepts with probability $1/2$. We can lower this probability to $1/n^2$ by repeating the algorithm $2 \log n$ times. Let us call this new algorithm A'. The running time of A' is $(2 \log n)T(n)$.

For a formula ϕ, the following randomized procedure with high probability either rejects ϕ if it is unsatisfiable or finds a satisfying assignment to ϕ if it is satisfiable: Run A' on the formula ϕ. If A' rejects, we know that ϕ cannot be satisfiable. If A' accepts, we assume that ϕ is satisfiable and try to deduce a satisfying assignment. Suppose that ϕ depends on the variables $x_1, \ldots, x_n$. Then we try to set x_1 to true in ϕ, and run A' on the resulting formula. If A' accepts, we keep x_1 true, otherwise we set x_1 to false. This process is repeated to obtain an assignment to all n variables. If A' never gave us a false positive during this process, we end up with a satisfying assignment. The probability of this event is at least $1 - 1/n$, since A' accepts inputs which are not in the language with probability at most $1/n^2$.

To sum up, the above procedure behaves as follows: For satisfiable formulas, it produces a satisfying assignment in time $O(n \log n)T(n)$ with probability $1 - 1/n$. For unsatisfiable formulas, it rejects in time $O(\log n)T(n)$ with probability $1 - 1/n^2$. Thus, it obtains a definitive answer in expected time $O(n \log n)T(n)$, both for satisfiable and unsatisfiable formulas. □

By combining Theorem 5.7 and Lemma 5.8, we obtain the following corollary:

Corollary 5.9. *If* $\mathbf{NP} \subseteq \mathbf{FPCP}_{1,s}[r, f]$ *for some* $r \in \Omega(\log n)$*, then, for any* $R > r$*, Max Clique in a graph with* $N = 2^{R+(R+2)\bar{f}}$ *vertices cannot be approximated within* $N^{1/(1+\bar{f})-r/R}$ *in polynomial time unless* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$.

In the case where $\bar{f}$ is some constant and $r \in O(\log n)$, this reduces to the well known theorem that Max Clique cannot be approximated within $n^{1/(1+\bar{f})-\epsilon}$, for any constant $\epsilon > 0$, unless $\mathbf{NP} = \mathbf{ZPP}$. To see this, just choose $R = r/\epsilon$ above.

If we could come up with a way to construct dispersers deterministically, we would get a hardness result for Max Clique under the assumption that $\mathbf{NP} \not\subseteq \mathbf{DTIME}(\cdot)$, but for now we get a hardness result under the assumption that $\mathbf{NP} \not\subseteq \mathbf{ZPTIME}(\cdot)$. But once we have this assumption we can prove a stronger version of Corollary 5.9.

Lemma 5.10. *Suppose that $L \in \mathbf{FPCP}_{1,s}[r, f]$, where $r \in \Omega(\log n)$, and that there is a probabilistic algorithm B that approximates clique within 2^{R-r} in a graph with $N = 2^{R+(R+2)\bar{f}}$ without making any errors. Then $L \in \mathbf{coRTIME}(4T(N)+p(N))$, where $T(N)$ is the expected running time of the algorithm B on a graph with N vertices and p is some polynomial.*

Proof. Given the verifier V from the proof system and an input x, we pick an acceptance tester $A(H, V, x)$ and construct the graph $G_{V_H,x}$ as described in Definition 3.6. The number of vertices in this graph is at most $2^{R+(R+2)\bar{f}}$. Run B on $G_{V_H,x}$ for $4T(n)$ time steps. If B has not terminated within this time—this happens with probability at most $1/4$ by Markov's inequality—we accept x. Otherwise we proceed as follows: If the value output by B is at least 2^r we accept, otherwise we reject.

By Lemma 5.5, the above algorithm never rejects if $x \in L$, and if $x \notin L$ the algorithm accepts with probability at most $1/4 + 2^{-2^r}$. The running time of the algorithm is $4T(N) + p(N)$ for some polynomial p, so $L \in \mathbf{coRTIME}(4T(N) + p(N))$. □

We now combine the above lemma with a slight restatement of Theorem 5.7 to obtain a slightly stronger hardness result.

Theorem 5.11. *If $\mathbf{NP} \subseteq \mathbf{FPCP}_{1,s}[r, f]$ and $n \log n = 2^{O(R+\bar{f}+R\bar{f})}$, then, for any $r \in \Omega(\log n)$ and any $R > r$, it is impossible to approximate Max Clique in a graph with N vertices within $N^{1/(1+\bar{f})-r/R}$ by algorithms with expected polynomial running time unless $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$.*

Proof. By Lemma 5.8, it suffices to prove that it is hard to approximate Max Clique unless $\mathbf{NP} \subseteq \mathbf{coRTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$. In Lemma 5.10, let the language L be some **NP**-complete language and the algorithm B be some algorithm with expected polynomial running time. Then, the lemma implies that Max Clique in a graph with $N = 2^{R+(R+2)\bar{f}}$ vertices cannot be approximated within 2^{R-r} in expected polynomial time, unless $\mathbf{NP} \subseteq \mathbf{coRTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$. As in the proof of Theorem 5.7, the approximation ratio is at least $N^{1/(1+\bar{f})-r/R}$. □

6 Our Extensions

In this section we show how to generalize the constructions from Sec. 6 to the case of non-perfect completeness (that is $c < 1$). We assume from now on that we have at hand a theorem saying that $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r, f]$, and try to deduce inapproximability results from this characterization. If we allow two-sided error in the reduction, the methods from Sec. 5 can be generalized as described in Sec. 6.1. The major new complication arises when we want to prove hardness results under the assumption that $\mathbf{NP} \not\subseteq \mathbf{ZPTIME}(\cdot)$,

which requires the reduction to produce an algorithm deciding a language in **NP** with one-sided error. A method achieving this is explored in Sec. 6.2.

6.1 Two-Sided Error

In this section, we want to use a theorem saying that $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r, f]$, to deduce inapproximability under the assumption that **NP** does not admit slightly superpolynomial algorithms with two-sided error. With the same approach as in the case of perfect completeness, we get the following analogue to Lemma 5.4:

Lemma 6.1. *Suppose that $L \in \mathbf{FPCP}_{c,s}[r, f]$, and that V is the verifier in that proof system. If we construct an acceptance tester $A(H, V, x)$ from V and some input x as described in Definition 5.3, it has the following properties:*

1. *If $x \in L$, then with probability $1-e^{-c^d 2^R/8}$ there is an oracle π such that at least $c^d 2^R/2$ of all R-bit strings are accepting strings for $A^\pi(H, V, x)$.*
2. *If $x \notin L$, then with probability $1 - 2^{-2^r}$, for all oracles π at most 2^r of all R-bit strings are accepting strings for $A^\pi(H, V, x)$.*

The probabilities are over the choice of the random bipartite graph H.

Proof. We first assume that $x \in L$. Then there is an oracle π such that a fraction c of the r-bit strings makes V^π accept x. This implies, by the construction of the graph $H = (E, V_1, V_2)$ in Lemma 5.2 and by Definition 5.3 that the probability (over the choice of H) that a vertex $v \in V_1$ corresponds to an accepting string for $A^\pi(H, V, x)$ is c^d. Let X_v be the indicator variable for this event, and let $Y = \sum_{v \in V_1} X_v$. Now, a standard Chernoff bound [20, Chapter 4] implies that

$$\Pr[Y \leq c^d 2^R/2] \leq e^{-c^d 2^R/8}.$$

The case $x \notin L$ is exactly as in the proof of Lemma 5.4. □

The following lemma is similar to Lemma 5.5.

Lemma 6.2. *Suppose that $L \in \mathbf{FPCP}_{c,s}[r, f]$ and that V is the verifier in the above proof system. Given some input x, pick an acceptance tester $A(H, V, x)$ as in Definition 5.3 and construct the graph $G_{V_H,x}$ as in Definition 3.6. Then the following holds:*

1. *If $x \in L$, the following holds with probability $1 - e^{-c^d 2^R/8}$ over the choice of H: There is a clique of size $c^d 2^R/2$ in $G_{V_H,x}$.*
2. *If $x \notin L$, the following holds with probability $1 - 2^{-2^r}$ over the choice of H: There is no clique larger than 2^r in $G_{V_H,x}$.*

The reduction is slightly different from Lemma 5.6, since the size of clique in the case $x \in L$ may be less than c^d.

Lemma 6.3. *Suppose that $L \in \mathbf{FPCP}_{c,s}[r, f]$, that $r \in \Omega(\log n)$, that R and r are such that $c^d 2^R/2 > 2^r$, where $d = (R+2)/\log s^{-1}$, and that there is an algorithm B which can approximate Max Clique within $c^d 2^{R-r}/2$ in a graph G with $N = 2^{R+(R+2)f/\log s^{-1}}$ vertices. Then $L \in \mathbf{BPTIME}(T(N) + p(N))$, where $T(N)$ is the running time of the algorithm B on a graph with N vertices and p is some polynomial.*

Proof. Given the verifier V from the proof system and an input x, we pick an acceptance tester $A(H, V, x)$ and construct the graph $G_{V_H,x}$ as described in Definition 3.6. Since the acceptance tester uses R random bits and has at most $2^{Df} = 2^{(R+2)f/\log s^{-1}}$ accepting computations, the number of vertices in $G_{V_H,x}$ is at most $2^{R+(R+2)f/\log s^{-1}}$. Run B on $G_{V_H,x}$. If B returns a value which is at most 2^r, reject, otherwise accept.

By Lemma 5.5, if $x \in L$ the above algorithm rejects with probability at most $e^{-c^d 2^R/8}$. This is at most $1/3$ provided that $8 \ln 3 < c^d 2^R$. Since we have assumed that $r \in \Omega(\log n)$ and $c^d 2^R > 2^{r+1}$, this holds when n is large enough. If $x \notin L$ the algorithm accepts with probability at most $2^{-2^r} < 1/3$. The running time of the algorithm is $T(N) + p(N)$ for some polynomial p, so $L \in \mathbf{BPTIME}(T(N) + p(N))$. □

We get the following analogue to Theorem 5.7:

Theorem 6.4. *Suppose that $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r, f]$ for some $r \in \Omega(\log n)$. Then, for any and $R > r$ such that $c^d 2^R/2 > 2^r$, where $d = (R+2)/\log s^{-1}$, the Max Clique problem in a graph with N vertices cannot be approximated within $N^{1/(1+\bar{f})-(r+3)/(R+2)}$ in polynomial time unless*

$$\mathbf{NP} \subseteq \mathbf{BPTIME}\big(2^{\Theta(R+\bar{f}+R\bar{f})}\big),$$

where $\bar{f}$ is our proposed amortized free bit complexity from Definition 4.1.

Proof. In Lemma 6.3, let the language L be some **NP**-complete language and the algorithm B be some algorithm with polynomial running time. Then, the lemma implies that Max Clique in a graph with $N = 2^{R+(R+2)f/\log s^{-1}}$ vertices cannot be approximated within $c^d 2^{R-r}/2$ in polynomial time, unless $\mathbf{NP} \subseteq \mathbf{BPTIME}(2^{\Theta(R+\bar{f}+R\bar{f})})$. To express the approximation ratio in terms of N, we note that

$$c^d 2^{R-r}/2 = N^{(R-d\log c^{-1}-r-1)/(R+(R+2)f/\log s^{-1})}$$

By the definition of d, the exponent is

$$\frac{R - (R+2)\frac{\log c^{-1}}{\log s^{-1}} - r - 1}{R + (R+2)f/\log s^{-1}},$$

which is at least

$$\frac{(R+2)\big(1-\frac{\log c^{-1}}{\log s^{-1}}\big)-(r+3)}{(R+2)(1+f/\log s^{-1})}\,.$$

This is at least

$$\frac{\log c-\log s}{f-\log s}-\frac{r+3}{R+2}\,.$$

The first term is equal to

$$\frac{\log c-\log s}{\log c-\log s+f+\log c^{-1}}=\frac{1}{1+\bar{f}}$$

if we use our proposed definition $\bar{f}=(f+\log c^{-1})/\log(c/s)$ of the amortized free bit complexity. To sum up, $c^d 2^{R-r} > N^{1/(1+\bar{f})-(r+3)/(R+2)}$. □

In the case where $\bar{f}$ is some constant and $r \in O(\log n)$, this reduces to the theorem that Max Clique cannot be approximated within $n^{1/(1+\bar{f})-\epsilon}$, for any constant $\epsilon > 0$, unless **NP** = **BPP**. To see this, just choose $R = (r+3)/\epsilon - 2$ above.

6.2 One-Sided Error

We want to use a theorem saying that $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r,f]$, to deduce inapproximability under the assumption that **NP** does not admit probabilistic algorithms that have slightly superpolynomial expected running time and never make any mistakes. By Lemma 5.8, it suffices to prove if Max Clique can be approximated within a certain factor, then **NP** admits a probabilistic algorithm with slightly superpolynomial running time and a *one sided error*. We will do this this by replacing the acceptance tester of Section 5 and Section 6.1 with a *threshold tester*, defined in this section. The threshold tester will be guaranteed large acceptance probability in the case when $x \in L$. We do this by making the condition for acceptance to be that a large fraction of all runs of the verifier V accepts, rather than requiring all runs to be accepting. The threshold tester will, as the acceptance tester, make use of a bipartite graph H, but we want the graph used in the threshold tester to have sligthly different properties.

Definition 6.5. *A bipartite graph $H = (E, V_1 \cup V_2)$ is a* regular (d, n_1, n_2, ν)-disperser *if all vertices in V_1 have degree d, all vertices in V_2 have degree $d|V_1|/|V_2|$, and no sets $S_1 \subseteq V_1$ and $S_2 \subseteq V_2$ of cardinality n_1 and n_2, respectively, have the property that for every vertex in S_1 at least a fraction ν of its neighbors are vertices in S_2.*

Before proving that there exists a probabilistic construction that produces certain regular (d, n_1, n_2, ν)-dispersers with high probability, we prove that such dispersers have an additional useful property.

Lemma 6.6. *Let H be a regular (d, n_1, n_2, ν)-disperser. For any set S_2 containing a fraction $1-c$ of the vertices in V_2, at most a fraction $(1-c)/(1-\nu)$ of the vertices in V_1 has the property that at least a fraction $1-\nu$ of their neighbors in S_2.*

Proof. Let S_2 be as in the statement of the lemma. Then there are at most $(1-c)|V_2| \cdot d|V_1|/|V_2| = (1-c)d|V_1|$ edges $E(S_2)$ with one vertex in S_2. If v is any vertex with at least a fraction $(1-\nu)$ of its neighbors in S_2, then at least $(1-\nu)d$ of the edges $E(S_2)$ must have v as one vertex—the other vertex is in S_2. Thus there can be at most

$$\frac{(1-c)d|V_1|}{(1-\nu)d} = \frac{1-c}{1-\nu}|V_1|$$

such vertices v. □

The probabilistic construction producing regular (d, n_1, n_2, ν)-dispersers is very similar to the one producing (d, n_1, n_2)-dispersers in Lemma 5.2, but the choice of the parameter d is different. To prove that the construction is correct with high probability we also need to rely on the following estimate of certain binomial coefficients:

Lemma 6.7. *Let $0 < \alpha < 1$. Then, for any non-negative integer n such that $n\alpha(1-\alpha) > 169/72\pi$,*

$$\log \binom{n}{\alpha n} \leq n\big(\alpha \log \alpha^{-1} + (1-\alpha)\log(1-\alpha)^{-1}\big). \tag{6}$$

Proof. By Stirling's formula, $k! = k^k e^{-k}\sqrt{2\pi k}(1+\theta)$, where $0 \leq \theta \leq 1/12$. If we use this estimate and the definition of the binomial coefficient, we obtain the bound

$$\begin{aligned}\binom{n}{\alpha n} &= \frac{\sqrt{2\pi n}(1+\theta_1)}{\alpha^{\alpha n}\sqrt{2\pi\alpha n}(1+\theta_2)(1-\alpha)^{(1-\alpha)n}\sqrt{2\pi(1-\alpha)n}(1+\theta_3)} \\ &= \big(\alpha^{-\alpha}(1-\alpha)^{-(1-\alpha)}\big)^n \cdot \frac{1}{\sqrt{n}} \cdot \frac{1}{\sqrt{2\pi\alpha(1-\alpha)}} \cdot \frac{(1+\theta_1)}{(1+\theta_2)(1+\theta_3)}.\end{aligned}$$

Since we assumed that $n\alpha(1-\alpha) > 169/72\pi$, the last three factors above multiply to something less than one. □

Lemma 6.8. *Let H be a random bipartite graph $H = (E, V_1 \cup V_2)$ where $|V_1| = 2^R$, $|V_2| = 2^r < 2^R$, and the edge set $E \subseteq V_1 \times V_2$ is chosen uniformly at random from the space of all edge sets with the property that the degree of every vertex in V_1 is d and the degree of every vertex in V_2 is $d2^{R-r}$. If*

$$d = \frac{R+2}{\nu \log(\nu/s) + (1-\nu)\log(1-\nu)}$$

for some s and ν such that $2^{r-R} \leq s < \nu/(1-\nu)^{1-1/\nu}$ and $\nu < 1$, the graph H is a regular $(d, 2^r, s2^r, \nu)$-disperser with probability at least $1 - 2^{-2^r}$.

Proof. Let S_1 be a subset of cardinality 2^r of the vertices in V_1 and S_2 be a subset of cardinality $s2^r$ of the vertices in V_2. Call a vertex $v \in V_1$ (S_2, ν)-*bad* if at least a fraction ν of the neighbors of v are in S_2. Let A_{S_1,S_2} be the event that all $v \in S_1$ are (S_2, ν)-bad. We now estimate $\Pr[A_{S_1,S_2}]$ for fixed S_1 and S_2.

The construction of H can be viewed as selecting, uniformly at random, a perfect matching on the vertices of a $d2^R \times d2^R$ bipartite graph G. Every vertex in V_1 corresponds to d vertices on the left side of G; every vertex in V_2 corresponds to $d2^{R-r}$ vertices on the right side. There are $(d2^R)!$ perfect matchings in G. For A_{S_1,S_2} to occur, the following must occur for every vertex in $v \in V_1$: At least νd of the vertices in G corresponding to v must be matched with vertices corresponding to accepting computations of the verifier. There are at most

$$\binom{d}{\nu d}^{2^r} \binom{sd2^R}{\nu d2^r} (\nu d2^r)!(d2^R - \nu d2^r)!$$

such matchings, thus

$$\Pr[A_{S_1,S_2}] \leq \binom{d}{\nu d}^{2^r} \frac{(sd2^R)!}{(sd2^R - \nu d2^r)!} \cdot \frac{(d2^R - \nu d2^r)!}{(d2^R)!} < \binom{d}{\nu d}^{2^r} s^{\nu d2^r},$$

where the last inequality follows since $s < 1$. By using the bound (6) from Lemma 6.7, we can rewrite the bound on $\Pr[A_{S_1,S_2}]$ as

$$\Pr[A_{S_1,S_2}] \leq 2^{(-(1-\nu)\log(1-\nu) - \nu\log(\nu/s))d2^r}.$$

This implies that

$$\Pr\left[\bigcup_{S_1}\bigcup_{S_2} A_{S_1,S_2}\right] \leq \sum_{S_1}\sum_{S_2} \Pr[A_{S_1,S_2}] \leq \binom{2^r}{s2^r}\binom{2^R}{2^r} \Pr[A_{S_1,S_2}].$$

The latter quantity above is at most 2^{-2^r} since

$$d = \frac{R+2}{\nu\log(\nu/s) + (1-\nu)\log(1-\nu)}. \qquad \square$$

The acceptance tester is modified in accordance with the new notion of a bad vertex from Lemma 6.8.

Definition 6.9. *Given a verifier V from some proof system and an input x, we construct a* threshold tester *$T(H, V, x, \nu)$ as follows: We pick a random bipartite graph H as in Lemma 6.8, with $V_1 = \{0,1\}^R$ and $V_2 = \{0,1\}^r$. Every vertex in V_1 has*

$$d = \frac{R+2}{\nu\log(\nu/s) + (1-\nu)\log(1-\nu)}$$

neighbors in V_2. The threshold tester $T(H, V, x, \nu)$ then behaves as follows on oracle π: It picks a random $v \in V_1$, and determines the d neighbors $u_1, \ldots, u_d$ of v in V_2. It simulates V on x and π with random strings

$u_1, \ldots, u_d$, and accepts if at least νd of the simulations accept. We write $T^\pi(H, V, x, \nu)$ for $T(H, V, x, \nu)$ with the oracle fixed to π. We say that a random string σ is an accepting string *for $T^\pi(H, V, x, \nu)$ if $T^\pi(H, V, x, \nu)$ accepts when the vertex $v \in V_1$ corresponding to σ is the random vertex picked.*

Lemma 6.10. *Suppose that $L \in \mathbf{FPCP}_{c,s}[r, f]$, and that V is the verifier in that proof system. If we construct a threshold tester $T(H, V, x, \nu)$ from V and some input x as described in Definition 6.9, it has the following properties:*

1. *If $x \in L$, then there is an oracle π such that at least a fraction $(c - \nu)/(1 - \nu)$ of all R-bit strings are accepting strings for $T^\pi(H, V, x, \nu)$.*
2. *If $x \notin L$, then with probability $1 - 2^{-2^r}$ over the choice of H, for all oracles π at most 2^r of all R-bit strings are accepting strings for $T^\pi(H, V, x, \nu)$.*

Proof. We first assume that $x \in L$. Then there is an oracle π such that at most $(1 - c)2^r$ of all possible r-bit strings make the verifier V reject. By Lemma 6.6 and Definition 6.9, this implies that at most a fraction $(1 - c)/(1 - \nu)$ of all R-bit strings are non-accepting strings for $T(H, V, x, \nu)$. Thus, at least a fraction $(c - \nu)/(1 - \nu)$ of all R-bit strings are accepting strings for $T(H, V, x, \nu)$.

Now we study the case $x \notin L$. By Lemma 6.8 with probability at least $1 - 2^{-2^r}$, the H used in the threshold tester is a regular $(d, 2^r, s2^r, \nu)$-disperser. Assume that H is such a disperser. Let π be any oracle. Let U be the set of all R-bit strings that are accepting strings for $T^\pi(H, V, x, \nu)$. Let S be the set of all r-bit strings which are accepting strings for $V^\pi(x)$. Since the proof system has soundness s, the set S can contain at most $s2^r$ strings. By Definition 6.9, a vertex is in U if at least fraction ν of its neighbors are connected to vertices in S. Since H is a regular $(d, 2^r, s2^r, \nu)$-disperser, U can contain at most 2^r elements. Thus with probability at least $1 - 2^{-2^r}$, for all oracles π, there are at most 2^r accepting R-bit string for $T^\pi(H, V, x, \nu)$. □

As usual we now want to construct a graph from the tester $T(H, V, x, \nu)$. To keep down the size of the graph, we let each vertex correspond to several accepting computations. Specifically, a vertex is defined by a random string and a list of νd accepting computations of the verifier V.

Definition 6.11. *Let $T(H, V, x, \nu)$ be a threshold tester from definition 6.9. A* ν-transcript *of an accepting computation of $T(H, V, x, \nu)$ is a list of νd lists of pairs of oracle queries and answers corresponding to the first νd accepting runs of V in this accepting computation.*

Definition 6.12. *Two ν-transcripts are* consistent *if whenever the same query occurs in the more than one transcript the answers are the same in both transcripts.*

Definition 6.13. *Given a threshold tester $T(H, V, x, \nu)$ we define the graph $G_{T(H,V,x,\nu)}$ as follows: Every vertex in $G_{T(H,V,x,\nu)}$ corresponds to a set of accepting computations of the threshold tester, namely those accepting computations of $T(H, V, x, \nu)$ that have the same random string and the same ν-transcript as per Definition 6.11. Two vertices in $G_{T(H,V,x,\nu)}$ are connected if they correspond to different random strings and the ν-transcripts are consistent.*

Lemma 6.14. *Suppose that we use a verifier V from a PCP with free bit complexity f to construct first $T(H, V, x, \nu)$ as described in Definition 6.9 and then $G_{T(H,V,x,\nu)}$ as described in Definition 6.13. Then $G_{T(H,V,x,\nu)}$ has at most $2^{R+(R+2)F_\nu}$ vertices, where F_ν is defined by equation (4).*

Proof. There are 2^R possible random strings. Since the free bit complexity of V is f, the number of accepting transcripts for each random string is at most $\binom{d}{\nu d}(2^f)^{\nu d} \leq 2^{(1+\nu f)d} \leq 2^{(R+2)F_\nu}$. □

Lemma 6.15. *Suppose there is a clique of size $p2^R$ in $G_{T(H,V,x,\nu)}$. Then there is a proof which the threshold tester $T(H, V, x, \nu)$ accepts with probability at least p. Conversely, if there is a proof which $T(H, V, x, \nu)$ accepts with probability p, then there is a clique in $G_{T(H,V,x,\nu)}$ of size at least $p2^R$.*

Proof. Let S be the vertices in a clique of size $p2^R$. Since there are no edges between vertices in $G_{T(H,V,x,\nu)}$ that correspond to the same random string, the vertices in S correspond to different random strings. Furthermore, each $v \in S$ corresponds to a list of νd lists of accepting computations of the verifier V. Construct a proof π as follows: If (q, a) occurs in some vertex, put a as the answer to q in the proof. Since the vertices in S form a clique, the ν-transcripts corresponding to vertices in S are consistent. Thus, the above process will not introduce any conflicts. For the queries which do not occur in any vertex, put an arbitrary answer in the proof. Now consider a random string σ for which there is a vertex in S. $T^\pi(H, V, x, \nu)$ will run the verifier V with d random strings. By the construction of π, at least νd of these runs accept, thus $T^\pi(H, V, x, \nu)$ will accept on this σ. To conclude, $T^\pi(H, V, x, \nu)$ accepts with probability at least p.

For the converse, consider a proof π making $T(H, V, x, \nu)$ accept with probability p. Then for each random string for which the verifier accepts there will be a vertex in the graph corresponding to the first νd accepting runs of V. Since the ν-transcripts corresponding to these vertices are consistent, all of these vertices will be connected. Thus, there is a clique of size $p2^R$ in $G_{T(H,V,x,\nu)}$. □

In the same way as in the proof of Lemmas 5.5 and 5.6 it follows that if a language $L \in \mathbf{FPCP}_{c,s}[r, f]$ and there is an algorithm B which can determine whether a graph with $2^{R-(R+2)F_\nu}$ vertices has a clique of size $(c-$

$\nu)2^R/(1-\nu)$ or a clique of at most size 2^r, then $L \in \mathbf{coRTIME}(T(N) + p(N))$, where $T(N)$ is the running time of the algorithm B on a graph with N vertices and p is some polynomial. Armed with this result, we can prove the following theorem:

Theorem 6.16. *Suppose that* $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r, f]$. *Let* ν *be any constant such that* $s < \nu < c$. *Let* $h = (c-\nu)/(1-\nu)$. *Then, for any* $R > r$, *it is impossible to approximate Max Clique in a graph with* $N = 2^{R+(R+2)F_\nu}$ *vertices within* $N^{1/(1+F_\nu)-r/R-(\log h^{-1})/R}$ *in polynomial time unless*

$$\mathbf{NP} \subseteq \mathbf{ZPTIME}\big(2^{\Theta(R+F_\nu+RF_\nu)}\big).$$

Proof. By Lemma 5.8, it suffices to show that it is hard to approximate Max Clique unless $\mathbf{NP} \subseteq \mathbf{coRTIME}(2^{\Theta(R+F_\nu+RF_\nu)})$.

Let the language L above be some **NP**-complete language, and the algorithm B some algorithm with polynomial running time. Then we obtain that unless $\mathbf{NP} \subseteq \mathbf{coRTIME}(2^{\Theta(R+F_\nu+RF_\nu)})$, it is impossible to approximate Max Clique in a graph with N vertices within

$$h2^{R-r} = N^{(R-r-\log h^{-1})/(R+(R+2)F_\nu)}$$

in polynomial time. The exponent in the above expression is at most

$$\frac{(R+2)-(r+2)-\log h^{-1}}{(R+2)(1+F_\nu)} < \frac{1}{1+F_\nu} - \frac{r}{R} - \frac{\log h^{-1}}{R}. \qquad \square$$

Also in this case it is possible to weaken the assumption on the approximation algorithm for Max Clique, which gives the following theorem (we omit the proof):

Theorem 6.17. *Suppose that* $\mathbf{NP} \subseteq \mathbf{FPCP}_{c,s}[r, f]$. *Let* ν *be any constant such that* $s < \nu < c$. *Let* $h = (c-\nu)/(1-\nu)$. *Then, for any* $R > r$, *it is impossible to approximate Max Clique in a graph with* $N = 2^{R+(R+2)F_\nu}$ *vertices within* $N^{1/(1+F_\nu)-r/R-(\log h^{-1})/R}$ *by algorithms with expected polynomial running time unless* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{\Theta(R+F_\nu+RF_\nu)})$.

If F_ν is a constant and $r(n) \in O(\log n)$, we get a theorem which says that Max Clique is hard to approximate within $N^{1/(1+F_\nu)-\epsilon-o(1)}$, for ν arbitrarily close to c, if we choose $R = r/\epsilon$ in the above theorem.

This might seem worse than in the case with two-sided error, where the interesting parameter was $\bar{f} = (f+\log c^{-1})/\log(c/s)$ instead of F_ν. However, when c is close to 1 and s is small, we expect $\bar{f}$ and F_ν to be close.

7 A PCP With Low Amortized Free Bit Complexity

In their recent paper, Samorodnitsky and Trevisan [22] give a new PCP for **NP** with optimal amortized query complexity, $1+\epsilon$ for any constant

$\epsilon > 0$. This result implies that the test has free bit complexity ϵ, for any constant $\epsilon > 0$. Since the analysis of the construction is much simpler than the analysis of the construction of Håstad [15], with reasonable effort it is possible to work through the construction with a non-constant ϵ; it turns out that the analysis is correct even in this case. In this section, we give a review of the construction of Samorodnitsky and Trevisan [22] rephrased in a way that is useful for the applications of this paper. The starting point is the same as in Håstad's recent construction [14] used to prove hardness of approximation for linear equations: The **NP**-hard problem G-gap E3-Sat-5 [2, 9].

Definition 7.1. *G-gap E3-Sat-5 is the following decision problem: We are given a Boolean formula ϕ in conjunctive normal form, where each clause contains exactly three literals and each literal occurs exactly five times. We know that either ϕ is satisfiable or at most a fraction G of the clauses in ϕ are satisfiable and are supposed to decide if the formula is satisfiable.*

7.1 An Interactive Proof System For G-gap E3-Sat-5

There is a well-known two-prover one-round interactive proof system that can be applied to G-gap E3-Sat-5. It consists of two provers, P_1 and P_2, and one verifier. Given an instance, i.e., an E3-Sat formula ϕ, the verifier behaves as follows:

1. Pick a clause C and variable x in C uniformly at random from the instance.
2. Send x to P_1 and C to P_2. P_1 returns an assignment to x and P_2 returns an assignment to the variables in C.
3. Accept if these assignments are consistent and satisfy C.

If the provers are honest, the verifier always accepts with probability 1 when ϕ is satisfiable. However, the provers can fool the verifier to accept an unsatisfiable instance of G-gap E3-Sat-5 with probability at most $(2 + G)/3$. To summarize this in the language of PCPs, the abovementioned proof system has completeness 1 and soundness $(2+G)/3$. The soundness can be lowered to $((2 + G)/3)^u$ by repeating the protocol u times independently, but it is also possible to construct a one-round proof system with lower soundness as follows: The verifier picks u clauses $\{C_1, \ldots, C_u\}$ uniformly at random from the instance. For each C_i, it also picks a variable x_i from C_i uniformly at random. The verifier then sends $\{x_1, \ldots, x_u\}$ to P_1 and the clauses $\{C_1, \ldots, C_u\}$ to P_2. It receives an assignment to $\{x_1, \ldots, x_u\}$ from P_1 and an assignment to the variables in $\{C_1, \ldots, C_u\}$ from P_2, and accepts if these assignments are consistent and satisfy $C_1 \wedge \cdots \wedge C_u$. As above, the completeness of this proof system is 1, and it can be shown [21] that the soundness is at most

c_G^u, where $c_G < 1$ is some constant depending on G but not on u or the size of the instance.

7.2 The PCP

The proof is what Håstad [14] calls a Standard Written Proof with parameter u. It is supposed to represent a string of length n. When ϕ is a satisfiable formula this string should be a satisfying assignment.

Definition 7.2. *If U is some set of variables taking values in $\{-1,1\}$, we denote by $\{-1,1\}^U$ the set of every possible assignment to those variables. Define $\mathcal{F}_U = \{f\colon \{-1,1\}^U \to \{-1,1\}\}$.*

Definition 7.3. *The* long code *of an assignment $x \in \{-1,1\}^U$ is a mapping $A_x\colon \mathcal{F}_U \to \{-1,1\}$ where $A_x(f) = f(x)$.*

Definition 7.4. *A* Standard Written Proof with parameter u *contains for each set $U \subseteq [n]$ of size at most u a string of length $2^{2^{|U|}}$, which we interpret as the table of a function $A_U\colon \mathcal{F}_U \to \{-1,1\}$. It also contains for each set W constructed as the set of variables in u clauses a function $A_W\colon \mathcal{F}_W \to \{-1,1\}$.*

Definition 7.5. *A Standard Written Proof with parameter u is a* correct proof *for a formula ϕ of n variables if there is an assignment x, satisfying ϕ, such that A_V is the long code of $x|_V$ for any V of size at most u or any V constructed as the set of variables of u clauses.*

The verifier is parameterized by the integer k and the positive real number $\epsilon > 0$, and it should accept with high probability if the proof is a correct Standard Written Proof for a given formula ϕ.

The verifier uses a convention called *folding* when it accesses the proof. This convention ensures that the constant term in the Fourier series—the term corresponding to the case when α is the function that always evaluates to 1 for all arguments—is zero, which turns out to be important below. Håstad [14] calls this *folding over true*. Another technical property that turns out to be important is that the only non-zero terms in the Fourier expansion of certain tables in the proof are terms corresponding to the case when α is a function evaluating to -1 only for arguments that form satisfying assignments to a Boolean formula Φ. This can be achieved by what Håstad [14] calls *conditioning upon Φ*. Folding over true and conditioning upon Φ can be done simultaneously, we refer the reader to Håstad's paper [14] for details. In this paper, all tables accessed by the verifier are assumed to be folded over true. The tables that are also conditioned upon Φ are denoted by $A_{V,\Phi}$ below.

We are now ready describe the procedure used by the verifier.

1. Select uniformly at random u variables $x_1, \ldots, x_u$. Let U be the set of those variables.
2. For $j = 1, \ldots, m$, select uniformly at random u clauses $C_{j,1}, \ldots, C_{j,u}$ such that clause $C_{j,i}$ contains variable x_i. Let Φ_j be the Boolean formula $C_{j,1} \wedge \cdots \wedge C_{j,u}$. Let W_j be the set of variables in the clauses $C_{j,1}, \ldots, C_{j,u}$.
3. For $i = 1, \ldots, k$, select uniformly at random $f_i \in \mathcal{F}_U$.
4. For $j = 1, \ldots, k$, select uniformly at random $g_j \in \mathcal{F}_{W_j}$.
5. For all $(i, j) \in [k] \times [k]$, choose $e_{ij} \in \mathcal{F}_{W_j}$ such that, independently for all $y \in W$,
   (a) With probability $1 - \epsilon$, $e_{ij}(y) = 1$.
   (b) With probability ϵ, $e_{ij}(y) = -1$.
6. Define h_{ij} such that $h_{ij}(y) = f_i(y|_U) g_j(y) e_{ij}(y)$.
7. If for all $(i, j) \in [k] \times [k]$, $A_U(f_i) A_{W_j, \Phi_j}(g_j) A_{W_j, \Phi_j}(h_{ij}) = 1$, then accept, else reject.

This verifier has query complexity $2k + k^2$ and since there are exactly $2k$ accepting configurations for every outcome of the verifier's random string, both the free bit and the average free bit complexity are $2k$.

Lemma 7.6. *The verifier needs*

$$r \leq u \log n + ku \log 5 + k2^u + k2^{3u} + k^2 2^{3u} \log \epsilon^{-1} \tag{7}$$

random bits.

Proof. To select the set U, at most $\log n^u$ random bits are needed. Once U has been selected, it is enough to use $k \log 5^u$ random bits to select the sets $W_1, \ldots, W_k$ since every variable occurs in five clauses. Since there are 2^{2^s} functions from a set of size s to $\{-1, 1\}$, it is enough to use $k2^u + k2^{3u}$ to select the functions $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$. To sample one of the error functions $e_{11}, \ldots, e_{kk}$, we need to use $\log \epsilon^{-1}$ random bits for every possible assignment to the variables it depends on. Thus, $k^2 2^{3u} \log \epsilon^{-1}$ random bits suffice to sample all the error functions. □

Lemma 7.7. *The completeness of the above PCP is at least* $(1 - \epsilon)^{k^2}$.

Proof. Given a correct proof, the verifier can only reject if one of the error functions e_{ij} are not 1 for the string encoded in the proof. Since the error functions are chosen pointwise uniformly at random, the probability that they all evaluate to 1 for the string encoded in the proof is $(1 - \epsilon)^{k^2}$. Thus, the verifier accepts a correct proof with at least this probability. □

To prove a bound on the soundness, Samorodnitsky and Trevisan [22] expand an expression for the acceptance condition in a Fourier series, and then manipulate this expression to obtain a strategy for the provers P_1 and P_2 in the two-prover one-round interactive proof system for G-gap E3-Sat-5 from Sec. 7.1. We outline the quite technical proof below. In the proof we need to use the Fourier expansion of the long code. More details on this topic can be found in the paper of Samorodnitsky and Trevisan [22] or in Håstad's paper [14], for the purposes in this paper it is sufficient to know that a function $A\colon \mathcal{F}_V \to \{-1,1\}$ can be written as

$$A(f) = \sum_{\alpha \in \mathcal{F}_V} \hat{A}_\alpha \chi_\alpha(f)$$

where

$$\chi_\alpha(f) = \prod_{\substack{x \in \{-1,1\}^V \\ \alpha(x) = -1}} f(x),$$
$$\hat{A}_\alpha = \langle A, \chi_\alpha \rangle = 2^{-2^{|V|}} \sum_{f \in \mathcal{F}_V} A(f)\chi_\alpha(f).$$

Lemma 7.8. *Suppose that the verifier in the above PCP accepts with probability $2^{-k^2} + \delta$. Then there exists a strategy for the provers P_1 and P_2 in the two-prover one-round interactive proof system for G-gap E3-Sat-5 from Sec. 7.1 such that the verifier accepts with probability at least $4\epsilon\delta^2$.*

Proof. To shorten the notation, we define the shorthands $A(f) = A_U(f)$ and $B_j(g) = A_{W_j,\Phi_j}(g)$. The test in the PCP accepts with probability $2^{-k^2} \sum_{S \subseteq E} \mathrm{E}[T_S]$ where

$$T_S = \prod_{(i,j) \in S} A(f_i) B_j(g_j) B_j(h_{ij}).$$

We use the convention that $T_\emptyset = 1$. Suppose that the acceptance probability is at least $2^{-k^2} + \delta$ for some $\delta > 0$. Then some term, corresponding to some $S \neq \emptyset$, in the above expression is at least δ. Number the vertices in this set S in such a way that there is at least one edge of the form $(1, j)$ and all edges of that form are $(1,1), \ldots, (1,d)$. Split the product in the definition of T_S into the two factors

$$T_S = \prod_{(i,j) \in S, i \neq 1} A(f_i) B_j(g_j) B_j(h_{ij}) \prod_{j=1}^{d} A(f_1) B_j(g_j) B_j(h_{1,j})$$

Since the first factor is independent of f_1 and $e_{1,1}, \ldots, e_{1,k}$, we use conditional expectation to rewrite $\mathrm{E}[T_S]$. If we let $\mathrm{E}_1[\cdot]$ denote the expected value taken over the random variables f_1 and $e_{1,1}, \ldots, e_{1,k}$, we obtain

$$\mathrm{E}[T_S] = \mathrm{E}\Bigg[\prod_{(i,j)\in S, i\neq 1} A(f_i)B_j(g_j)B_j(h_{ij})\, \mathrm{E}_1\Bigg[\prod_{j=1}^{d} A(f_1)B_j(g_j)B_j(h_{1,j})\Bigg]\Bigg],$$

which implies, since $|A(\cdot)| = |B_j(\cdot)| = 1$, that by the Cauchy-Schwartz inequality

$$\big|\mathrm{E}[T_S]\big|^2 \leq \mathrm{E}\Bigg[\Bigg|\mathrm{E}_1\Bigg[\big(A(f_1)\big)^d \prod_{j=1}^{d} B_j(h_{1,j})\Bigg]\Bigg|^2\Bigg]. \tag{8}$$

The remaining factors are expressed using the Fourier transform:

$$A(f_1) = \sum_{\alpha\in\mathcal{F}_U} \hat{A}_\alpha \chi_\alpha(f_1), \tag{9}$$

$$B_j(h_{1,j}) = \sum_{\beta_j\in\mathcal{F}_{W_j}} \hat{B}_{j,\beta_j} \chi_{\beta_j}(h_{1,j}), \tag{10}$$

where $\hat{A}_\alpha = \langle A, \chi_\alpha\rangle$ and $\hat{B}_{j,\beta_j} = \langle B_j, \chi_{\beta_j}\rangle$. When we insert the Fourier expansions (9) and (10) into the bound (8) and expand the products, we obtain one term for each possible combination of α and $\beta_1, \ldots, \beta_d$. It turns out that many of these terms vanish and that the remaining bound is as follows:

$$\big|\mathrm{E}[T_S]\big|^2 \leq \mathrm{E}\Bigg[\sum_{\substack{\alpha,\beta_1,\ldots,\beta_d \\ \alpha=\pi_U(\beta_1)\cdots\pi_U(\beta_d)}} \hat{A}_\alpha^2 \hat{B}_{1,\beta_1}^2 \cdots \hat{B}_{d,\beta_d}^2 (1-2\epsilon)^{2(|\beta_1|+\cdots+|\beta_d|)}\Bigg],$$

where the projections $\pi_U(\beta_j)$ are defined as the functions sending a y in W_j to the element $\prod_{y:y|_U=x} \beta(y)$.

Let us sum up what we have done so far: Given that the verifier in the PCP accepts with probability at least $2^{-k^2} + \delta$,

$$\mathrm{E}\Bigg[\sum_{\substack{\alpha,\beta_1,\ldots,\beta_d \\ \alpha=\pi_U(\beta_1)\cdots\pi_U(\beta_d)}} \hat{A}_\alpha^2 \hat{B}_{1,\beta_1}^2 \cdots \hat{B}_{d,\beta_d}^2 (1-2\epsilon)^{2(|\beta_1|+\cdots+|\beta_d|)}\Bigg] \geq \delta^2,$$

where the expectation is over the verifier's choice of U and $W_1, \ldots, W_k$.

We now construct a strategy for the provers P_1 and P_2. Prover P_1 receives a set U of u variables. For $j = 2, \ldots, d$, P_1 selects uniformly at random u clauses $C_{j,1}, \ldots, C_{j,u}$ such that clause $C_{j,i}$ contains variable x_i. Let Φ_j be the Boolean formula $C_{j,1} \wedge \cdots \wedge C_{j,u}$. Let W_j be the set of variables in the clauses $C_{j,1}, \ldots, C_{j,u}$. Then P_1 computes the Fourier coefficients

$\hat{A}_\alpha = \langle A, \chi_\alpha \rangle$ and $\hat{B}_{j,\beta_j} = \langle B_j, \chi_{\beta_j} \rangle$ for $j = 2, \ldots, d$, selects $(\alpha, \beta_2, \ldots, \beta_d)$ randomly such that $\Pr[(\alpha, \beta_2, \ldots, \beta_d)] = \hat{A}_\alpha^2 \hat{B}_{2,\beta_2}^2 \cdots \hat{B}_{d,\beta_d}^2$, forms the function $\alpha' = \alpha \pi_U(\beta_2) \cdots \pi_U(\beta_d)$ and returns an arbitrary x such that $\alpha'(x) \neq 1$. If no such x exists, P_1 returns an arbitrary $x \in \{-1, 1\}^U$.

Prover P_2 receives Φ_1 consisting of u clauses, computes $\hat{B}_{1,\beta_1} = \langle B_j, \chi_{\beta_1} \rangle$, selects a random β_1 with the distribution $\Pr[\beta_1] = \hat{B}_{1,\beta_1}^2$, and returns a random y such that $\beta_1(y) \neq 1$. Such a y always exists since the tables in the proof are folded, and since $\hat{B}_{1,\beta_1}$ are the Fourier coefficients for B_1, such assignments satisfy Φ_1.

The acceptance probability of this strategy can be written

$$\Pr[\text{accept}] = \mathrm{E}\big[\Pr[\text{accept} \mid U, \Phi_1, \ldots, \Phi_d]\big].$$

Thus, we assume from now on that U and $\Phi_1, \ldots, \Phi_d$ are fixed and try to estimate the acceptance probability under these assumptions.

The folding implies that there exists x such that $(\pi_U(\beta_1))(x) \neq 1$; by the conditioning the function α' sends every such x to the element -1. This implies that there exists a y such that $x = y|_U$ and $\beta_1(y) \neq 1$. Given the x chosen by P_1, the probability that P_2 chooses a y such that $y|_U = x$ and $\beta_1(y) \neq 1$ is at least $1/|\beta_1|$. All this put together implies that the acceptance probability can be bounded from below by

$$\Pr[\text{accept} \mid U, \Phi_1, \ldots, \Phi_k] \geq \sum_{\substack{\alpha, \beta_1, \ldots, \beta_d \\ \alpha = \pi_U(\beta_1) \cdots \pi_U(\beta_d)}} \frac{\hat{A}_\alpha^2 \hat{B}_{1,\beta_1}^2 \cdots \hat{B}_{d,\beta_d}^2}{|\beta_1|}.$$

We now use the bounds

$$\frac{1}{|\beta|} > 4\epsilon e^{-4\epsilon|\beta|} > 4\epsilon(1 - 2\epsilon)^{2|\beta|},$$

where the first inequality follows since $e^x > 1 + x > x$ for any real positive x and the second inequality follows from $e^{-x} > 1 - x$, which is true for any real positive x. Thus,

$$\Pr[\text{accept} \mid U, \Phi_1, \ldots, \Phi_m] \geq 4\epsilon\delta^2. \qquad \square$$

Corollary 7.9. *The PCP described above is a PCP for the* ***NP****-hard language G-gap E3-Sat-5 provided that* $c_G^u < 4\epsilon\delta^2$.

Proof. The provers P_1 and P_2 with the properties stated in Lemma 7.8 can be used to make the verifier in the two-prover one-round protocol for G-gap E3-Sat-5 from Sec. 7.1 accept an incorrect proof with probability at least $4\epsilon\delta^2$. On the other hand, we know [21] that the soundness of the two-prover one-round protocol for G-gap E3-Sat-5 from Sec. 7.1 is c_G^u. $\square$

Theorem 7.10. *G*-gap E3-Sat-5 $\in \mathbf{FPCP}_{c,s}[r, f]$, *where*

$$\begin{aligned}
&c \geq 1/e,\\
&s \leq 2^{1-k^2},\\
&r = C_0 k^2\big(\log n + k \log 5\big) + k 2^{3C_0 k^2} + k^2 \log k^2 2^{3C_0 k^2},\\
&f = 2k,
\end{aligned}$$

for any increasing function $k(n)$ *and some* $C_0 \in \Theta(1)$.

Proof. To get $c \geq 1/e$ we need to have $\epsilon = 1/k^2$ and to get $s \leq 2^{1-k^2}$ we need to have $\delta = 2^{-k^2}$. These choices imply that

$$u > \frac{\log \epsilon^{-1}\delta^{-2} - 2}{\log c_G^{-1}} \geq \frac{2k^2 + \log k^2 - 2}{\log c_G^{-1}} = \Theta(k^2),$$

thus we select $u = C_0 k^2$ for some $C_0 \in \Theta(1)$. When we insert these choices of parameters into the bound (7) on the number of random bits from Lemma 7.6, we obtain

$$r \leq C_0 k^2\big(\log n + k \log 5\big) + k 2^{3C_0 k^2} + k^2 \log k^2 2^{3C_0 k^2}. \qquad \square$$

Corollary 7.11. *G*-gap E3-Sat-5 $\in \mathbf{FPCP}_{c,s}[r, f]$, *where*

$$\begin{aligned}
&c \geq 1/e,\\
&s \leq 2^{-\Theta(\log\log n)},\\
&r \in \Theta(\log n \log\log n),\\
&f \in \Theta(\sqrt{\log\log n}).
\end{aligned}$$

Proof. Select $k^2 = C_1 \log\log n$, where $3C_0C_1 < 1$. Then

$$\begin{aligned}
r &= \Theta(\log n \log\log n) + \Theta(\log\log n \log\log\log n)(\log n)^{3C_0C_1}\\
&= \Theta(\log n \log\log n),
\end{aligned}$$

where the last equality follows since $3C_0C_1 < 1$. $\square$

8 Hardness Of Approximating Max Clique

When we combine Corollary 7.11 from Sec. 7.2 with our reductions from Sec. 6.2, we obtain the following result regarding the approximability of Max Clique:

Theorem 8.1. *The size of the largest clique in a graph with* N *vertices cannot be approximated within* $N^{1-O(1/\sqrt{\log\log N})}$ *in polynomial time unless* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{O(\log n(\log\log n)^{3/2})})$.

Proof. By Corollary 7.11, G-gap E3-Sat-5 $\in \mathbf{FPCP}_{c,s}[r, f]$, where

$$
\begin{aligned}
&c \geq 1/e, \\
&s \leq 2^{-\Theta(\log\log n)}, \\
&r \in \Theta(\log n \log\log n), \\
&f \in \Theta(\sqrt{\log\log n}).
\end{aligned}
$$

If we select $\nu = 1/3 < c$,

$$F_\nu = \frac{\Theta(\sqrt{\log\log n})}{\Theta(\log\log n)} = \Theta(1/\sqrt{\log\log n}).$$

Then we set $R = r/F_\nu = \Theta(\log n(\log\log n)^{3/2})$ in Theorem 6.16 and get that

$$h = \frac{c-\nu}{1-\nu} \geq \frac{3-e}{2e} \in \Theta(1) \tag{11}$$

and that it is impossible to approximate Max Clique in a graph with $N = 2^{R+(R+2)F_\nu}$ vertices within

$$N^{1/(1+F_\nu)-r/R-(\log h^{-1})/R} \geq N^{1-\Theta(1/\sqrt{\log\log n})} \tag{12}$$

in polynomial time, unless

$$\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{\Theta(r/F_\nu+F_\nu+r)}) = \mathbf{ZPTIME}(2^{\Theta(\log n(\log\log n)^{3/2})})$$

Since $N = 2^{R+(R+2)F_\nu} = 2^{\Theta(\log n(\log\log n)^{3/2})}$, $\log\log N = \Theta(\log\log n)$, which implies that the ratio (12) can be written as $N^{1-O(1/\sqrt{\log\log N})}$. □

Note that we do not gain anything if we use Theorem 6.4 instead of Theorem 6.16. In the former case we get

$$\bar{f} = \frac{2k + O(1)}{k^2 + O(1)} = \frac{2}{k} + o(1/k). \tag{13}$$

and to get a reasonable value for r, we need to set $k^2 = O(\log\log n)$. Thus we get the same hardness result (except for the constant), but with a stronger assumption, i.e., $\mathbf{NP} \not\subseteq \mathbf{BPTIME}(\cdot)$ instead of $\mathbf{NP} \not\subseteq \mathbf{ZPTIME}(\cdot)$, if we use Theorem 6.4.

We could try to improve the result results by strengthening the assumptions on **NP**. To do this, we probably need a new PCP which does not use the long code. Basically, the what happens when we try to improve the lower bound is the following: As we try to increase the gap by querying more bits in the proof, the number of random bits used increases, with the result that the size of the FGLSS graph grows to fast for us to get any improvement.

To see this, suppose we try to get an even larger gap in Corollary 7.11. To get this, we would need a parameter $k = \Omega(\sqrt{\log\log n})$. This would make

the last term in the bound (7) dominate, that is, $r = \Theta(k^2 \log k^2 2^{3C_0 k^2})$. If we choose the rest of the parameters as before (with respect to k), we get that the size of the FGLSS graph, $N = 2^{\Theta(r/F_\nu)} = 2^{\Theta(rk)}$. It follows that $\log\log N = \Theta(k^2)$, or equivalently, that $k = \Theta(\sqrt{\log\log N})$.

If we look at the important term in Theorem 6.16, we see that the best result we can get is that Max Clique is hard to approximate within $N^{1/(1+F_\nu)} = N^{1/(1+\Theta(1/k))}$. Since $k = \Theta(\sqrt{\log\log N})$, the best lower bound we can hope for is one of the form $N^{1-O(1/\sqrt{\log\log N})}$.

Since the last term in r comes from choosing bits to read in the proof, a more efficient encoding of the proof would decrease this term, and thus we would get a better expression of k as a function of N.

9 The Construction Of Feige and Kilian

In their reduction, Feige and Kilian [8, 11] do not compute a lower bound on the approximability of the chromatic number *ab initio*. Instead, they computed a lower bound on the approximability of a relaxation of the chromatic number, the *fractional chromatic number*. Their proof circles around the following four quantities of a graph G:

1. $V(G)$ = the set of vertices in G.
2. $\alpha(G)$ = the size of G's maximum independent set.
3. $\chi(G)$ = the minimum number of colors needed to vertex-color G.
4. $\chi_f(G)$ = the fractional chromatic number of G.

The above parameters are intuitively related, since a graph with a large maximum independent set should not need many colors to properly vertex color it.

Lemma 9.1. *For any graph G*

$$\alpha(G) \cdot \chi_f(G) \geq |V(G)|. \tag{14}$$

Proof. By the definition of the fractional chromatic number, there exists a distribution D on G's independent sets with the property that for any vertex $v \in V(G)$, $\Pr_D[v \text{ covered by } I] \geq 1/\chi_f(G)$ when I is selected according to D. Introduce an indicator random variable

$$X_v = \begin{cases} 1 & \text{if } v \text{ is covered by } I, \\ 0 & \text{otherwise.} \end{cases}$$

Then $\sum_{v \in V(G)} X_v$ is the size of the independent set selected according to D. Since all of G's independent sets have cardinality at most $\alpha(G)$,

$$\alpha(G) \geq \mathrm{E}_D\left[\sum_{v \in V(G)} X_v\right] = \sum_{v \in V(G)} \Pr_D[X_v = 1] \geq |V(G)|/\chi_f(G). \qquad \square$$

It turns out that the fractional chromatic number is sandwiched between two expressions involving the chromatic number. Since we can pick the independent sets corresponding to the color classes obtained from a proper vertex-coloring of G with $\chi(G)$ colors and use the uniform distribution on these independent sets as the distribution D above, the fractional chromatic number can be bounded from above by

$$\chi_f(G) \leq \chi(G). \tag{15}$$

As for a lower bound, Lovász [18] has shown that

$$\chi_f(G) \geq \frac{\chi(G)}{1 + \ln \alpha(G)}. \tag{16}$$

This implies that a hardness result of the form n^β for the fractional chromatic number translates into a similar hardness result for the chromatic number.

Lemma 9.2. *Suppose that the fractional chromatic number of a graph G with n vertices cannot be approximated within $n^{\beta(n)}$. Then the chromatic number of this graph cannot be approximated within $n^{\beta(n)-O(\log\log n/\log n)}$.*

Proof. Suppose that we have an algorithm A that outputs a value $A(G) \leq \chi(G) \cdot n^{\beta(n)}/(1 + \ln \alpha(G))$. By the bound (16), we can use

$$A(G) \leq \chi(G) \cdot n^{\beta(n)}/\big(1 + \ln \alpha(G)\big) \leq \chi_f(G) \cdot n^{\beta(n)}$$

to approximate $\chi_f(G)$ within $n^{\beta(n)}$, which contradicts the assumption of the lemma. Since $\alpha(G) \leq n$, we conclude that it is impossible to approximate $\chi(G)$ within

$$\frac{n^{\beta(n)}}{1 + \ln \alpha(G)} \geq \frac{n^{\beta(n)}}{1 + \ln n} = n^{\beta(n)-\log(1+\ln n)/\log n} = n^{\beta(n)-O(\log\log n/\log n)}. \quad \square$$

The above lemma implies that if we can prove a result similar to Theorem 3.11 for the fractional chromatic number, we more or less automatically obtain a result for Min Chromatic Number. As we saw in Theorem 3.18, Feige and Kilian [11] obtained such a result by introducing a new parameter, the covering radius, associated with the prover in a PCP. To amplify the bound, Feige and Kilian [8, 11] used graph products.

Definition 9.3. *For any two graphs G and H, we define the* graph product *$G \times H$ as the graph with vertex set $V(G) \times V(H)$ and edge set*

$$\big\{\big((v_G, v_H), (w_G, w_H)\big) : (v_G, w_G) \in E(G) \vee (v_H, w_H) \in E(H)\big\}$$

The k-wise graph product of G with itself is denoted by G^k.

It is easy to see that $\alpha(G^D) = (\alpha(G))^D$ and Lovász [18] has shown that $\chi_f(G^D) = (\chi_f(G))^D$. These observations immediately strengthens Corollary 3.19 as follows:

Corollary 9.4. *Suppose that* $\mathbf{NP} \subseteq \mathbf{RPCP}_{\rho,s}[r, f]$ *where* $r \in O(\log n)$ *and the other parameters are constants. Then it is impossible to approximate Min Chromatic Number within any constant in polynomial time unless* $\mathbf{P} = \mathbf{NP}$.

To get from a constant lower bound to a bound of the form n^β we must use $D = \Omega(\log n)$. However, this gives a graph with superpolynomial size. Feige and Kilian [8, 11] studied vertex induced subgraphs of graphs formed by a graph product and proved that for any graph G, vertex induced subgraphs of G^D can be used to obtained an amplified lower bound on the approximability of $\chi_f(G)$.

Lemma 9.5. *For any graph* G *and any integer* D*, a vertex induced subgraph* G' *obtained by selecting—independently and uniformly at random with probability* $2/C^D$*—vertices from* G^D*, satisfy the following relations with high probability:*

$$(|V(G)|/C)^D \le |V(G')| \le 4(|V(G)|/C)^D,$$
$$\alpha(G) \le C \implies \alpha(G') \le D|V(G)|.$$

Proof. The expected number of vertices in $V(G')$ is $2(|V(G)|/C)^D$. By standard Chernoff bounds [20, Chapter 4],

$$\Pr\bigl[\mathrm{E}[|V(G')|]/2 \le |V(G')| \le 2\,\mathrm{E}[|V(G')|]\bigr] \ge 1 - 2e^{-\mathrm{E}[|V(G')|]/8}.$$

To prove that the second property holds with high probability, first let S be any independent set in G^D. For every $s \in S$, let X_s be an indicator random variable for the event $\{s \in G'\}$. Since $\Pr[X_s = 1] = 2/C^D$ and we assume that $|S| \le C^D$,

$$\sum_{s \in S} \mathrm{E}[X_s] = 2|S|/C^D \le 2.$$

From [8, Lemma 2.12], it follows that

$$\Pr\left[\sum_{s \in S} X_s > D|V(G)|\right] \le e^{-D|V(G)|(\ln(D|V(G)|)-2)/2}.$$

The probability that the second property does not hold is at most the sum over all independent sets S of the above probability. Since there are at most $3^{D|V(G)|/3}$ independent sets in G^D [8, Lemma 2.13], this sum is at most

$$3^{D|V(G)|/3} e^{-D|V(G)|(\ln(D|V(G)|)-2)/2}.$$

Since $3^{1/3}/e < 1$, the sum tends to 0 as n grows. □

Corollary 9.6. *Suppose that* $L \in \mathbf{RPCP}_{\rho,s}[r, f]$*. Then it is possible to randomly construct a graph* G' *which has the following properties:*

$$|V(G')| \ge s^{-D} 2^{Df} \text{ with high probability,}$$
$$x \in L \implies \chi_f(G') \le 1/\rho^D,$$
$$x \notin L \implies \chi_f(G') \ge |V(G')|/D2^{r+f} \text{ with high probability.}$$

Proof. By Theorem 3.18, the graph $\overline{G_{V,x}}$ has the following properties:

$$|V(\overline{G_{V,x}})| = 2^{r+f},$$
$$x \in L \implies \chi_f(\overline{G_{V,x}}) \leq 1/\rho,$$
$$x \notin L \implies \alpha(\overline{G_{V,x}}) \leq s2^r.$$

By the choices $G = \overline{G_{V,x}}$ and $C = s2^r$ in Lemma 9.5, the following holds with high probability:

$$V(G') \geq (|V(\overline{G_{V,x}})|/s2^r)^D = 2^{Df}/s^D,$$
$$\alpha(\overline{G_{V,x}}) \leq s2^r \implies \alpha(G') \leq D|V(\overline{G_{V,x}})|.$$

When these properties are combined with the bound (14) from Lemma 9.1, the proof follows. □

Theorem 9.7. *Suppose that* $L \in \mathbf{RPCP}_{\rho,s}[r, f]$. *Let*

$$\beta = 1 - \frac{\log \rho^{-1}}{f + \log s^{-1}} - \frac{\log D + r + f}{D(f + \log s^{-1})}.$$

Then it is impossible to approximate χ_f *in a graph with* $N \leq 2^{D(f+\log s^{-1})}$ *vertices within* N^β *in polynomial time unless*

$$L \in \mathbf{ZPTIME}(2^{\Theta(r+D(f+\log s^{-1}))}).$$

Proof. Suppose there is an algorithm A approximating χ_f within N^β in polynomial time. Consider the following probabilistic algorithm:

1. On input x, construct the graph $G_{V,x}$ from the RPCP.
2. Sample a random subgraph $G' \subset \overline{G_{V,x}}^D$ with $N = |V(\overline{G_{V,x}}^D)|/(s2^r) = 2^{Df}/s^D$ vertices.
3. Run A on G'. Accept x if $A(G') \leq N/D2^{r+f}$, reject otherwise.

By Corollary 9.6, the above algorithm probabilistically decides L with one-sided error if A approximates χ_f within

$$\frac{N/D2^{r+f}}{1/\rho^D} = \frac{N\rho^D}{D2^{r+f}}.$$

This error can be removed with self-reduction. To relate the above ratio to N^β, we try to express it as $N^{\beta'}$. Solving for β' gives:

$$\beta' = \frac{\log N\rho^D - \log D2^{r+f}}{\log N} \tag{17}$$
$$= 1 + \frac{D\log\rho}{\log N} - \frac{\log D + r + f}{\log N} \tag{18}$$
$$\geq 1 - \frac{\log \rho^{-1}}{f + \log s^{-1}} - \frac{\log D + r + f}{D(f + \log s^{-1})} \tag{19}$$
$$= \beta. \tag{20}$$

As for the underlying assumption, step 1 takes time $\Theta(2^{r+f})$, step 2 can be done in time $\Theta(N)$, and step 3 takes time $\Theta(N^k)$ for some constant k. Since $N \leq (1/s)^D 2^{Df}$, we have that $L \in \mathbf{ZPTIME}(2^{\Theta(r+D(f+\log s^{-1}))})$ □

10 Randomizing the Protocol From Sec. 7

We now need to construct a protocol with good covering radius. We will do this by randomizing the protocol of Samorodnitsky and Trevisan [22], described in Section 7.

The intuition behind the randomized version of the protocol is that prover adds some dummy clauses—which are not ordinary 3-Sat clauses, but clauses which contain three literals and are satisfied for any assignment—to the instance of G-gap E3-Sat-5. The new variables added to the instance in this way are given a random assignment. This introduces a distribution on the proofs, which enables us to prove something about the covering radius. Since the verifier has free bit complexity $2k$, there are 2^{2k} different accepting views. We want to prove that each of those accepting views occur with probability 2^{-2k} according to the above distribution. Actually, we cannot achieve exactly this, but we can get within a constant factor, which is close enough.

In the case of perfect completeness, it is enough to prove that every possible outcome of the free bits occur with high enough probability when the prover selects a proof for the case $x \in L$. This was done by Feige and Kilian [8, 11] by requiring the selected functions to be *well balanced*. The verifier was then modified to abort its execution in a suitable way—it accepted in the original construction [8, 11], but we want it to reject in this paper—whenever it happened to select an f_i or a g_j without this property.

In our case, however, it may occur that a proof selected by the prover is rejected because of the error functions e_{ij}, which means that a naive computation of the probability of a certain outcome of the free bits overestimates the probability that this outcome occurs as an accepting computation. Indeed, for some random strings, there are no correct proofs which the verifier accepts. (Let us remark at this point that this does not contradict the fact that the free bit complexity is $2k$. Once the verifier has fixed its random string, there are 2^{2k} accepting views, it is just that some of those accepting views may correspond to an incorrect proof.) We resolve this issue by ensuring that there are no views for which very few proofs are accepted. This can be accomplished by requiring that the functions e_{ij} must be *sparse* and modify the verifier to reject if it selects an e_{ij} lacking this property.

Since we make the verifier reject if it selects any bad function, the size of the FGLSS graph decreases. We must prove that this decrease is $1 - o(1)$; Lemma 9.5 and Corollary 9.6 are still valid if that is the case.

10.1 The Randomized PCP

The proof in the randomized PCP is an extension of the Standard Written Proof with parameter u defined in Sec. 7.2.

Definition 10.1. *Let $R = \{r_{ij} : i \in [n] \wedge j \in [M]\}$. For any set $V \subseteq [n]$, let $R_V = \{r_{ij} : i \in V \wedge j \in [M]\}$.*

Definition 10.2. *A* Randomized Standard Written Proof with parameters u and M *contains for each set $U \subseteq [n]$ of size at most u a string of length $2^{2^{|U|(1+M)}}$, which we interpret as the table of a function $A_{U \cup R_U} : \mathcal{F}_{U \cup R_U} \to \{-1, 1\}$. It also contains for each set W constructed as the set of variables in u clauses a function $A_{W \cup R_W} : \mathcal{F}_{W \cup R_W} \to \{-1, 1\}$.*

Definition 10.3. *A Randomized Standard Written Proof with parameters u and M is a* correct proof *for a formula ϕ of n variables if there is an assignment x, to the $(M+1)n$ variables in ϕ and R, satisfying ϕ, such that $A_{V \cup R_V}$ is the long code of $x|_V$ concatenated with the variables in R_V for any V constructed as a subset of size at most u of the variables and any V constructed as the set of variables of u clauses.*

The intuition behind the above definitions is that we add, for every original variable, M new variables that are given a random assignment. This random assignment induces a distribution on the correct proofs. Note that the enumeration of the r_{ij} variables in the construction implies that $U \cup R_U \subseteq W \cup R_W$ whenever $U \subseteq W$.

Definition 10.4. *Given U and $W_1, \ldots, W_k$, the functions $f_i \in \mathcal{F}_U$ and $g_j \in \mathcal{F}_{W_j}$ are* well balanced *if, for an arbitrary fixed x,*

$$\Pr_{r_i \in_U \{-1,1\}} \left[\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v} \right] \geq 2^{-2k-1}. \tag{21}$$

for all settings of the non-random variables, all $\vec{u}$, and all $\vec{v}$.

Definition 10.5. *We call e_{ij} ϵ-*sparse *if, for an arbitrary fixed x,*

$$\Pr_{r_i \in_U \{-1,1\}} \left[e_{ij}(x) = -1 \;\middle|\; \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v} \right] < \epsilon \tag{22}$$

for all settings of the non-random variables in W_j, any well-balanced $\vec{f}$ and $\vec{g}$, all $\vec{u}$, and all $\vec{v}$.

The verifier is extended correspondingly. It is still parameterized by the integer k and the positive real number $\epsilon > 0$, but it should accept with high probability if the proof is a correct Randomized Standard Written Proof for a given formula ϕ. As in Sec. 7.2, all tables accessed by the verifier are assumed to be folded over true. The tables that are also conditioned upon Φ are denoted by $A_{V,\Phi}$ below.

1. Select uniformly at random u variables $U' = \{x_1, \ldots, x_u\}$. Let $U = U' \cup R_{U'}$.
2. For $j \in \{1, \ldots, m\}$, select uniformly at random u clauses $C_{j,1}, \ldots, C_{j,u}$ such that clause $C_{j,i}$ contains variable x_i. Let Φ_j be the Boolean formula $C_{j,1} \wedge \cdots \wedge C_{j,u}$, W'_j be the set of variables in Φ_j, and $W_j = W'_j \cup R_{W'_j}$.
3. For $i \in \{1, \ldots, k\}$, select uniformly at random $f_i \in \mathcal{F}_U$.
4. For $j \in \{1, \ldots, k\}$, select uniformly at random $g_j \in \mathcal{F}_{W_j}$.
5. Reject unless the functions $f_1, \ldots, f_k, g_1, \ldots, g_k$ are well-balanced.
6. For all $(i, j) \in [k] \times [k]$, choose $e_{ij} \in \mathcal{F}_{W_j}$ such that, independently for all $y \in W_j$,
   (a) With probability $1 - \epsilon$, $e_{ij}(y) = 1$.
   (b) With probability ϵ, $e_{ij}(y) = -1$.

   Reject unless the functions $e_{11}, \ldots, e_{kk}$ are 2ϵ-sparse.
7. Define h_{ij} such that $h_{ij}(y) = f_i(y|_U) g_j(y) e_{ij}(y)$.
8. If for all $(i, j) \in [k] \times [k]$, $A_U(f_i) A_{W_j, \Phi_j}(g_j) A_{W_j, \Phi_j}(h_{ij}) = 1$, then accept, else reject.

The prover, given a satisfying assignment to the formula ϕ, chooses the proof according to the distribution induced by choosing each r_{ij} uniformly and independently at random in $\{-1, 1\}$.

Lemma 10.6. *The verifier needs*

$$u \log n + ku \log 5 + k2^{u(1+M)} + k2^{3u(1+M)} + k^2 2^{3u(1+M)} \log \epsilon^{-1} \qquad (23)$$

random bits.

Proof. To select the set U, at most $\log n^u$ random bits are needed. Once U has been selected, it is enough to use $k \log 5^u$ random bits to select the sets $W_1, \ldots, W_k$ since every variable occurs in five clauses. Since there are 2^{2^s} functions from a set of size s to $\{-1, 1\}$, it is enough to use $k2^{u(1+M)} + k2^{3u(1+M)}$ to select the functions $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$. To sample one of the error functions $e_{11}, \ldots, e_{kk}$, we need to use $\log \epsilon^{-1}$ random bits for every possible assignment to the variables it depends on. Thus, $k^2 2^{3u(1+M)} \log \epsilon^{-1}$ random bits suffice to sample all the error functions. □

We note that the verifier can check whether its choice of $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$ resulted only in well balanced functions in time $O(2k2^{3u+2k})$, and whether its choice of $e_{11}, \ldots, e_{kk}$ resulted only in 2ϵ-sparse functions in time $O(k^2 2^{3u+2k})$. This time is polynomial in the size of the proof given that k and u are both $O(\log n)$.

10.2 The Prover's Perspective

The prover gives a random assignment the variables in the dummy clauses, i.e., to the r_i variables. This introduces a distribution on the correct proofs and enables us to prove something about the covering radius.

Lemma 10.7. *The RPCP in Sec. 10.1 has covering radius at least*

$$\min_{\vec{u},\vec{v}} \Pr_{\Pi}\big[\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v} \cap \vec{e}(x) = \vec{1}\big].$$

where Π is the distribution on the proofs induced by choosing each r_{ij} uniformly and independently at random in $\{-1, 1\}$.

Proof. By definition, ρ is a lower bound on $\Pr_{\Pi}[\text{view}]$ for any view making the verifier accept when $x \in L$. When we have a correctly encoded proof the verifier accepts if the error functions all evaluate to 1. Thus, the probability of the accepting view $(\vec{u}, \vec{v})$ is $\Pr_{\Pi}\big[\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v} \cap \vec{e}(x) = \vec{1}\big]$. □

Corollary 10.8. *The RPCP in Sec. 10.1 has covering radius at least*

$$\min_{\vec{u},\vec{v}} \Pr_{\Pi}\big[\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big] \min_{\vec{u},\vec{v}} \Pr_{\Pi}\big[\vec{e}(x) = \vec{1} \;\big|\; \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big].$$

where Π is the distribution on the proofs induced by choosing each r_{ij} uniformly and independently at random in $\{-1, 1\}$.

To bound the above probabilities we require that the functions $f_1, \ldots, f_k$, $g_1, \ldots, g_k$, and $e_{11}, \ldots, e_{kk}$ must have certain properties and claim that the covering radius is large whenever the functions have these properties.

Theorem 10.9. *Suppose that $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$ are well balanced and that $e_{11}, \ldots, e_{kk}$ are 2ϵ-sparse. Then $\rho \geq (1 - 2k^2\epsilon)2^{-2k-1}$.*

Proof. Since $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$ are well balanced,

$$\min_{\vec{u},\vec{v}} \Pr_{\Pi}\big[\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big] \geq 2^{-2k-1};$$

since $e_{11}, \ldots, e_{kk}$ are 2ϵ-sparse,

$$\min_{\vec{u},\vec{v}} \Pr_{\Pi}\big[\vec{e}(x) = \vec{1} \;\big|\; \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big] \geq 1 - k^2 \cdot 2\epsilon.$$

By Corollary 10.8, this implies that $\rho \geq (1 - 2k^2\epsilon)2^{-2k-1}$. □

Let us sum up what we have done so far: If the verifier selects only functions with certain nice properties, we obtain an RPCP with a covering radius that is only a constant factor from the optimal one—provided that we select $\epsilon \in O(1/k^2)$.

10.3 The Verifier's Perspective

Let us—for a moment—ignore the fact that the verifier may reject prematurely should it select a "bad" function. Then, with an argument identical to that in Sec. 7.2, it follows that the above PCP has completeness at least $(1-\epsilon)^{k^2}$ and soundness at most $2^{-k^2}+\delta$ provided that $c_G^u \leq 4\epsilon\delta^2$. Regarding the soundness, the calculations in Lemma 7.8 work also in this extended case by letting the set U in Lemma 7.8 correspond to $U' \cup R_{U'}$, and the sets W_j correspond to $W_j' \cup R_{W_j'}$. This gives a strategy for the provers P_1 and P_2 which gives an assignment to both the non-random and the random variables, and the success rate for this strategy is at least $4\epsilon\delta^2$.

That the verifier rejects if it selects a bad function can never increase the soundness. Thus, as far as the soundness is concerned we may ignore the fact that the verifier may reject prematurely. However, the size of the FGLSS graph corresponding to the verifier in question decreases slightly since some vertices, the vertices corresponding to some random strings, are removed entirely. We now prove that this decrease is miniscule. In fact, the only critical bound is the bound on χ_f when $x \notin L$. For this bound, i.e., the bound from Corollary 9.6, to remain valid, it is enough to prove that the decrease is less than a $1-o(1)$ factor. We start by proving a bound on the probability that $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$ are well balanced.

Lemma 10.10 [11, Lemma 5]. *Let r_0 be the number of random variables in U and let s_0 be the number of non-random variables in U. Let r_j and s_j be defined accordingly for the W_j. Let $n_j = r_j + s_j$. Suppose that*

$$r_j - 2k - 1 \geq 5n_j/6,$$
$$n_j \geq 6\log 2k + 12,$$
$$2^{(1/3)n_j - 1} - 2^{(1/6)n_j} \geq 2k + s_1 + \ldots + s_k.$$

Then all but a $2k\sum_{i=0}^{k} e^{-2^{n_j}}$ fraction of the choices of $f_1, \ldots, f_k, g_1, \ldots, g_k$ are well balanced.

Corollary 10.11. *Suppose that $k(n)$ is an increasing function, $M \geq 12$, and $u \geq 26k + 13$. Then all but a $2k(k+1)e^{-2^{u/2}}$ fraction of the choices of functions $f_1, \ldots, f_k, g_1, \ldots, g_k$ are well balanced.*

Proof. It follows from steps 1 and 2 in the description of the verifier in Sec. 10.1 that $r_j \geq n_j/(1+1/M)$. We want to select our parameters in such a way that $r_j - 2k - 1 \geq 5n_j/6$, which is, by the above bound on r_j, equivalent to

$$n_j \frac{1-5/M}{6+6/M} \geq 2k+1.$$

Since $M \geq 12$, the left hand side is at least

$$n_j \frac{1-5/12}{6+1/2} \geq \frac{n_j}{13}.$$

Thus, we need to select our parameters in such a way that $n_j \geq 26k + 13$. This bound will hold if $u \geq 26k+13$, which also implies that $n_j \geq 6 \log 2k+12$ for all $k > 0$. Since

$$2^{(1/3)n_j - 1} - 2^{(1/6)n_j} = \Omega(2^{n_j/3}),$$
$$2k + s_1 + \ldots + s_k = O(n_j^2),$$

clearly $2^{(1/3)n_j-1} - 2^{(1/6)n_j} \geq 2k + s_1 + \ldots + s_k$ for large enough n_j. Thus, the conditions in Lemma 10.10 are met and the result follows. □

Now we turn to proving that the error functions $e_{11}, \ldots, e_{kk}$ are 2ϵ-sparse with high probability given that the verifier selected well balanced functions $f_1, \ldots, f_k, g_1, \ldots, g_k$. Let Π be the distribution induced by choosing all the random variables uniformly and independently at random.

Lemma 10.12. *Assume that $f_1, \ldots, f_k, g_1, \ldots, g_k$ are well balanced and that $Mu \geq 4k + 2$. Then $\Pr_\Pi[\vec{a}_j \mid \vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}] \leq 2^{-5Mu/2}$ for any fixed assignment $\vec{a}_j$ to the variables in W_j and any fixed $\vec{u}, \vec{v}$*

Proof. By the definition of conditional probability,

$$\Pr_\Pi\left[\vec{a}_j \mid \vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}\right] = \frac{\Pr_\Pi\left[\vec{a}_j \cap \vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}\right]}{\Pr_\Pi\left[\vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}\right]} \leq \frac{\Pr_\Pi[\vec{a}_j]}{\Pr_\Pi\left[\vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}\right]}.$$

Since the $3Mu$ random variables in W_j are set independently at random, $\Pr_\Pi[\vec{a}_j] \leq 2^{-3Mu}$ and since the $\vec{f}$ and $\vec{g}$ are well balanced, $\Pr_\Pi[\vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}] \geq 2^{-2k-1}$. Thus,

$$\Pr_\Pi\left[\vec{a}_j \mid \vec{f}(\vec{x}) = \vec{u} \cap \vec{g}(\vec{x}) = \vec{v}\right] \leq 2^{-3Mu+2k+1} \leq 2^{-5Mu/2},$$

where the last inequality follows from our assumption on k. □

Corollary 10.13. *Suppose that $f_1, \ldots, f_k, g_1, \ldots, g_k$ are well balanced. Let r_j be the number of random variables in W_j. Assume that $Mu \geq 4k + 2$. Then the probability, over the choice of e_{ij} for fixed i and j, that e_{ij} is 2ϵ-sparse is at least $1 - \exp(3u + 2k - \epsilon^2 2^{Mu/3-1})$.*

Proof. By Definition 10.5, we must bound the probability that

$$\Pr_{\Pi}\big[e_{ij}(x) = -1 \;\big|\; \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big] \geq 2\epsilon \tag{24}$$

for some setting of the non-random variables in W_j and for some $\vec{u}$ and $\vec{v}$. By Lemma 10.12, we can assume that for a fixed j

$$\Pr_{\Pi}\big[\vec{a}_j \;\big|\; \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big] \leq 2^{-5Mu/2}$$

for any assignment $\vec{a}_j$ to the variables in W_j. Now fix $\vec{u}$, $\vec{v}$, and an assignment to the $3u$ non-random variables in W_j. Let

$$X_{\vec{a}_j} = -\epsilon + I_{\{e_{ij}(\vec{a}_j)=-1 | \vec{f}(x)=\vec{u}\cap\vec{g}(x)=\vec{v}\}} \tag{25}$$

$$X = \sum_{\vec{a}_j} \Pr_{\Pi}[\vec{a_j}|\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}]X_{\vec{a}_j}. \tag{26}$$

Then X is a random variable depending on the choice of e_{ij}. Furthermore,

$$\begin{aligned} X + \epsilon &= \sum_{\vec{a}_j} \Pr_{\Pi}[\vec{a_j}|\vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}]I_{\{e_{ij}(\vec{a}_j)=-1|\vec{f}(x)=\vec{u}\cap\vec{g}(x)=\vec{v}\}} \\ &= \Pr_{\Pi}\big[e_{ij}(x) = -1 \;\big|\; \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}\big]. \end{aligned}$$

We bound the probability that $X \geq \epsilon$ by noting that

$$\begin{aligned} &\mathrm{E}_{e_{ij}}\big[\Pr_{\Pi}[\vec{a_j} \mid \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}]X_{\vec{a}_j}\big] = 0, \\ &\big|\Pr_{\Pi}[\vec{a_j} \mid \vec{f}(x) = \vec{u} \cap \vec{g}(x) = \vec{v}]X_{\vec{a}_j}\big| \leq 2^{-5Mu/2}. \end{aligned}$$

The first identity follows by the construction of the functions e_{ij}, the second bound follows by the assumption in the formulation of the lemma. Since there are 2^{3Mu} random variables in W_j, there are 2^{3Mu} terms in the sum (26). From [1, Theorem A.16], it thus follows that

$$\Pr[X > \epsilon] \leq \exp\left(-\left(\frac{\epsilon}{2^{-5Mu/3}}\right)^2 \Big/ 2^{3Mu+1}\right) = \exp\big(-\epsilon^2 2^{Mu/3-1}\big).$$

This bound holds for fixed $\vec{u}$, $\vec{v}$, and a fixed assignment to the non-random variables in W_j.

Since there are 2^{2k} possible $\vec{u}, \vec{v}$ and 2^{3u} possible assignments to the non-random variables in W_j, we obtain that the probability, over the choice of e_{ij}, that e_{ij} is 2ϵ-sparse is at least $1 - 2^{3u+2k}\exp(-\epsilon^2 2^{Mu/3-1})$. □

By combining the above, we obtain the following bound on the probability that the verifier rejects prematurely.

Theorem 10.14. *Suppose that $k(n)$ is an increasing function, $M \geq 12$, and $u \geq 26k + 13$. Then, the verifier selects well balanced functions $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$ and 2ϵ-sparse error functions $e_{11}, \ldots, e_{kk}$ with probability at least $1 - 2k(k+1)e^{-2^{u/2}} - k^2\exp(3u + 2k - \epsilon^2 2^{Mu/3-1})$.*

Proof. By Corollary 10.11 all but a $2k(k+1)e^{-2^{u/2}}$ fraction of the choices of functions $f_1, \ldots, f_k, g_1, \ldots, g_k$ are well balanced. Since the verifier chooses uniformly at random among these functions, the probability that it selects a function that is not well-balanced is less than $2k(k+1)e^{-2^{u/2}}$.

Corollary 10.13 implies that, given that the verifier selected well balanced functions $f_1, \ldots, f_k, g_1, \ldots, g_k$, it selects an error function which is not 2ϵ-sparse with probability at most $k^2 \exp(3u + 2k - \epsilon^2 2^{Mu/3-1})$. □

10.4 Wrapping It Together

Theorem 10.15. G-gap E3-Sat-5 $\in \mathbf{RPCP}_{\rho,s}[r, f]$, *where*

$$\begin{aligned}
&\rho \geq 2^{-2k-2},\\
&s \leq 2^{1-k^2},\\
&r = C_0 k^2\big(\log n + 3k + \log(3M+1)\big) + k2^{C_0 k^2} + k^2(\log k^2 + 2)2^{3C_0 k^2},\\
&f = 2k,
\end{aligned}$$

for any increasing function $k(n)$ and some $C_0 \in \Theta(1)$.

Proof. By Theorem 10.9, the RPCP described in Section 10.1 has covering radius $\rho \geq (1 - 2k^2\epsilon)2^{-2k-1}$. We need to select ϵ in such a way that ρ is large. We choose $\epsilon = 1/4k^2$, which implies that

$$\rho \geq (1 - 2k^2\epsilon)2^{-2k-1} = 2^{-2k-2}.$$

To get soundness $s \leq 2^{1-k^2}$ we choose $\delta = 2^{-k^2}$. This gives the following bound on u:

$$u > \frac{\log \epsilon^{-1}\delta^{-2} - 2}{\log c_G^{-1}} \geq \frac{2k^2 + \log k^2}{\log c_G^{-1}} = \Theta(k^2),$$

thus we select $u = C_0 k^2$ for some $C_0 \in \Theta(1)$. When we insert these choices of parameters into the bound (23) on the number of random bits from Lemma 10.6, we obtain

$$r \leq C_0 k^2(\log n + 3k) + k2^{3C_0(1+M)k^2} + k^2(\log k^2 + 2)2^{3C_0(1+M)k^2}. \quad \square$$

Corollary 10.16. G-gap E3-Sat-5 $\in \mathbf{RPCP}_{\rho,s}[r, f]$, *where*

$$\begin{aligned}
&\rho \geq 2^{-\Theta(\sqrt{\log\log n})},\\
&s \leq 2^{-\Theta(\log\log n)},\\
&r \in \Theta(\log n \log\log n),\\
&f \in \Theta(\sqrt{\log\log n}).
\end{aligned}$$

Proof. Select $k^2 = C_1 \log\log n$, where $3C_0C_1(1+M) < 1$. Then the expression for the number of random bits becomes

$$\begin{aligned} r &= \Theta(\log n \log\log n) + \Theta(\log\log n \log\log\log n)(\log n)^{3C_0C_1(1+M)} \\ &= \Theta(\log n \log\log n), \end{aligned}$$

where the last equality follows since $3C_0C_1(1+M) < 1$. □

Let us remark at this point that the above choices of parameters satisfy the requirement that the decrease in the FGLSS graph due to the possibility of rejecting prematurely is bounded by a factor $1 - o(1)$.

11 Hardness Of Approximating Min Chromatic Number

When we combine Corollary 10.16 with the reductions from Sec. 9 we obtain our hardness result regarding the approximability of Min Chromatic Number:

Theorem 11.1. *The fractional chromatic number in a graph with N vertices cannot be approximated within $N^{1-O(1/\sqrt{\log\log N})}$ in polynomial time unless* $\mathbf{NP} \subseteq \mathbf{ZPTIME}(2^{O(\log n(\log\log n)^{3/2})})$.

Proof. By Corollary 10.16, G-gap E3-Sat-5 has an RPCP with the following parameters:

$$\begin{aligned} &\rho \geq 2^{-\Theta(\sqrt{\log\log n})}, \\ &s \leq 2^{-\Theta(\log\log n)}, \\ &r \in \Theta(\log n \log\log n), \\ &f \in \Theta(\sqrt{\log\log n}). \end{aligned}$$

If we set $D = r/k = \Theta(\log n\sqrt{\log\log n})$ and use Theorem 9.7, it follows that χ_f is impossible to approximate within N^β, where

$$\begin{aligned} \beta &= 1 - \frac{\log \rho^{-1}}{f + \log s^{-1}} - \frac{\log D + r + f}{D(f + \log s^{-1})} \\ &= 1 - \frac{O(\sqrt{\log\log n})}{\Omega(\log\log n)} - \frac{O(\log n \log\log n)}{\Omega(\log n(\log\log n)^{3/2})} \\ &= 1 - O(1/\sqrt{\log\log n}), \end{aligned}$$

unless

$$\mathbf{NP} \subseteq \mathbf{ZPTIME}\big(2^{\Theta(r+r(2+k))}\big) = \mathbf{ZPTIME}\big(2^{\Theta(\log n(\log\log n)^{3/2})}\big).$$

The number of vertices in the graph is

$$N = 2^{D(f+\log s^{-1})} = 2^{\Theta(\log n(\log\log n)^{3/2})}$$

and since $\log\log N = \Theta(\log\log n)$, we obtain $\beta > 1 - O(1/\sqrt{\log\log N})$.

□

Theorem 11.2. *Unless* **NP** $\subseteq$ **ZPTIME**$(2^{O(\log n(\log\log n)^{3/2})})$*, it is impossible to approximate Min Chromatic Number on a graph with n vertices within $n^{1-O(1/\sqrt{\log\log n})}$ in polynomial time.*

Proof. This follows when Theorem 11.1 is combined with Lemma 9.2. □

12 Future Work

An obvious improvement of this work would be to weaken the assumptions on **NP** we used in our hardness result. Best of all, of course, would be to construct deterministic reductions, since this would allow us to replace the probabilistic complexity classes with deterministic ones in all our assumptions on **NP**.

As discussed in Section 8, another way to improve the result would be to use a more efficient coding of the proof than the long code. This would make the number of random bits used increase slower as the gap increases, which would give us a stronger result.

Until this is done, an interesting open question is to determine the best definition of the amortized free bit complexity. We have proposed that the definition should be

$$\bar{f} = \frac{f + \log c^{-1}}{\log(c/s)}. \tag{27}$$

This definition works well in the sense that a PCP with one-sided error gives a hardness result for Max Clique under the assumption that **NP**-complete problems cannot be decided in expected slightly superpolynomial time, and similarly a PCP with two-sided error gives a hardness result for Max Clique under the assumption that **NP**-complete problems cannot be decided with two-sided error in probabilistic slightly superpolynomial time.

However, we have seen in Sec. 6.2 that if one wants to use a PCP with two-sided error to obtain hardness results under the assumption that **NP**-complete problems cannot be decided in expected slightly superpolynomial time, the interesting parameter is (close to) F_c, defined in equation 4. To establish whether it is possible to improve this to our proposed definition of $\bar{f}$, or if F_c is the best possible in this case is an interesting open question.

Trying to obtain an upper bound is also interesting, especially since it is currently unknown how well the Lovász ϑ-function approximates Max Clique and Min Chromatic Number. It has been shown by Feige [8] that it cannot

approximate Max Clique within $n^{1-O(1/\sqrt{\log n})}$, but, in light of the results of this paper, this does not compromise the Lovász ϑ-function very much. It may very well be that it beats the combinatorial algorithm of Boppana and Halldórsson [6] for Max Clique and Halldórsson's algorithm [13] for Min Chromatic Number.

13 Acknowledgments

We would like to thank Alex Samorodnitsky and Luca Trevisan for providing us with preliminary versions of their paper [22]. We would also like to thank Johan Håstad for useful discussions.

References

1. Noga Alon and Joel H. Spencer. *The Probabilistic Method*. John Wiley & Sons, New York, 1991.
2. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Márió Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.
3. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, January 1998.
4. Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs and non-approximability—towards tight results. *SIAM Journal on Computing*, 27(3):804–915, June 1998.
5. Mihir Bellare and Madhu Sudan. Improved non-approximability results. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 184–193, Montréal, Québec, Canada, 23–25 May 1994.
6. Ravi Boppana and Magnús M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *Bit*, 32(2):180–196, June 1992.
7. Lars Engebretsen and Jonas Holmerin. Clique is hard to approximate within $n^{1-o(1)}$. In Ugo Montanari, José D. P. Rolim, and Emo Welzl, editors, *Proceedings of 27th International Colloquium on Automata, Languages and Programming*, volume 1853 of *Lecture Notes in Computer Science*, pages 2–12, Geneva, 9–15 July 2000. Springer-Verlag.
8. Uriel Feige. Randomized graph products, chromatic numbers, and the Lovász ϑ-function. *Combinatorica*, 17(1):79–90, 1997.
9. Uriel Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, July 1998.
10. Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Márió Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, March 1996.
11. Uriel Feige and Joe Kilian. Zero knowledge and the chromatic number. *Journal of Computer and System Sciences*, 57(2):187–199, October 1998.
12. Martin Fürer. Improved hardness results for approximating the chromatic number. In *36th Annual Symposium on Foundations of Computer Science*, pages 414–421, Milwaukee, Wisconsin, 23–25 October 1995. IEEE.

13. Magnús M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Information Processing Letters*, 45(1):19–23, January 1993.

14. Johan Håstad. Some optimal inapproximability results. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 1–10, El Paso, Texas, 4–6 May 1997. Accepted for publication in *Journal of the ACM*.

15. Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182(1):105–142, 1999.

16. Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

17. Sanjeev Khanna, Nathan Linial, and Shmuel Safra. On the hardness of approximating the chromatic number. In *Proceedings of 2nd Israel Symposium on Theory of Computing and Systems*, pages 250–260, Natanya, Israel, 1993. IEEE Computer Society.

18. László Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13(4):383–390, December 1975.

19. Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981, September 1994.

20. Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, 1995.

21. Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998.

22. Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing*, pages 191–199, Portland, Oregon, 21–23 May 2000.

23. David Zuckerman. On unapproximable versions of NP-complete problems. *SIAM Journal on Computing*, 25(6):1293–1304, December 1996.