Electronic Colloquium on Computational Complexity, Report No. 128 (2016)

Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover

Irit Dinur
Weizmann Institute of Science

August 13, 2016

Abstract

We show that if gap-3SAT has no sub-exponential time algorithms then a weak form of the sliding scale conjecture holds. Namely, for every $\alpha > 0$ any algorithm for n^{α}-approximating the value of label cover must run in time at least $n^{\Omega(\exp(1/\alpha))}$, where n is the size of the instance.

Put differently, if there is a polynomial time algorithm for approximating the value of label cover to within a factor of approximation of $n^{o(1)}$ then gap-3SAT (and thus, random-3SAT) must have faster-than-exponential-time algorithms.

Our proof is a twist on the well-studied parallel repetition reduction from 3SAT to label cover. Our key observation is that if we take unordered repetition, replacing tuples by sets, then we can afford to take the number of repetitions to be linear in n, and generate an instance of size $\exp(n)$ which results in the claimed parameters.

1 Introduction

The PCP theorem [AS98, ALM+98] implies that every NP language can be checked by a verifier that makes very few queries into a proof, and uses randomness. A basic goal is to understand the tradeoff between the number of bits read by the verifier and the soundness error, which is the probability that the verifier accepts a false proof. Denoting by t the number of bits queried from the proof, the soundness error cannot go below 2^{-t}, because even a proof chosen uniformly at random will cause the verifier to accept with such probability[1].

How close to the random threshold can the soundness error be? For t a constant there is a developed theory pinpointing a precise threshold for the soundness error. This threshold depends on the precise predicate computed by the verifier (this threshold is established for some predicates, as pioneered in [Has01], and in others it is based on Khot's unique games conjecture [Kho02], see

[1]This is true as long as for every local view of the verifier there is at least one accepting configuration (such an assumption is guaranteed in the perfect completeness case, but is also reasonable otherwise). The probability to see this configuration in a random proof is at least 2^{-t}.

ISSN 1433-8092

[Rag08, AM09]). For larger $t = \omega(1)$ the territory is less understood and a coarser ballpark behavior must first be studied. A reasonable question is whether there are always PCP verifiers that read t bits from the proof and whose soundness error is a *constant power* of the random threshold, i.e. $\varepsilon \leqslant \exp(-t)$.

When $t = \omega(1)$ one must take into account both the number of bits t and the number of queries. Making t one-bit queries is very different from making a constant number of $\theta(t)$-bit queries. A t-query verifier is quite powerful and small soundness for it is easy to prove. In contrast, a $O(1)$-query verifier is more restricted, so proving soundness for it is more challenging. Such a verifier is much more useful though, e.g. it corresponds to multi-prover interactive proofs with *a constant* number of provers. It corresponds to hardness of *label cover* when the number of queries is two, or to hardness of k-CSPs when the number of queries is $2 < k \ll t$.

The *sliding scale conjecture* of [BGLR93] says that there is a PCP verifier that makes a constant number of queries, each query resulting in an answer whose length is $O(t)$ bits, has perfect completeness, and soundness error $\varepsilon \leqslant \exp(-t)$. Let us call such PCP verifiers, namely that have perfect completeness and soundness error that is exponentially small in the number of bits read from the proof, *sliding-scale verifiers*. The current work focuses on the range of t for which sliding scale verifiers exist. Jumping ahead, sliding scale verifiers have been known to exist for $t = o(\log n)$ and we are interested in the question of their existence when $t = \Omega(\log n)$.[2]

Sliding scale conjecture in terms of hardness of approximation. The sliding scale conjecture can be equivalently formulated in terms of hardness of approximation of constraint-satisfaction problems (CSPs) with a constant number of variables in each constraint. The soundness translates to the approximation factor, and the number of bits read from the proof translates to the logarithm of the alphabet size (i.e. the range of the variables). A PCP verifier that makes exactly two queries corresponds to *label-cover*[3] (see definitions in Section 2). The sliding scale conjecture in terms of label cover says that label cover with alphabet size $h(n)$ is hard to approximate to within a factor of $h(n)^{\Omega(1)}$. More generally in terms of CSPs the sliding scale conjecture is that it is hard to approximate the value of a CSP to within a factor that is a constant power of the alphabet size. In these terms this work is concerned with finding the largest possible hardness of approximation gap for CSPs and for label cover, while maintaining a polynomial relation between the alphabet size and the gap. In particular, is there a *polynomially large* hardness of approximation gap?

[2]The sliding scale conjecture is only meaningful for values of t up to the number of random bits used by the verifier, i.e. $t \leqslant O(\log n)$.

[3]Label cover has a *bipartite* graph structure, but there is an easy reduction from the non-bipartite to the bipartite case so this is without loss of generality.

Two queries or more. In terms of known results, there is a distinct difference between PCPs that make two queries, and those that make three queries or more. For three queries or more, sliding-scale PCP verifiers have been constructed [RS97, AS97, DFK$^+$11, DHK15] for all alphabet sizes up to $2^{(\log n)^\beta}$ for any constant $\beta < 1$. The conjecture is still quite open for polynomial alphabet sizes, i.e., for $h(n) = n^\alpha$ for some $\alpha > 0$.

The case of two queries, corresponding to hardness of label-cover, is more difficult to analyze. Here the only known way to construct "sliding-scale" PCP verifiers is based on a direct product construction, called parallel repetition. The parallel repetition reduction converts a gap-3SAT instance of size n to an instance of size n^t and soundness $exp(-t)$. The resulting instance is used often for tight hardness-of-approximation results (see, e.g. [Has01]) and is sometimes called *the Raz verifier* because the soundness error is bounded using Raz's parallel repetition theorem [Raz98]. This reduction maintains the desired inverse polynomial relation between the alphabet size and the soundness error, while keeping the number of queries two. Its only drawback is that it causes a super-polynomial blowup in the *instance size*. No matter what t we choose, even if we allow a super-polynomial increase in the instance size, the soundness error $\exp(-t)$ will never be inverse polynomial in the instance size $n^t = \exp(t \log n)$ (because $\exp(t)$ is not a constant power of n^t). So this reduction falls short of outputting an instance with a polynomial gap.

This state of affairs (both for two queries and for more) has been stuck for many years now, hinting that perhaps the sliding scale conjecture does not hold for polynomial gaps, and suggesting to look for an approximation algorithm with factor n^δ for all $\delta > 0$. Our result, in contrast, gives evidence in support of a polynomial-gap sliding scale conjecture, showing that such an algorithm would have very surprising implications for gap-3SAT. Specifically, we show an exponential-time reduction from gap-3SAT to label cover instances with a polynomial gap.

Our reduction is a twist on the well-studied parallel repetition reduction. The key observation is that if we take unordered repetition, replacing tuples by sets, then we can afford to take the number of repetitions t to be linear in n, and derive non-trivial sliding-scale PCP verifiers. Our main theorem is as follows (see definitions in Section 2),

Theorem 1.1. *Assume the exponential time hypothesis for gap-3SAT, namely,*

> *For some constant $c > 0$, any algorithm that is given a 3SAT formula Φ on n variables and $O(n)$ clauses, and decides between* $\mathrm{sat}(\Phi) = 1$ *and* $\mathrm{sat}(\Phi) < 0.9$ *must run in time at least 2^{cn}. (Here* $\mathrm{sat}(\Phi)$ *denotes the maximal fraction of satisfied clauses).*

Then, for every $\alpha > 0$ any algorithm that decides if a given label cover instance whose size is n, and whose alphabet size is at most n, has value 1 *or at most $n^{-\alpha}$ must run in* *time at least $n^{\exp(\Omega(1/\alpha))}$. Furthermore, one can assume that the label-cover has projection constraints.*

The theorem relates polynomial-gap label cover to a new hypothesis called gap-ETH. We discuss this hypothesis, how likely it is to hold, and how it relates to other well known hypotheses, in Section 2.2. Independently of this work, the same hypothesis is introduced in a recent paper of Manurangsi and Raghavendra [MR16] under the name ETHA. Our main result is proved in Section 3.

In Section 4, we explore whether Theorem 1.1 can be upgraded from relying on gap-ETH, which is a new hypothesis, to relying on the well-studied exponential time hypothesis of [IPZ01]. Our hope was to replace gap-3SAT, which serves as a starting point of our reduction, by one of the existing low-soundness PCP verifiers [RS97, AS97, DFK+11, MR10a, MR10b, DHK15]. However, our reduction falls slightly short of yielding meaningful parameters, essentially since the existing PCPs are not length-efficient enough. We conjecture that such length-efficient PCPs, which we call "linear-size sliding scale verifiers", do exist, for *any* gap ε.

For constant ε, the existence of a linear-size sliding scale verifier is equivalent to having a linear-size reduction from 3SAT to gap-3SAT, and would certainly suffice for our reduction. However, it seems particularly difficult to construct such efficient reductions in the $\varepsilon = O(1)$ regime because the reduction is not allowed to incur even a multiplicative $\log n$ factor. Potentially it could be easier to construct liner-size sliding scale verifiers for slightly smaller values of ε, where the initial $\log n$ factors would be swallowed by t. We show that the existence of "linear-size sliding scale verifiers" for *any* gap ε already implies the *polynomial-gap* sliding scale conjecture.

2 Preliminaries

We define label cover, the sliding scale conjecture, and the exponential time hypothesis for gap-3SAT.

2.1 Label cover and the sliding scale conjecture

Definition 2.1 (Label Cover)**.** A *label cover* instance is given by a bipartite graph G with vertex sets U and V; an alphabet Σ and a constraint $\pi_{uv} : \Sigma^2 \to \{0, 1\}$ per each edge (u, v). We allow parallel edges (and non-negative polynomially-bounded weights). The *value* of the instance is the maximum, over all labelings $f : U \to \Sigma$ and $g : V \to \Sigma$ of the fraction of satisfied constraints,

$$\mathrm{sat}(G) = \max_{f,g} \mathop{\mathbb{P}}_{uv\in E(G)} [\pi_{uv}(f(u), g(v)) = 1].$$

The gap problem label-cover$_n(1, \varepsilon)$ is the problem of deciding if a given label cover over n variables whose alphabet size is at most n has value 1 or at most ε.

A label cover instance is called "projection" if the constraints on each edge have a special projection form, in which the value to the first variable determines only one valid value to the second variable.

Label-cover is a special case of the following,

Definition 2.2 (CSP)**.** A *constraint satisfaction problem* (k-CSP) is a system of constraints over n variables. Each constraint is specified by a tuple $\mathbf{i} = (i_1, \ldots, i_k) \in [n]^k$ and a function $\psi : \Sigma^k \to \{0, 1\}$. The CSP is called k-ary since each constraint looks at at most k variables, and Σ is called the alphabet. The value of a CSP C is

$$\mathrm{sat}(C) = \max_{f:[n]\to\Sigma} \mathop{\mathbb{P}}_{(\mathbf{i},\psi)\in C} [\psi(f(i_1), \ldots, f(i_k)) = 1].$$

The gap problem $k\text{-CSP}^n_\Sigma(1, \varepsilon)$ is the problem of deciding if a given CSP over n variables taking values in Σ has value 1 or at most ε.

The Sliding Scale Conjecture. The sliding scale conjecture was suggested (somewhat implicitly) in the open problems section of [BGLR93]. It was made more explicit in subsequent works, e.g. appearing as Conjecture 3 in [RS97]. The conjecture has been proven for some of the parameter regime. We state here a version of it that pertains to polynomially small error probability, which is the most interesting, and wide open.

Conjecture 2.3 (Sliding Scale Conjecture with parameter ε)**.** *There exists some constant $k > 1$ and a polynomial-time reduction from SAT on n variables to k-$CSP^{\mathrm{poly}(n)}_{[\mathrm{poly}(1/\varepsilon)]}(1, \varepsilon)$.*

Moshkovitz [Mos12] has studied the following special case (i.e. stronger version) of this conjecture,

Conjecture 2.4 (Sliding Scale Conjecture - projection games variant)**.** *There exists a polynomial-time reduction from SAT on n variables to a label-cover instance with projection constraints, alphabet at most $1/\mathrm{poly}(\varepsilon)$, and a $(1, \varepsilon)$ gap.*

2.2 Exponential Time Hypothesis for gap-3SAT

The *Exponential Time Hypothesis* [IPZ01] says that any algorithm that, on input a 3SAT formula Φ on n variables, must take at least $2^{\gamma n}$ time to decide if it is satisfiable or not. We shall entertain the following strengthening of this hypothesis

Hypothesis 2.5 (gap-ETH)**.** *There are constants $c, D > 0$ such that any algorithm that, on input a 3SAT formula Φ on n variables and at most Dn clauses, can distinguish between $\mathrm{sat}(\Phi) = 1$ and $\mathrm{sat}(\Phi) < 0.9$, must run in time at least 2^{cn}.*

Several remarks are in order

– PCP: Hypothesis 2.5 with slightly weaker parameters follows from the exponential time hypothesis [IPZ01]. This is a consequence of the length-efficient PCP theorems [Din07, BS05] and the following claim,

Claim 2.6. Suppose there is a reduction that runs in sub-exponential time and converts a 3SAT instance on n variables into a gap-3SAT instance on $n \cdot \ell(n)$ variables. Then, assuming ETH, no algorithm for 3SAT can run in time less than $2^{\Omega(n/\ell(n))}$.

In particular, since we know a length-efficient PCP theorem [Din07, BS05], which gives a reduction as above with $\ell(n) = \operatorname{poly}\log(n)$, we deduce, assuming the ETH, that any algorithm for solving gap-3SAT must run in time at least $2^{n/\operatorname{poly}\log(n)}$. Moreover, if one were to prove an even stronger PCP theorem in that it has a linear blowup in size (even if the reduction itself were to run for sub-exponential time), then this would equate ETH with gap-ETH. (This stronger PCP theorem is a special case of a conjecture discussed in Section 4).

– Feige's R3SAT: Hypothesis 2.5 follows from an exponentially-hard version of Feige's R3SAT hypothesis. To see why recall that a random 3SAT formula with Dn clauses (obtained by choosing each clause independently at random) has value very close to $7/8 < 0.9$. Thus, the task of distinguishing a random 3SAT from a satisfiable one is a special case of gap-ETH. This task is exactly the subject of Feige's R3SAT hypothesis [Fei02]. Feige hypothesized that this task is "hard" (although he did not explicitly require it to be exponential-time hard). Currently there is no known algorithmic difference between random 3SAT and worst-case 3SAT in terms of the time it takes to decide if the formula is satisfiable or not.

– Raghavendra and Manugrasi [MR16] in recent independent work introduced the same hypothesis, under the name ETHA. They used this hypothesis to prove tight results on hardness of dense CSPs. They point out that the $\Omega(n)$ Lasserre hierarchy cannot distinguish between a satisfiable 3SAT formula and one that is only $7/8 + \varepsilon$ satisfiable, as shown in [Sch08].

– Goldreich's One Way Function: Hypothesis 2.5 might be weaker than Goldreich's one-way-function assumption, although we don't know how to prove this at this point. Goldreich [Gol11] suggested that a certain distribution of CSPs is a one-way function, and Applebaum [App13] showed that this assumption implies that a certain distribution of CSPs is a pseudorandom generator with linear stretch. Had Applebaum's reduction worked for exponential hardness (currently it is only known for super-polynomial hardness) it would mean that a sub-exponential algorithm for gap-3SAT can distinguish a pseudo-random output from a random one and thus break Goldreich's assumption.

We remark that the assumption regarding the number of clauses being a constant multiple of n is for convenience only. This is without loss of generality since if the hypothesis were true for some larger clause-variable ratio, then a sparsification argument would imply it is true as written above. The reason is that a randomly chosen set of Dn clauses is an instance with roughly the same value with very high probability. (Note however that the sparse instance is randomly constructed so the implication would be only for randomized algorithms). We give a formal proof of this claim in a slightly more generalized setting, which will be useful for a later section.

Claim 2.7 (Sparsification). Let Φ be a k-ary CSP with m constraints over alphabet Σ and let $\beta > 0$. Assume $\mathrm{val}(\Phi) < \varepsilon$ for ε such that $|\Sigma| = 1/\varepsilon^{\beta}$. Let Φ' be obtained from Φ by selecting each constraint to participate in Φ' independently with probability t/m where $t = \beta n/\varepsilon^2 \log 1/\varepsilon$. Then with high probability $\mathrm{val}(\Phi') < 2\varepsilon$.

Proof. This is a standard union bound argument. For each assignment, at most εm of the constraints are satisfied. When we select Φ', we expect to see εt satisfied constraints. The probability of seeing $2\varepsilon t$ or more is bounded, using the Chernoff-Hoeffding bound, by

$$\mathbb{P}\left[\text{number of unsat constraints} > 2\varepsilon t\right] < 2\exp(2t\varepsilon^2) < 2\varepsilon^{2\beta n}$$

where the last inequality is due to the choice of t as a function of β and ε. The number of possible assignments is $\Sigma^n = \varepsilon^{-\beta n}$ and thus with high probability none of them satisfy more than $2\varepsilon t$ constraints in Φ'. □

2.3 Binomial coefficient estimate

We rely on the following estimate, for large n and constant $\delta > 0$,

$$\binom{n}{\delta n} \approx 2^{nH(\delta)} \approx 2^{\delta n \log(1/\delta)}$$

where $\approx$ implies up to $(1 \pm o(1))$ multiplicative factors in the exponent, and $H(\delta) = -\delta \log_2 \delta - (1-\delta)\log_2(1-\delta)$ is the entropy function.

3 Main Lemma

Our main theorem follows from the following lemma,

Lemma 3.1. *For every $\delta > 0$ there is reduction mapping a 3SAT instance Φ on n variables and $m = D \cdot n$ clauses to a label cover instance U of size $N = 2^{O(n\cdot\delta\log(D/\delta))}$ such that*

- *If* $\mathrm{sat}(\Phi) = 1$ *then* $\mathrm{sat}(U) = 1$
- *If* $\mathrm{sat}(\Phi) < 0.9$ *then* $\mathrm{sat}(U) < 2^{-\Omega(\delta n)} = N^{-1/\Omega(\log(D/\delta))}$

The reduction runs in time that is at most quadratic in the output size N.

Let us first derive the main theorem from the lemma, and then prove the lemma.

Proof of Theorem 1.1. Assume the ETH for gap-3SAT holds with constants $c, D > 0$. Let Φ be a 3SAT formula on n variables and $m = Dn$ clauses.

Fix $\alpha > 0$ and let $\delta = D \cdot 2^{-1/\alpha}$ (so that $\log \frac{D}{\delta} = \frac{1}{\alpha}$). Let $N = 2^{c_1 \cdot n\delta/\alpha}$ where c_1 is the constant hidden in the $O(\cdot)$ notation of Lemma 3.1. By the completeness and soundness of Lemma 3.1 an N^k-time algorithm for deciding if the value of U is 1 or at most $N^{-\alpha}$ will distinguish the case that the value of Φ is 1 from the case it is at most 0.9. Hence it must be that $2^{cn} < N^k \leqslant (2^{c_1 n\delta/\alpha})^k$. This means that $k\delta c_1/\alpha > c$, that is, $k > c\alpha 2^{1/\alpha}/Dc_1 = \exp(\Omega(1/\alpha))$. □

Proof of Lemma. The reduction can be summarized as follows: transform Φ into a two-player game, apply parallel repetition with $t = \delta n$ rounds, and derive from it a label-cover instance whose vertices are the *unordered* question-tuples. One has to take care about implementing the construction in time polynomial in the output size. Details follow. First, let us assume wlog[4] that each variable in Φ participates in the same number of clauses. Next, we take the three steps outlined above.

1. Defining a two player game (standard): A 3SAT formula Φ gives rise to the following "clause vs. variable" player game: the referee selects a variable x uniformly at random, and then selects a random clause c that contains x. The referee sends the clause c to the clause player and the variable x to the variable player. The clause player answers a number $a \in [7]$ which is interpreted as one of seven possible assignments satisfying the clause. The variable player outputs a bit assignment for the variable x. The game is won if the answer of the clause player is consistent with the answer of the variable player on x. Let us call this game G_Φ. It is clear that $\text{sat}(\Phi) = 1$ implies $val(G_\Phi) = 1$, whereas $\text{sat}(\Phi) < 0.9$ implies $\text{val}(G_\Phi) < 0.99$.

2. Parallel repetition (standard): The t-parallel-repetition of the above game is the game in which the referee selects t variables $x_1, \ldots, x_t$ independently at random and t clauses $(c_1, \ldots, c_t)$ such that $c_i \ni x_i$. The answer of each player is now a t-tuple of answers. By definition, the game is won if all t answers of the clause player satisfy the t clauses and are consistent with all t answers of the variable player. We denote this game by G_Φ^t. Clearly if $\text{val}(G_\Phi) = 1$ then $\text{val}(G_\Phi^t) = 1$, and by the parallel repetition theorem [Raz98] if $\text{val}(G_\Phi) < 0.99$ then $\text{val}(G_\Phi^t) < 2^{-\Omega(t)}$. Note that the constant hidden in the exponent is really an absolute constant (it only depends on 0.99).

   It is important that this step is only conceptual and not really part of a construction. In particular, the standard way of making a label cover out of a parallel repetition instance would result in too large an instance size.

3. Label–cover (idealized weights): Here is where we crucially make a non-standard move. Rather than considering a graph in which the vertices are

[4] There is a well-known transformation for converting Φ into such form (by so-called "expander replacement") while increasing the number of variables at most linearly and losing a negligible amount in $\text{val}(\Phi)$.

all possible ordered t-tuples, we consider a graph in which the vertices are all possible multi-sets of size t.

We construct a label-cover instance. The underlying bipartite graph $U = (X, C, E)$ has a vertex set $C \cup X$, where C has a vertex c_S for every multiset S of t clauses, and X has a vertex x_T for every multiset of t variables. The number of vertices is thus[5] $\binom{n+t-1}{t} + \binom{m+t-1}{t}$.

We put the edge (c_S, x_T) with weight given according to the probability of getting the pair (S, T) in following random process: Select a random tuple of variables $(x_1, \dots, x_t)$ and then a random tuple of clauses $c = (c_1, \dots, c_t)$ such that $c_i \ni x_i$. Output $S = \{\{c_1, \dots, c_t\}\}$ and $T = \{\{x_1, \dots, x_t\}\}$ where the notation $\{\{\cdot\}\}$ stands for a multiset.

The label-set is $[7]^t$ for C and $[2]^t$ for X. A labeling $L : X \to [2]^t, C \to [7]^t$ is interpreted as assigning each variable in x_T a Boolean value, and each clause in c_S a number in[6] $[7]$. The constraint on an edge (c_S, x_T) will check all the relevant consistency checks, that is, for every pair of variable $x \in T$ and clause $c \in S$ such that $x \in c$ it checks that the assignment for c_S gives the same value to x as the assignment for x_T.

Clearly if $\mathrm{sat}(\Phi) = 1$ then $\mathrm{sat}(U) = 1$. We claim that $\mathrm{sat}(U) \leqslant \mathrm{sat}(G^t_\Phi)$. This is because every labelling $L : C \to [7]^t, X \to [2]^t$ specifies a strategy for the players in the two player game. The value of this strategy lower bounds the value of the labelling, because in the game only "aligned" constraints are checked whereas in the label-cover constraints across different coordinates are checked as well. In particular, if an edge (c_S, x_T) in U is satisfied by the labelling then any question pair in G^t_Φ that is obtained by an ordering of S and T will be satisfied by the corresponding strategy.

4. Efficiently implementing the weights: The whole point of this construction is the super-polynomial savings we obtain by replacing all t-tuples by all t-multisets. This reduces the size from n^t to $\binom{n+t-1}{t}$ which is the number of distinct t-multi-sets. Thus, it is crucial that we can implement this construction, and in particular the weights of the edges, in time that is polynomial in $\binom{n+t-1}{t}$ rather than n^t.

   For a multiset S, the weight of c_S is the number of tuples that resulted in c_S, denoted $w(S)$. This number clearly computable in time $\mathrm{poly}(t)$. Let us think of the multiset S as a list $(c_1, r_1), \dots, (c_m, r_m)$ where $r_i \geqslant 0$ indicates the number of times the i-th clause is repeated, and $\sum_i r_i = t$. We enumerate over all neighboring multisets T of variables in time $\exp(t)$ as follows. For each $\sigma \in [3^t] = \prod_i [3^{r_i}]$ we think of $\sigma = (\sigma_1, \dots, \sigma_m)$ where $\sigma_i \in \{1, 2, 3\}^{r_i}$ is a list of r_i indices indicating which variables of the i-th clause are included in

[5] This can be easily seen through a bars and stars argument.

[6] This can be done by fixing an order on the clauses and then sorting the clauses in S according to this order: $(C_1, C_2, \dots, C_{|S|})$; then viewing a label $(\ell_1, \dots, \ell_t) \in [7]^t$ as giving C_i the value ℓ_i. Similarly for the variables.

T. For each $i = 1, \dots, m$ add into T the variables of the i-th clause indexed by σ_i. For each such pair (c_S, x_T) place an edge with weight $w(S)$ (calculated above). If a pair occurs more than once (this happens if the multiset T happens to occur more than once as a neighbor of S), then we add up the weights.

To conclude, if $\mathrm{sat}(\Phi) < 0.9$, then we get $\mathrm{sat}(U) \leqslant \mathrm{sat}(G^t_\Phi) \leqslant 2^{-\Omega(\delta n)}$. The size of U, denoted N, is polynomial in the size of C, and denoting $m' = \delta m + \delta n$,

$$|C| = \binom{m'}{\delta n} < \binom{m'}{m' \cdot \frac{\delta}{D}} \approx 2^{m' \frac{\delta}{D} \log(D/\delta)} \leqslant 2^{2n\delta \log(D/\delta)},$$

where the approximate equality is up to $(1 + o(1))$ in the exponent.

Expressing $\varepsilon = 2^{-\delta n}$ in terms of N gives $\varepsilon = 1/N^{1/\log(D/\delta)}$. □

4 Linear-size sliding-scale verifiers

In this section we show that sliding scale verifiers for smaller-than-polynomial gaps can be used to obtain, via our reduction, a sliding scale verifier with a polynomial gap. Our initial hope was that we could rely on pre-existing sliding scale verifiers [RS97, AS97, DFK+11, DHK15] as a starting point of the reduction[7], instead of gap-3SAT. This would place our conclusion about polynomial gaps for label cover on firmer ground, replacing gap-ETH with a more conventional hypothesis such as ETH.

However, it turns out that our reduction is not strong enough to utilize existing sliding scale verifiers, because they are not length-efficient. There is a line of work [MR10a, MR10b] which tries to minimizes all three parameters (soundness error, alphabet size, proof length) but unfortunately it does not meet our needs. In particular, it does not have a polynomial relation between the gap and the alphabet size.

We propose the following conjecture regarding the existence of linear size sliding scale verifiers. The conjecture is parameterized by ε, which is thought of as some function of n. The point is (see Lemma 4.4 below) that existence of a linear sliding scale verifier for *some* ε, even much larger than $1/\mathrm{poly}(n)$, will imply the sliding scale conjecture for $\varepsilon = 1/\mathrm{poly}(n)$.

Conjecture 4.1 (ε-linear-sliding-scale conjecture - CSP phrasing)**.** *There exists some $k > 1$ and a polynomial-time reduction from SAT on n variables to a k-CSP on $n/\mathrm{poly}(\varepsilon)$ variables taking values in an alphabet of size $1/\mathrm{poly}(\varepsilon)$, and a $(1, \varepsilon)$ gap.*

Let us also include the equivalent phrasing in terms of PCP verifiers for NP,

Conjecture 4.2 (ε-linear-sliding-scale conjecture - PCP verifier phrasing)**.** *There exists some $k > 1$ such that every language in NP can be verified by a polynomial-time*

[7]Another issue that would need to be addressed in this approach is the need of a parallel repetition theorem for more than two players.

verifier that makes k queries into a proof written over an alphabet of size $\mathrm{poly}(1/\varepsilon)$*. The verifier uses* $\log n + O(\log \frac{1}{\varepsilon})$ *random bits to decide on the queries, has perfect completeness, and soundness error at most* ε*.*

We will sketch a reduction (very similar to the one in Lemma 3.1) that shows that if the conjecture above held true for some value of ε, then by unordered parallel repetition we would get a result analogous to Theorem 1.1 based on the exponential-time hypothesis for SAT rather than for gap-SAT. This is a weaker, and much more widely studied assumption.

First, let us give a quick comparison between the verifier in the conjecture and currently known constructions.

– When $\varepsilon = O(1)$ if we consider a constant number of queries then the best known proof length is $n \cdot \mathrm{poly}\log n$ [Din07, BS05]. Linear proof length is achieved in [BKK+13] but with a number of queries that is n^{β} for some $\beta > 0$.

– When $\varepsilon = 2^{-(\log n)^{\alpha}}$ for some $\alpha > 0$ two constructions are known. The first, [RS97, AS97], constructs sliding scale verifiers whose proof length is much larger than $n/\mathrm{poly}(\varepsilon)$. The second, [MR10a], comes closer to this goal and constructs verifiers with soundness ε but whose alphabet size is $1/\mathrm{qpoly}(\varepsilon)$ and whose proof length and is $n/\mathrm{qpoly}(\varepsilon)$ (where $\mathrm{qpoly}(x)$ denotes quasi-polynomial in x, i.e. $x^{\mathrm{poly}(\log x)}$).

Reduction 4.3 (Unordered parallel repetition from a k-query PCP verifier)**.** Suppose there exists a linear size sliding scale verifier that makes k queries. We construct a new k-CSP instance as follows. We view the PCP verifier system as a k-CSP, and sparsify by leaving a random set of n/ε^3 constraints, as in Claim 2.7. The new instance is constructed as follows. The new variables $\{x_S\}$ will correspond to subsets S of old variables, such that $|S| \leqslant t$. The new constraints and their weights are described by the following random process: Run the verifier t times independently generating k t-tuples of variables. From each tuple "erase" the order (say by sorting) and send it to one of the k provers.

The reduction clearly has perfect completeness. For $k = 2$ we can also prove soundness based on the parallel repetition theorem for small soundness [DS13, BG15]. For $k > 2$ a parallel repetition theorem is not known, let alone for small soundness, but we believe such a theorem is correct. We summarize what we can prove in the following lemma.

Lemma 4.4. *Reduction 4.3 can be computed in time polynomial in the output size, which itself is polynomially bounded by* $\binom{n/\varepsilon^3}{t} \leqslant \exp(O(\delta \cdot n/\varepsilon^3 \cdot \log 1/\delta))$.

– *Completeness: The reduction is complete for all k. Namely, if the initial instance is perfectly satisfiable, then so is the new instance.*

– *Soundness: If* $k = 2$ *then the reduction is sound. Namely, if the initial instance has value at most* ε *then the resulting instance has value at most* $\exp(-\Omega(\delta n/\varepsilon^3 \cdot \log 1/\varepsilon)) = \mathrm{poly}(1/N)$.

Proof. The size of the output instance is dominated by the number of constraints which is at most $N = \binom{n/\varepsilon^3}{t}$. We omit details of approximating the weights but this can be done similarly to Lemma 3.1.

We now bound the value of the constructed instance, in case we started from a SAT instance whose value was smaller than ε. For the case of $k = 2$ we can apply the low value parallel repetition theorem [DS13, BG15], then by similar calculations to those in Lemma 3.1, the value of the new instance would be at most

$$\varepsilon^{\Omega(t)} = \varepsilon^{\Omega(\delta m)} = \exp(-\Omega(\delta n/\varepsilon^3 \cdot \log 1/\varepsilon)),$$

which is polynomial in the output size N so long as $\delta = \text{poly}(\varepsilon)$. $\square$

We conclude that Conjecture 4.1 implies (at least for $k = 2$), assuming the "standard" exponential time hypothesis, that for some $\alpha > 0$ there is no polynomial-time algorithm that n^α approximates the value of a binary CSP with alphabet size n.

The interesting point in our opinion is that a sliding scale verifier with polynomial gap can be obtained from a sliding scale verifier with any gap ε, but only if the latter has size that is *linear* in n and polynomial in $1/\varepsilon$. This points towards the question of constructing such verifiers.

5 Discussion

It is not known whether gap-3SAT is exponentially hard, but currently there is no known algorithmic difference between random 3SAT and worst-case 3SAT in terms of the time it takes to decide if the formula is satisfiable or not. If we think that the sliding-scale conjecture is "very" false in that there is a polynomial-time approximation for the value of label cover with an $n^{o(1)}$ factor of approximation, then our result shows that random-3SAT can be refuted in faster-than-exponential time.

6 Acknowledgements

I wish to thank Oded Goldreich for encouraging me to write this paper, and for his many illuminating and helpful comments, including the suggestion to consider multisets instead of subsets, which simplified the analysis. Thanks also to Uriel Feige for an early discussion.

References

[ALM+98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. 1

[AM09] Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *Computational Complexity*, 18(2):249–271, 2009. 2

[App13] Benny Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM J. Comput.*, 42(5):2008–2037, 2013. 6

[AS97] Sanjeev Arora and Madhu Sudan. Improved low degree testing and its applications. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 485–495, El Paso, Texas, 4–6 May 1997. 3, 4, 10, 11

[AS98] S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. 1

[BG15] Mark Braverman and Ankit Garg. Small value parallel repetition for general games. In *Proc. 47th ACM Symp. on Theory of Computing*, pages 335–340, 2015. 11, 12

[BGLR93] M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient multi-prover interactive proofs with applications to approximation problems. In *Proc. 25th ACM Symp. on Theory of Computing*, pages 113–131, 1993. 2, 5

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal on Computing*, 27(3):804–915, June 1998.

[BKK$^+$13] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant rate pcps for circuit-sat with sublinear query complexity. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 320–329, 2013. 11

[BS05] Eli Ben-Sasson and Madhu Sudan. Simple PCPs with poly-log rate and query complexity. In *Proc. 37th ACM Symp. on Theory of Computing*, pages 266–275, 2005. 5, 6, 11

[DFK$^+$11] Irit Dinur, Eldar Fischer, Guy Kindler, Ran Raz, and Shmuel Safra. PCP characterizations of NP: Toward a polynomially-small error-probability. *Computational Complexity*, 20(3):413–504, 2011. 3, 4, 10

[DHK15] Irit Dinur, Prahladh Harsha, and Guy Kindler. Polynomially low error PCPs with polyloglog n queries via modular composition. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 267–276, 2015. 3, 4, 10

[Din07] Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3), 2007. 5, 6, 11

[DS13] Irit Dinur and David Steurer. Analytical approach to parallel repetition. *CoRR*, abs/1305.1979, 2013. 11, 12

[Fei02] Uriel Feige. Relations between average case complexity and approximation complexity. In *IEEE Conference on Computational Complexity*, page 5, 2002. 6

[Gol11] Oded Goldreich. Candidate one-way functions based on expander graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 76–87. 2011. 6

[Has01] Johan Hastad. Some optimal inapproximability results. *Journal of ACM*, 48:798–859, 2001. 1, 3

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. 4, 5

[Kho02] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775. ACM Press, 2002. 1

[Mos12] Dana Moshkovitz. The projection games conjecture and the np-hardness of ln n-approximating set-cover. In *Proceedings of RANDOM*, pages 276–287, 2012. 5

[MR10a] Dana Moshkovitz and Ran Raz. Sub-constant error probabilistically checkable proof of almost-linear size. *Computational Complexity*, 19(3):367–422, 2010. 4, 10, 11

[MR10b] Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5), 2010. 4, 10

[MR16] P. Manurangsi and P. Raghavendra. A Birthday Repetition Theorem and Complexity of Approximating Dense CSPs. *ArXiv e-prints*, July 2016. 4, 6

[Rag08] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proc. 40th ACM Symp. on Theory of Computing*, pages 245–254, 2008. 2

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998. 3, 8

[RS97] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In

Proc. 29th ACM Symp. on Theory of Computing, pages 475–484, 1997. 3, 4, 5, 10, 11

[Sch08] Grant Schoenebeck. Linear level lasserre lower bounds for certain k-csps. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 593–602, 2008. 6

ECCC ISSN 1433-8092

http://eccc.hpi-web.de