Sensitivity Lower Bounds for Approximation Algorithms

Noah Fleming*
Memorial University
nfleming@mun.ca

Yuichi Yoshida†
National Institute of Informatics
yyoshida@nii.ac.jp

November 22, 2024

Abstract

Sensitivity measures how much the output of an algorithm changes, in terms of Hamming distance, when part of the input is modified. While approximation algorithms with low sensitivity have been developed for many problems, no sensitivity lower bounds were previously known for approximation algorithms. In this work, we establish the first polynomial lower bound on the sensitivity of (randomized) approximation algorithms for constraint satisfaction problems (CSPs) by adapting the probabilistically checkable proof (PCP) framework to preserve sensitivity lower bounds. From this, we derive polynomial sensitivity lower bounds for approximation algorithms for a variety of problems, including maximum clique, minimum vertex cover, and maximum cut.

Given the connection between sensitivity and distributed algorithms, our sensitivity lower bounds also allow us to recover various round complexity lower bounds for distributed algorithms in the LOCAL model. Additionally, we present new lower bounds for distributed CSPs.

*Research supported by NSERC.
†Supported by JSPS KAKENHI Grant Number 24K02903.

Contents

1 Introduction **1**
- 1.1 Proof Overview . . . 3
- 1.2 Discussions . . . 4
- 1.3 Related Work . . . 6

2 Preliminaries **6**

3 Sensitivity-Preserving Reductions **8**

4 Lower Bounds for E2LIN **9**

5 PCPs and Sensitivity **10**
- 5.1 Degree Reduction . . . 10
- 5.2 Expanderization . . . 13
- 5.3 Gap Amplification . . . 14
  - 5.3.1 Graph Powering . . . 14
  - 5.3.2 Sensitivity Increase in the Recovery Procedure . . . 16
  - 5.3.3 Gap Analysis . . . 17
  - 5.3.4 Sensitivity Recovery . . . 22
- 5.4 Alphabet Reduction . . . 22
  - 5.4.1 Assignment Tester . . . 22
  - 5.4.2 Reduction Procedure . . . 25
  - 5.4.3 Sensitivity Increase in the Recovery Procedure . . . 25
  - 5.4.4 Gap Analysis . . . 27
  - 5.4.5 Sensitivity Recovery . . . 28
- 5.5 Proof of Theorem 5.2 . . . 28

6 Maximum Clique and Related Problems **29**
- 6.1 Lower Bounds for $(1-\varepsilon)$-Approximation Algorithms . . . 29
- 6.2 Lower Bounds for $n^{-\varepsilon}$-Approximation Algorithms . . . 30

7 Max CSPs **33**

8 Distributed Algorithms **34**

References **36**

1 Introduction

The notion of algorithmic sensitivity, introduced by Varma and Yoshida [VY23], measures the stability of an algorithm's output in Hamming distance (or the earth mover's distance if the algorithm is randomized) when an element in the input is added or deleted. In practical situations, low sensitivity is desirable because inputs can easily change due to noise or over time. If an algorithm has high sensitivity, it may lose reproducibility, erode user trust, and lead to inconsistent decisions. Since this concept was introduced, low-sensitivity algorithms have been developed for many graph problems. Some representative results include a 2-approximation algorithm for the minimum vertex cover problem with sensitivity $O(1)$ [CHHK16, VY23], a $(1-\varepsilon)$-approximation algorithm for the maximum matching problem with sensitivity $\Delta^{O(1/\varepsilon)}$ [YZ24], where Δ is the maximum degree, an additive $O(n^{2/3})$-approximation algorithm for the minimum s-t cut problem with sensitivity $O(n^{2/3})$, where the output is a vertex set [VY23], and a $(1+\varepsilon)$-approximation algorithm for the shortest path problem with sensitivity $O(\varepsilon^{-1}\log^3 n)$ (with respect to edge contractions instead of additions or deletions) [KY23]. Moreover, it has been shown that algorithms with low sensitivity can be used to design online algorithms with small recourse [YI22], as well as online learning algorithms with low regret [DY24]. Therefore, it is important to understand whether low sensitivity algorithms can be achieved for each problem.

On the lower bound side, the only known general result that we are aware of is that any randomized algorithm for finding a proper 2-coloring of a bipartite graph with n vertices requires sensitivity $\Omega(n)$ [VY23], which follows easily from the fact that a connected bipartite graph has only two proper 2-colorings such that the Hamming distance between them is $\Omega(n)$. Indeed, there are no known lower bounds on the sensitivity of approximation algorithms, leaving a significant gap in our knowledge. This is especially true, since all of the known upper bounds are derived from approximation algorithms.

In this work, we show polynomial sensitivity lower bounds for (even randomized) approximation algorithms for various combinatorial problems by adapting the probabilistically checkable proofs (PCP) framework [ALM+98, AS98]. We emphasize that this is the first time that lower bounds for *approximation* algorithms have been established. We are able to derive lower bounds for a variety of combinatorial problems through reductions, and we present some representative results next.

We start with the maximum clique problem. Consider an (inefficient) $n^{-\varepsilon}$-approximation algorithm that outputs a clique of size at most $n^{1-\varepsilon}$. Its sensitivity is trivially bounded by $O(n^{1-\varepsilon})$ and this is the only known upper bound. We show that the polynomial dependency on n is necessary:

Theorem 1.1. There are universal constants $\varepsilon, \delta > 0$ such that any algorithm for the maximum clique problem that outputs an $n^{-\varepsilon}$-approximate clique with probability $1 - O(1/n)$ has sensitivity $\Omega(n^{\delta})$.

We note that the lower bound also applies to the maximum independent set problem. Additionally, our proof is information-theoretic, and the lower bounds presented in this work apply even to inefficient algorithms.

Next, we consider the minimum vertex cover problem. As we mentioned, there exists a 2-approximation algorithm for the minimum vertex cover problem with sensitivity $O(1)$ [CHHK16, VY23]. We prove that the sensitivity must increase significantly as the approximation ratio approaches one.

Theorem 1.2. There are universal constants $\varepsilon, \delta > 0$ such that any (possibly randomized) $(1+\varepsilon)$-approximation algorithm for the minimum vertex cover problem has sensitivity $\Omega(n^{\delta})$.

It is known that computing an $\sqrt{2}$-approximate vertex cover is NP-hard [Kho02, SMS18]. Thus, Theorem 1.2 implies that achieving a small approximation ratio not only makes the problem computationally hard but also makes it impossible to achieve low sensitivity.

Finally, we discuss the maximum cut problem, where the output is a vertex set. Trivially, the random assignment algorithm is a $1/2$-approximation algorithm with sensitivity zero, and we can obtain an (inefficient) additive $\widetilde{O}(\sqrt{\varepsilon^{-1}nm})$-approximation algorithm with sensitivity $O(\varepsilon n)$ by converting a differentially private algorithm [EKKL20]. As mentioned, the only known lower bound is that exact algorithms, which can solve 2-coloring, must have sensitivity $\Omega(n)$ [VY23]. We show that the sensitivity must remain polynomial as the approximation ratio approaches one:

Theorem 1.3. There exist universal constants $\varepsilon, \delta > 0$ such that any (possibly randomized) $(1-\varepsilon)$-approximation algorithm for the maximum cut problem has sensitivity $\Omega(n^{\delta})$.

We can obtain similar lower bounds for other constraint satisfaction problems (CSPs) including E3SAT and 3LIN.

Although the notion of sensitivity is interesting in its own right, it has a close connection to distributed algorithms in the LOCAL model of distributed computing [Lin92]. In this model, a network is represented as a graph $G = (V, E)$, where each vertex $v \in V$ corresponds to an agent, and each edge $e \in E$ represents a communication link. Communication occurs in synchronous rounds. In each round, every vertex $v \in V$ receives messages from its neighbors, performs some local computation, and sends messages of arbitrary size to its neighbors. Note that, in t rounds, a vertex can compute its output based on the topology of its t-hop neighborhood. The goal of a distributed algorithm is for the outputs of the vertices to form a feasible solution to the problem.

We can show that if there exists a distributed algorithm in the LOCAL model for a graph problem that runs in t rounds, then there is an algorithm for the problem with sensitivity $O(\Delta^t)$, where Δ is the maximum degree of the graph. Building on this connection, we can recover various lower bounds on round complexity, including an $\Omega(1/\varepsilon)$ lower bound for an $n^{-\varepsilon}$-approximation algorithm for the maximum independent set problem [BHKK16], an $\Omega(\log n)$ lower bound for a $(1+\varepsilon)$-approximation algorithm for the minimum vertex cover problem for some constant $\varepsilon > 0$ [GS14, FFK22], and an $\Omega(\log n)$ lower bound for a $(1-\varepsilon)$-approximation algorithm for the maximum cut problem for some constant $\varepsilon > 0$ [CL23]. We note that these lower bounds match the known upper bounds [CL23].

Moreover, we can use this observation to establish lower bounds for distributed constraint satisfaction problems (DCSPs) [YIDK92], which can model tasks such as resource allocation and scheduling in distributed settings, including wireless networks (see the surveys [YH00, FPY18] and references therein). In DCSPs, each variable and constraint is associated with an agent. The agent for a variable can communicate with the agent for any constraint involving that variable, and vice versa. The goal of a distributed algorithm is for the assignment produced by the variables to maximize the number of satisfied constraints. We can immediately establish a round complexity lower bound of $\Omega(\log n)$ for specific CSPs, such as E3SAT and 3LIN, using the corresponding sensitivity lower bounds. Moreover, we show that there exists an arbitrarily hard CSP in the distributed setting:

Theorem 1.4. There exists an $O(\log\log n)$-ary CSP such that any distributed algorithm for the CSP that outputs an $\omega(1/\log n)$-approximate solution with probability $1 - O(1/n)$ requires $\Omega(\log n/\log\log n)$ rounds.

We believe this result is interesting because, apparently, it cannot be obtained by applying a gadget reduction to known lower bounds. We hope this work sheds new light on the study of distributed approximation algorithms for graph problems and CSPs.

1.1 Proof Overview

An instance of a *constraint satisfaction problem (CSP)* is a tuple $(V, E, \Sigma, \mathcal{R} = \{R_e\}_{e \in E})$, where (V, E) is a hypergraph, Σ is a finite domain (often referred to as the *alphabet*), and R_e is a $|e|$-ary relation over Σ, i.e., $R_e \subseteq \Sigma^{|e|}$. We say that an assignment $\sigma : V \to \Sigma$ *satisfies* a constraint on $e = (v_1, \ldots, v_k)$ if $(\sigma(v_1), \ldots, \sigma(v_k)) \in R_e$. For $1 \geq c \geq s \geq 0$, we define $\mathsf{MaxCSP}_{c,s}$ as the problem where, given a c-satisfiable CSP instance, i.e., there exists an assignment that satisfies at least a c-fraction of the constraints, the goal is to compute an s-satisfying assignment for the instance, i.e., an assignment that satisfies at least an s-fraction of the constraints. Here, c and s are referred to as the *completeness* and *soundness* parameters.

We first present a sensitivity lower bound for the case where the gap between completeness and soundness is extremely small. Specifically, we show that any algorithm for $\mathsf{MaxCSP}_{1,1-\Theta(1/n)}$ requires sensitivity of $\Omega(n)$ for $\mathsf{E2LIN}$ on cycles, where each constraint is of the form $x+y = 0 \pmod 2$ or $x+y = 1 \pmod 2$.

Next, we gradually increase the gap in multiple rounds without reducing the sensitivity lower bound. To do so, we adapt the probabilistically checkable proof (PCP) framework [AS98, ALM+98], which was originally developed as a characterization of NP and have been useful for proving inapproximability results. More specifically, we build on Dinur's PCP construction [Din07], which consists of four steps: degree reduction, expanderization, gap amplification, and alphabet reduction. Below, we explain each step and describe how we modify them to ensure the sensitivity lower bound remains (almost) intact. Since the argument for gap amplification and reduction is already covered in Dinur's original proof, we will primarily focus on how to analyze the increase and decrease in sensitivity.

We first note that the reduction in each step generally works as follows: Let $\mathcal{I}$ be a family of instances. Consider converting a CSP instance $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$ to another instance $I' = (V', E', \Sigma', \mathcal{R}')$. Suppose we have an algorithm A' for the latter. We then run A' on I' to obtain an assignment σ', and subsequently convert $\sigma' : V' \to \Sigma'$ back into an assignment $\sigma : V \to \Sigma$ for I. Let $\mathcal{I}'$ be the family of instances obtained from instances in $\mathcal{I}$ by converting them. The sensitivity of algorithms for $\mathcal{I}'$ is bounded from below by the sensitivity for $\mathcal{I}'$, multiplied by two parameters, C_I and C_σ, which are determined by the conversion of instances and assignments, respectively. Specifically, C_I represents the number of changes to the new instance I' when a constraint in the original instance I is changed, and C_σ represents the number of changes to the assignment σ for I when a value in the assignment σ' for I' is altered. To maintain a large sensitivity lower bound, C_I and C_σ need to be small.

The proof of Dinur's PCP construction proceeds by iteratively applying the following four steps; we now sketch how they can be modified in order to preserve sensitivity.

Degree Reduction. The goal of this step is to reduce the degree of each variable to a small constant and make the underlying graph regular. The reduction is straightforward: for each variable $v \in V$ with degree d in the original instance I, we split v into d copies, $v_1, \ldots, v_d$, and connect them with an expander, adding equality constraints on the newly introduced edges. To obtain a label for v in the original instance I, we select one of $v_1, \ldots, v_d$ uniformly at random and use the label of the chosen v_i in I'.

A key feature of this transformation is that it actually *increases* the sensitivity lower bound by d, which is crucial to offset the sensitivity decrease in other steps.

Expanderization. In this step, we simply superimpose an expander with a trivial constraint (satisfied by any assignment) onto the instance I to transform the underlying graph into an expander, which is important for the gap amplification step. We use the assignment for I' directly as the assignment for I.

Gap amplification. The goal of this step is to increase the gap between completeness and soundness by a constant factor t. Suppose the underlying graph is a d-regular expander. To achieve this, we sample a vertex $u \in V$, perform a random walk of length $O(t)$, and reach vertex v. We then add a constraint between u and v over the alphabet $\Sigma^{1+d+\cdots+d^t}$, which encodes the assignments of the t-hop neighborhoods, checking whether the labels of the variables along the path satisfy all the constraints. (This only defines a distribution over constraints; we convert this into an unweighted instance by multiplying the probability mass of each constraints by $d^{O(t)}$ to compute the number of copies needed for the constraint.)

To recover an assignment σ for I from an assignment σ' for I', for each variable $u \in V$, we gather the labels of variables in the t-hop neighborhood via a random walk. In Dinur's original proof [Din07], the majority of these labels would be used as the label for u in I. However, this recovery procedure does not allow us to preserve the sensitivity lower bound, as changing a single label could change the majority for many variables. To smooth this transition, we would like to choose a label from a distribution weighted according to the frequency that it occurs in this random walk. However, the possibility of returning a label with low probability mass ruins the soundness argument. Instead, we consider a distribution only over labels with sufficiently large probability mass. Combined with a careful conditioning argument, this allows us to preserve the sensitivity lower bound while still increasing the gap.

Alphabet reduction. This step decreases the size of the alphabet to a small constant. To do so, while preserving the gap, we take the Hadamard encoding of our alphabet. However, doing this converts our graph into a hypergraph, as every constraint now depends on many Boolean variables. To remedy this, we instead want to check whether a given assignment is *close* to a satisfying assignment to that constraint by examining only a few bits of that assignment. To do so, we apply a PCP to this constraint: we introduce sets of variables which purportedly encode the Hadamard table L of a satisfying assignment α for our constraint, as well as variables for the Hadamard table Q of $\alpha \otimes \alpha$. We then tests (via constraints) whether this is indeed the case, using applications of the BLR linearity test [BLR93].

To recover an assignment σ for I from an assignment σ' for I', for each variable $u \in V$, we consider the bits $\sigma'[u]$ which purportedly contain a label for u — a Hadamard codeword. We would like to map $\sigma'[u]$ to the closest Hadamard codeword (and hence label to u), however this would not allow us to preserve our bound on sensitivity: consider any assignments to $\sigma'[u]$ which lies on the unique decoding radius of a codeword. Then, changing a single bit in this assignment could change which codeword $\sigma'[u]$ is mapped to, and hence which label σ assigns to u. Instead, we smooth this transition by employing a randomized thresholding argument: We choose a random threshold $\tau \in [0, 1/4]$ (as one may recover uniquely a Hadamard codeword which has been $1/4$ corrupted) and assign $\sigma'[u]$ to its closest codeword if its relative distance to that codeword is at most τ, and to an arbitrary but fixed codeword otherwise. This smooth transition, along with a careful conditioning argument, allows us to maintain our bound on sensitivity.

1.2 Discussions

This work presents the first sensitivity lower bounds for approximation algorithms and establishes lower bounds for various graph problems. However, it also opens up several interesting directions for future research.

Parallel Repetition The *label cover problem* is a special type of a CSP where (i) every constraint is binary and has the so-called projection property (See Section 5 for details), and (ii) the underlying graph formed

by the constraints is bipartite. For $1 \geq c \geq s \geq 0$, let $\mathsf{LabelCover}_{c,s}$ denote the problem that, given a c-satisfiable label cover instance, the goal is to compute an s-satisfying assignment.

Parallel Repetition. [DS14,Raz95] is a powerful framework that reduces $\mathsf{LabelCover}_{1,1-\varepsilon}$ to $\mathsf{LabelCover}_{1,\varepsilon}$ without increasing the arity of constraints, though it does increase the alphabet size. The reduction itself is straightforward: for an integer parameter $t \geq 1$, for every choice of t constraints, which are over Σ, we add a new constraint over Σ^t that represents the conjunction of these t constraints. As a result, the number of constraints increases from m to m^t. This reduction plays a crucial role in deriving tight inapproximability results for problems like the set cover problem [Fei98, Mos12] and systems of linear equations [Hås01].

Unfortunately, it is not clear whether the parallel repetition framework can be used to derive sensitivity lower bounds for $\mathsf{LabelCover}_{1,\varepsilon}$ from those for $\mathsf{LabelCover}_{1,1-\varepsilon}$. The challenge arises because a modification to a label cover instance may lead to approximately $m^t-(m-1)^t \approx tm^{t-1}$ modifications in the instance produced by parallel repetition. Therefore, a naive lower bound for the latter problem would be $\frac{1}{tm^{t-1}}$ times that of the former problem, which becomes vacuous. An intriguing open question is whether recent techniques [BMV24] can be employed to establish meaningful sensitivity lower bounds for $\mathsf{LabelCover}_{1,\varepsilon}$.

Stronger Sensitivity Bounds. The reason that the lower bounds we obtain take the form $\Omega(n^\delta)$ is that, through the reductions, the instance size grows polynomially. It is known that there is a PCP of length $O(n \log^2 n)$ [BSGH+04, Din07], and an interesting question is whether we can use the PCP construction to obtain higher lower bounds. However, note that the lower bound cannot be of the form $n^{1-o(1)}$, because that would imply a lower bound of $n^{1-o(1)}$ for an $n^{-\varepsilon}$-approximation algorithm, which contradicts the fact that there is a trivial $n^{-\varepsilon}$-approximation algorithm with sensitivity $O(n^{1-\varepsilon})$.

Serial Repetition. By applying *serial repetition* to the label cover instance obtained from our PCP theorem — i.e., by taking the ANDs of subsets of constraints —- we are able to obtain polynomial sensitivity lower bounds against randomized algorithms which output a non-negligible approximation with high probability. When then use this lower bound to prove Theorems 1.1 and 1.4. When we are instead interested polynomial-time tractability, rather than low sensitivity, there is not much of a difference between such with-high-probability guarantees and and guarantees in expectation. Indeed, we can simply compute multiple assignments and take the best one that we find. However, this is not a sensitivity-preserving process. Indeed, it is not clear whether with-high-probability guarantees and in-expectation guarantees are equivalent for low-sensitivity algorithms, making it an interesting open problem to extend Theorems 1.1 and 1.4 to approximation algorithms with in-expectation guarantees.

Hardness for Polynomial-time Tractable Problems. Although our lower bound argument begins with $\mathsf{E2LIN}$, a polynomial-time tractable CSP, the CSP we obtain at the end of the PCP construction is NP-hard (as can be verified using Schaeffer's dichotomy theorem [Sch78]). Therefore, with the current approach, we can only establish lower bounds for NP-hard problems. An interesting open question is whether similar lower bounds can be established for polynomial-time tractable problems, such as finding a $(1-\varepsilon)$-satisfying assignment for satisfiable $\mathsf{E3LIN}$ instances, where each constraint is of the form $x+y+z=0 \pmod 2$ or $x+y+z=1 \pmod 2$.

Average Sensitivity. Average sensitivity [MY19, VY23] is a variation of sensitivity where we measure an algorithm's stability against average-case modifications to the instance. Specifically, the average sensitivity of an algorithm A on a graph $G=(V,E)$ is defined as the average Hamming distance between $A(G)$ and $A(G-e)$, where the average is over edges $e \in E$ deleted from G. We note that the sensitivity of an

algorithm provides an upper bound on average sensitivity, making the latter easier to bound. Algorithms with low average sensitivity have been proposed for various problems, including graph problems [VY23], dynamic programming problems [KY22b, KY22a], clustering problems [PY20, YI22], and learning problems [HY23].

An interesting question is whether the techniques developed in this work can be applied to bound average sensitivity. Our approach heavily relies on the fact that, in the context of CSPs, the sensitivity with respect to deleting a constraint can be lower bounded by half of the sensitivity with respect to swapping a constraint. However, this relationship does not generally hold for average sensitivity, and a more delicate argument is needed to account for changes in the underlying graph of a CSP instance.

1.3 Related Work

Differential privacy [Dwo06] is a fundamental concept in private data analysis, requiring that the output distributions of an algorithm be similar for neighboring instances. Though we do not define it formally here, it is straightforward to show that an ε-differentially private algorithm implies an algorithm with sensitivity $O(\varepsilon n)$, where n is a bound on the output size. By combining this with known differentially private algorithms, we can derive approximation algorithms with sensitivity $O(\varepsilon n)$ for the global minimum cut problem [GLM+10], the densest subgraph problem [DLR+22], and the correlation clustering problem [CAFL+22], although this sensitivity bound is relatively weak.

Aside from the linear sensitivity lower bound for 2-coloring, the only other known lower bound we are aware of is that any *deterministic* constant-factor approximation algorithm for the maximum matching problem requires $\Omega(\log^* n)$ queries [YZ21], where $\log^* n$ is the iterated logarithm of n. However, their argument relies on Ramsey theory, and thus it cannot be extended to randomized approximation algorithms.

Although DCSPs have been extensively studied in the AI community [FPY18, YH00], their theoretical aspects remain largely unexplored. One exception is the work by Butti and Dalmau [BD24], who provided a characterization of CSPs that can be solved by a deterministic distributed algorithm in finite time, assuming each agent lacks an identifier. We hope that our lower bound offers new insights into this problem.

Individual fairness [DHP+12] is another area which shares similarities with sensitivity. Individual fairness requires that similar individuals should be assigned similar labels (more specifically, their distributions over labels should have low statistical distance). Hence, our sensitivity bounds give lower bounds in the case when individuals are represented as graphs and are classified according to, for example, their cliques or cuts; such representations have been used, for example in [KHMT20].

2 Preliminaries

For positive integer n, let $[n]$ denote the set $\{1, 2, \ldots, n\}$. We use bold symbols to denote random variables. For a real number $x \in \mathbb{R}$, let $[x]_+ = \max\{x, 0\}$.

Graphs. We often use n and m to denote the number of vertices and edges in a graph when the graph is clear from context. For a graph $G = (V, E)$ and a vertex $v \in V$, let $\deg(v)$ denote the degree of v. For a set E and an element $e \in E$, let $E - e$ denote the set $E \setminus \{e\}$.

For a graph $G = (V, E)$, let $\lambda_i(G)$ denote the i-th largest eigenvalue of the adjacency matrix of G. It is known that $\lambda_1 = d$ when G is d-regular. Let $\lambda(G) = \max\{\lambda_2(G), |\lambda_n(G)|\}$. A d-regular graph $G = (V, E)$ is called an (n, d, λ)-*expander* if $\lambda(G) \leq \lambda < d$. The following lemmas guarantee the existence of sufficiently strong expanders.

Lemma 2.1 (see, e.g., [HLW06])**.** Let $d \geq 3$ be an integer. Then, there exist explicit constant $\lambda \leq d/2$ and an explicit (polynomial-time computable) family of (n, d, λ)-expanders.

Lemma 2.2 (see, e.g., [HLW06])**.** If $G = (V, E)$ is a d-regular graph and $G' = (V, E')$ is a d'-regular graph, then $H = (V, E \cup E')$ is a $(d + d')$-regular graph such that $\lambda(H) \leq \lambda(G) + \lambda(G')$.

Constraint satisfaction problems. We formally define *constraint satisfaction problems* (CSPs). An instance of a CSP is a tuple $I = (V, E, \Sigma, \mathcal{R} = \{R_e\}_{e \in E})$ consisting of a variable set V, a set of hyperedges E over V, a finite domain Σ (also referred to as the alphabet), and a relation $R_e \subseteq \Sigma^{|e|}$ for each $e \in E$. A *constraint* refers to a pair (e, R_e) for $e \in E$. We say that an assignment $\sigma : V \to \Sigma$ *satisfies* a constraint $(e = (v_1, \ldots, v_k), R_e)$ if $(\sigma(v_1), \ldots, \sigma(v_k)) \in R_e$. Also, we say that σ satisfies I if it satisfies all the constraints. The goal of a CSP is, given an instance I of the CSP, to find a satisfying assignment for I.

Now, we consider an optimization version of CSPs. For a CSP instance $I = (V, E, \Sigma, \mathcal{R})$ and an assignment $\sigma : V \to \Sigma$, let $\mathsf{val}_I(\sigma)$ denote the fraction of constraints in I satisfied by σ. We define $\mathsf{cost}_I(\sigma) = 1 - \mathsf{val}_I(\sigma)$ as the fraction of constraints violated by σ. Let $\mathsf{opt}(I) = \max_{\sigma : V \to \Sigma} \mathsf{val}_I(\sigma)$. For $1 \geq c \geq s \geq 0$, we define $\mathsf{MaxCSP}_{c,s}$ as the problem that, given an instance $I = (V, E, \Sigma, \mathcal{R})$ with $\mathsf{opt}(I) \geq c$, the goal is to find an assignment $\sigma : V \to \Sigma$ with $\mathsf{val}_I(\sigma) \geq s$.[1]

For two instances $I = (V, E, \Sigma, \{R_e\}_{e \in E})$ and $\tilde{I} = (V, E, \Sigma, \{\tilde{R}_e\}_{e \in E})$ on the same underlying hypergraph and domain, we define the *swap distance* between I and $\tilde{I}$ as $\mathsf{SwapDist}(I, \tilde{I}) := \#\{R_e \neq \tilde{R}_e : e \in E\}$.

Sensitivity. Let $I = (V, E, \Sigma, \{R_e\}_{e \in E})$ be a CSP instance. For a hyperedge $e \in E$, let $I - e$ denote the instance $(V, E - e, \Sigma, \{R_f\}_{f \in E - e})$. For two assignments $\sigma, \sigma' : V \to \Sigma$, let $\mathrm{Ham}(\sigma, \sigma') = \#\{v \in V : \sigma(v) \neq \sigma(v')\}$ denote their Hamming distance. The *sensitivity* of a deterministic algorithm A on I is defined as

$$\mathsf{Sens}(A, I) := \max_{e \in E} \mathrm{Ham}(A(I), A(I - e)), \tag{1}$$

where $A(I)$ denotes the output assignment of A on I.

For a distribution μ over X and another distribution $\tilde{\mu}$ over $\tilde{X}$, we say that a joint distribution π over $X \times \tilde{X}$ is a *coupling* between them if the marginal distributions on the first and the second coordinates are equal to μ and $\tilde{\mu}$, respectively. Let $\Pi(\mu, \tilde{\mu})$ denote the set of all couplings between μ and $\tilde{\mu}$. For two distributions $\mu, \tilde{\mu}$ over assignments on the same domain, we define the earth mover's distance between them as

$$\mathsf{EMD}(\mu, \tilde{\mu}) = \min_{\pi \in \Pi(\mu, \tilde{\mu})} \mathop{\mathbf{E}}_{(\sigma, \tilde{\sigma}) \sim \pi} \mathrm{Ham}(\sigma, \tilde{\sigma}).$$

The *sensitivity* of a randomized algorithm A on a CSP instance $I = (V, E, \Sigma, \mathcal{R})$ is defined as

$$\mathsf{Sens}(A, I) := \max_{e \in E} \mathsf{EMD}(A(I), A(I - e)),$$

where we identify random variables $A(I)$ and $A(I - e)$ with their distributions. Note that this definition matches (1) when the algorithm is deterministic. For a family of algorithms $\mathcal{A}$ and a family of instances $\mathcal{I}$ over the same domain Σ, we define

$$\mathsf{Sens}(\mathcal{A}, \mathcal{I}) := \min_{A \in \mathcal{A}} \max_{I \in \mathcal{I}} \mathsf{Sens}(A, I).$$

[1] Usually, $\mathsf{MaxCSP}_{c,s}$ refers to the decision problem where the goal is to distinguish instances I with $\mathsf{opt}(I) \geq c$ from instances I with $\mathsf{opt}(I) \leq s$. However, we define it as a search problem because we are focused on the sensitivity of algorithms.

Also, we define $\mathsf{Sens}_{c,s}(\mathcal{I}) := \mathsf{Sens}(\mathcal{A}, \mathcal{I})$, where $\mathcal{A}$ is the family of all possible algorithms for $\mathsf{MaxCSP}_{c,s}$.

We define a variant of sensitivity which is more convenient when studying CSPs. Let $I = (V, E, \Sigma, \{R_e\}_{e \in E})$ be a CSP instance. For $e \in E$ and $R \subseteq \Sigma^{|e|}$, $I^{e \leftarrow R}$ denote the instance obtained from I by replacing R_e with R. Then, the *swap sensitivity* of a randomized algorithm A on an instance $I = (V, E, \Sigma, \mathcal{R})$ is defined as

$$\mathsf{SwapSens}(A, I) := \max_{e \in E} \max_{R \subseteq \Sigma^{|e|}} \mathsf{EMD}\big(A(I), A(I^{e \leftarrow R})\big).$$

We define $\mathsf{SwapSens}(\mathcal{A}, \mathcal{I})$ and $\mathsf{SwapSens}_{c,s}(\mathcal{I})$ as with sensitivity. We say that a family of instances $\mathcal{I}$ is *swap-closed* if for any $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$, $e \in E$, and $R \subseteq \Sigma^{|e|}$, the instance $I^{e \leftarrow R}$ also belongs to $\mathcal{I}$. Note that for any $A \in \mathcal{A}$ and $I, \tilde{I} \in \mathcal{I}$ for a swap-closed family of instances $\mathcal{I}$, we have $\mathsf{EMD}(A(I), A(\tilde{I})) \leq \mathsf{SwapSens}(\mathcal{A}, \mathcal{I}) \cdot \mathsf{SwapDist}(I, \tilde{I})$. For a family of instances $\mathcal{I}$, we define $\mathsf{SwapClo}(\mathcal{I})$ as the *swap-closure* of $\mathcal{I}$, i.e., the family of instances obtained by (repeatedly) swapping constraints in $I \in \mathcal{I}$.

The following lemma shows that we can reduce the problem of bounding sensitivity to that of bounding swap sensitivity.

Lemma 2.3. For any family of algorithms $\mathcal{A}$ and family of CSP instances $\mathcal{I}$, we have

$$\mathsf{Sens}(\mathcal{A}, \mathcal{I}) \geq \frac{1}{2}\mathsf{SwapSens}(\mathcal{A}, \mathcal{I}).$$

Proof. Let $A \in \mathcal{A}$ be an algorithm that attains $\mathsf{Sens}(\mathcal{A}, \mathcal{I})$. For any $I = (V, E, \Sigma, \{R_e\}_{e \in E}) \in \mathcal{I}$, $e \in E$, and $R \subseteq \Sigma^{|e|}$, we have

$$\mathsf{EMD}(A(I), A(I^{e \leftarrow R})) \leq \mathsf{EMD}(A(I), A(I - e)) + \mathsf{EMD}(A(I - e), A(I^{e \leftarrow R})) \leq 2\mathsf{Sens}(\mathcal{A}, \mathcal{I}).$$

Hence, we have

$$\mathsf{SwapSens}(\mathcal{A}, \mathcal{I}) \leq \max_{I \in \mathcal{I}} \mathsf{SwapSens}(A, I) \leq \max_{I \in \mathcal{I}, e \in E, R \subseteq \Sigma^{|e|}} \mathsf{EMD}(A(I), A(I^{e \leftarrow R})) \leq 2\mathsf{Sens}(\mathcal{A}, \mathcal{I}). \quad \square$$

3 Sensitivity-Preserving Reductions

The PCP framework can be viewed as a sequence of reductions between CSPs. Therefore, it is useful to introduce a template for the reductions that we will use throughout our analysis.

Definition 3.1. Let $\mathcal{I}$ be a family of CSP instances on the same underlying hypergraph and domain. Let T_I be a procedure that transforms a CSP instance $I \in \mathcal{I}$ to another CSP instance I', and let T_σ be a (possibly randomized) procedure that transforms an assignment $\sigma' : V' \to \Sigma'$ for I' to an assignment $\boldsymbol{\sigma} : V \to \Sigma$ for I (T_σ might depend on I). For $c, s, c', s' \in [0, 1]$ and $C_I, C_\sigma > 0$, we say that the pair (T_I, T_σ) is a $(c, s, c', s', C_I, C_\sigma)$-sensitivity-preserving reduction for $\mathcal{I}$ if the following holds:

1. If $\mathsf{opt}(I) \geq c$, then $\mathsf{opt}(I') \geq c'$.
2. If a (possibly random) assignment $\boldsymbol{\sigma}'$ for I' satisfies $\mathbf{E}[\mathsf{val}_{I'}(\boldsymbol{\sigma}')] \geq s'$, then the assignment $\boldsymbol{\sigma} = T_\sigma(\boldsymbol{\sigma}')$ satisfies $\mathbf{E}[\mathsf{val}_I(\boldsymbol{\sigma})] \geq s$.
3. Let $I, \tilde{I} \in \mathcal{I}$ be two CSP instances. Then, we have $\mathsf{SwapDist}(T_I(I), T_I(\tilde{I})) \leq C_I \cdot \mathsf{SwapDist}(I, \tilde{I})$. In particular, this implies that T_I generates CSP instances on the same underlying hypergraph and domain for any $I \in \mathcal{I}$.
4. Let $I, \tilde{I} \in \mathcal{I}$ be two CSP instances and let $\boldsymbol{\sigma}', \tilde{\boldsymbol{\sigma}}'$ be assignments for $T_I(I)$ and $T_I(\tilde{I})$, respectively. Then, we have $\mathsf{EMD}(T_\sigma(\boldsymbol{\sigma}'), T_\sigma(\tilde{\boldsymbol{\sigma}}')) \leq C_\sigma \cdot \mathsf{EMD}(\boldsymbol{\sigma}', \tilde{\boldsymbol{\sigma}}')$.

Lemma 3.2. Let $\mathcal{I}$ be a swap-closed family of CSP instances on the same underlying hypergraph and the same domain. Suppose that there exists a $(c, s, c', s', C_I, C_\sigma)$-sensitivity-preserving reduction (T_I, T_σ) for $\mathcal{I}$. Then we have

$$\mathsf{SwapSens}_{c',s'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \frac{1}{C_I C_\sigma}\mathsf{SwapSens}_{c,s}(\mathcal{I}),$$

where $\mathcal{I}' = T_I(\mathcal{I}) := \{T_I(I) : I \in \mathcal{I}\}$.

Proof. Let A' be an algorithm that attains $\mathsf{SwapSens}_{c',s'}(\mathsf{SwapClo}(\mathcal{I}'))$. Then, we design an algorithm A for $\mathcal{I}$ using A' as follows: Given an instance $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$, we construct an instance $I' = (V', E', \Sigma', \mathcal{R}') = T_I(I)$. Then, we run the algorithm A' on I' to obtain a (possibly random) assignment $\boldsymbol{\sigma}' : V' \to \Sigma'$ for I'. Then, we output an assignment $\boldsymbol{\sigma} = T_\sigma(\boldsymbol{\sigma}')$ for $\mathcal{I}$.

We analyze the approximation guarantee of A. Suppose $\mathsf{opt}_I(I) \geq c$. Then by Item 1 of Definition 3.1, we have $\mathsf{opt}(I') \geq c'$. Hence, the output assignment $\boldsymbol{\sigma}'$ satisfies $\mathbf{E}\,\mathsf{val}(I', \boldsymbol{\sigma}') \geq s'$. By Item 2 of Definition 3.1, we have that $\mathbf{E}\,\mathsf{val}(I, \boldsymbol{\sigma}) \geq s$.

Now, we analyze the swap sensitivity of A. Let $I, \tilde{I} \in \mathcal{I}$ be two CSP instances with swap distance one. Then by Item 3 of Definition 3.1, we have $\mathsf{SwapDist}(I', \tilde{I}') \leq C_I \cdot \mathsf{SwapDist}(I, \tilde{I})$, which implies that

$$\mathsf{EMD}(\boldsymbol{\sigma}', \tilde{\boldsymbol{\sigma}}') \leq C_I \cdot \mathsf{SwapSens}(A', \mathsf{SwapClo}(\mathcal{I}')) = C_I \cdot \mathsf{SwapSens}_{c',s'}(\mathsf{SwapClo}(\mathcal{I}')).$$

By Item 4 of Definition 3.1, we have $\mathsf{EMD}(T_\sigma(\boldsymbol{\sigma}'), T_\sigma(\tilde{\boldsymbol{\sigma}}')) \leq C_I C_\sigma \cdot \mathsf{SwapSens}_{c',s'}(\mathsf{SwapClo}(\mathcal{I}'))$. Hence, we must have $\mathsf{SwapSens}_{c',s'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \mathsf{SwapSens}_{c,s}(\mathcal{I})/(C_I C_\sigma)$. □

4 Lower Bounds for E2LIN

Let $R_0, R_1 \in \mathbb{Z}_2^2$ be binary relations over $\mathbb{Z}_2$ such that $(a, b) \in R_0$ if and only if $a + b = 0 \pmod 2$ and $(a, b) \in R_1$ if and only $a + b = 1 \pmod 2$, respectively. Then, we define E2LIN as the CSP over $\mathbb{Z}_2$ where only R_0 and R_1 are used to define the constraints. Also, let $\mathsf{E2LIN}_{c,s}$ be the special case of $\mathsf{MaxCSP}_{c,s}$, where the instances are restricted to those of E2LIN. In this section, we establish a sensitivity lower bound for $\mathsf{E2LIN}_{1,1-1/2n}$, which we will later amplify using the PCP framework.

Let $\mathcal{I}_n$ denote the set of all E2LIN instances $(V, E, \mathbb{Z}_2, \mathcal{R})$ such that the graph (V, E) forms a cycle on n vertices. Note that $\mathcal{I}_n$ is swap-closed. Let $\mathcal{A}$ denote the set of randomized algorithms such that the expected number of errors on satisfiable instances in $\mathcal{I}_n$ is at most half. We show the following lower bound.

Lemma 4.1. For any even integer $n \geq 4$, we have $\mathsf{SwapSens}(\mathcal{A}, \mathcal{I}_n) = \Omega(n)$. In particular, we have $\mathsf{SwapSens}_{1,1-1/2n}(\mathcal{I}_n) = \Omega(n)$.

Proof. Let $A \in \mathcal{A}$ be the algorithm that attains $\mathsf{SwapSens}(\mathcal{A}, \mathcal{I}_n)$. Consider an instance $I = (V = (v_1, \ldots, v_n), E, \mathbb{Z}_2, \{R_e\}_{e \in E}) \in \mathcal{I}_n$ such that $R_e = R_1$ for every $e \in E$. Note that I is satisfiable and any assignment violates an even number of constraints in I, and hence A must output a satisfying assignment on I with probability at least $3/4$ (otherwise, the expected number of errors is more than $1/4 \cdot 2 = 1/2$). Note that there are only two satisfying assignments $\sigma_1, \sigma_2 : V \to \mathbb{Z}_2$ for I. Without loss of generality, we can assume that they are of the following form:

$$\sigma_1(v_i) = \begin{cases} 0 & i \text{ is odd,} \\ 1 & i \text{ is even,} \end{cases} \qquad \sigma_2(v_i) = \begin{cases} 1 & i \text{ is odd,} \\ 0 & i \text{ is even.} \end{cases}$$

Let $e = (v_{n/2}, v_{n/2} + 1)$ and $e' = (v_n, v_1)$ and consider an instance $\tilde{I} = I^{e \leftarrow R_0, e' \leftarrow R_0}$. As $\tilde{I}$ is also satisfiable, A must output a satisfying assignment on I' with probability at least $3/4$. Note that there are

only two satisfying assignments $\tilde{\sigma}_1, \tilde{\sigma}_2 : V \to \{0,1\}$ for $\tilde{I}$. Without loss of generality, we can assume that they are of the following form:

$$\tilde{\sigma}_1(v_i) = \begin{cases} 0 & i \text{ is odd and } i \le n/2, \\ 1 & i \text{ is even and } i \le n/2, \\ 1 & i \text{ is odd and } i \ge n/2+1, \\ 0 & i \text{ is even and } i \ge n/2+1, \end{cases} \qquad \tilde{\sigma}_2(v_i) = \begin{cases} 1 & i \text{ is odd and } i \le n/2, \\ 0 & i \text{ is even and } i \le n/2, \\ 0 & i \text{ is odd and } i \ge n/2+1, \\ 1 & i \text{ is even and } i \ge n/2+1. \end{cases}$$

For every $i, j \in \{1,2\}$, we have $\mathsf{Ham}(\sigma_i, \tilde{\sigma}_j) \ge n/2$. Hence, we have

$$\mathsf{SwapSens}(\mathcal{A}, \mathcal{I}_n) \ge \frac{\mathsf{EMD}(A(I), A(\tilde{I}))}{\mathsf{SwapDist}(I, \tilde{I})} = \frac{1}{2} \cdot \left(1 - \frac{1}{4} - \frac{1}{4}\right) \cdot \frac{n}{2} = \Omega(n). \qquad \square$$

5 PCPs and Sensitivity

In this section, we show a sensitivity lower bound for a problem called LabelCover, which is a special case of CSPs.

Definition 5.1. An instance of LabelCover is a tuple $I = (U, V, E, \Sigma_U, \Sigma_V, \mathcal{R} = \{R_e\}_{e \in E})$, where $(U \cup V, E)$ forms a bipartite graph, Σ_U, Σ_V are finite domains, and each relation $R_e \subseteq \Sigma_L \times \Sigma_R$ is a projection. Here, a *projection* refers to a relation of the form $R_e = \{(a, \phi_e(a)) : a \in \Sigma_U\}$ for some map $\phi_e : \Sigma_U \to \Sigma_V$. For $1 \ge c \ge s \ge 0$, we define $\mathsf{LabelCover}_{c,s}$ as the problem that, given a label cover instance I with $\mathsf{opt}(I) \ge c$, the goal is to find an assignment σ for I with $\mathsf{val}_I(\sigma) \ge s$.

The goal of this section is to show the following:

Theorem 5.2. There exist universal constants $\varepsilon, \delta > 0$ and $d, k \ge 1$ such that any algorithm for $\mathsf{LabelCover}_{1,1-\varepsilon}$ on a bipartite graph of maximum degree d and a domain of size k has sensitivity $\Omega(n^\delta)$.

As we discussed in Section 1.1, the proof of Theorem 5.2 consists of four steps: Degree reduction, expanderization, gap amplification, and alphabet reduction. We discuss these four steps in Sections 5.1 to 5.4, respectively. Finally, we prove Theorem 5.2 in Section 5.5.

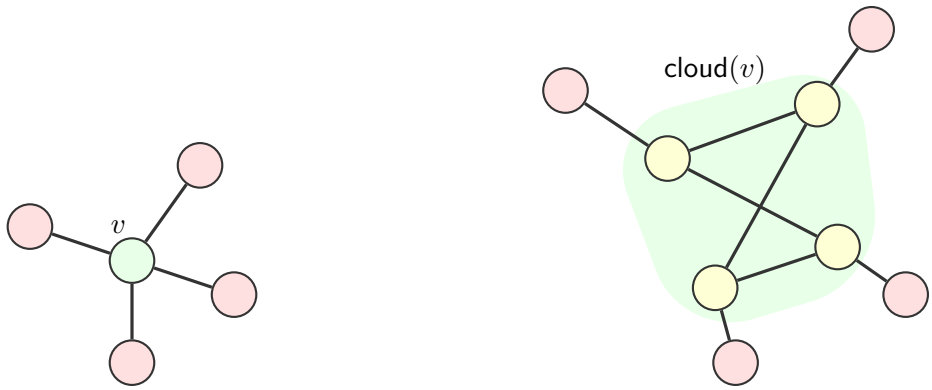
5.1 Degree Reduction

In this section, we introduce a transformation that reduces degrees of vertices in the underlying graph of a CSP instance. Recall that, by Lemma 2.1, there exist universal constants $\lambda_0 < d_0$ such that (n, d_0, λ_0)-expanders can be explicitly constructed in polynomial time. Let $I = (V, E, \Sigma, \mathcal{R})$ be a d-regular CSP instance with $m = |E|$. We consider the following procedure, called DEGREEREDUCTION, to construct an instance $I' = (V', E', \Sigma', \mathcal{R}')$:

- Replace each vertex $v \in V$ by d many vertices to get the new vertex set V'. Denote the set of new vertices corresponding to v by $\mathrm{cloud}(v)$. Each vertex in $\mathrm{cloud}(v)$ naturally corresponds with a neighbor of v from $G = (V, E)$.
- For each edge $e \in E$, place an "inter-cloud" edge e' in E' between the associated cloud vertices. This gives exactly one inter-cloud edge per vertex in V'. Whatever the old constraint R_e on e was, put the exact same constraint on e'.

– For each $v \in V$, put a (d, d_0, λ_0)-expander on $\mathrm{cloud}(v)$ given by Lemma 2.1. Further, put equality constraints on these expander edges.

We can observe that in this process each new vertex in V' has degree exactly equal to $d_0 + 1$. Thus we have created a $(d_0 + 1)$-regular graph, as desired. Also the number of newly added edges is equal to $\sum_{u \in V} dd_0/2 = d_0 \sum_{u \in V} d/2 = d_0 m$, and hence $m' = (d_0 + 1)m$. Note that the domain Σ does not change and in particular $\Sigma' = \Sigma$. A depiction is given in Figure 1.



(a) Vertex v prior to DEGREEREDUCTION. **(b)** cloud(v) after DEGREEREDUCTION.

Figure 1: DEGREEREDUCTION on a vertex v with $d = 4$. The intra-cloud edges represent an expander with $d_0 = 2$.

We now show how the sensitivity bound translates through the transformation.

Lemma 5.3. Let $\mathcal{I}$ be a swap-closed family of binary d-regular CSP instances over Σ and let $\mathcal{I}' = \textsc{DegreeReduction}(\mathcal{I})$. Then for any $\varepsilon > 0$, we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) \geq d \cdot \mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I}),$$

for $\varepsilon' := \varepsilon/C$, where $C > 0$ is a universal constant.

We note that this process indeed *increases* the sensitivity lower bound, which is crucial for offsetting the sensitivity decrease in the gap amplification and alphabet reduction steps.

Proof of Lemma 5.3. Let $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$ and $I' = (V', E', \Sigma, \mathcal{R}') = \textsc{DegreeReduction}(\mathcal{I})$. We consider an algorithm T_σ that, given an assignment $\sigma' : V' \to \Sigma$ for I', constructs an assignment $\boldsymbol{\sigma} : V \to \Sigma$ for I by setting $\boldsymbol{\sigma}(u) = \sigma'(\boldsymbol{u}')$, where $\boldsymbol{u}' \in V'$ is a vertex sampled from $\mathrm{cloud}(u)$ uniformly at random.

We show that the pair $(\textsc{DegreeReduction}, T_\sigma)$ is a $(1, 1-\varepsilon, c' = 1, s' = 1-\varepsilon', C_I = 1, C_\sigma = 1/d)$-sensitivity-preserving reduction, from which the claim follows. As the analysis for $c' = 1$ and $C_I = 1$ is obvious, we analyze s' and C_σ below.

Let $\boldsymbol{\sigma}'$ be an assignment for I' with $\mathbf{E}\,\mathsf{val}_{I'}(\boldsymbol{\sigma}') \geq 1-\varepsilon' = s'$. Our goal is to show that $\mathbf{E}\,\mathsf{val}_I(\boldsymbol{\sigma}) \geq 1-\varepsilon$. We condition on $\boldsymbol{\sigma}' = \sigma'$ and aim to show that $\mathbf{E}[\mathsf{cost}_I(\boldsymbol{\sigma})] \leq C \cdot \mathbf{E}[\mathsf{cost}_{I'}(\sigma')]$. We then obtain the result by unconditioning $\boldsymbol{\sigma}'$.

For a vertex $u \in V$, let us define $\boldsymbol{S}^u$ to be the set of vertices in $\mathrm{cloud}(u)$ on which σ' disagrees with $\boldsymbol{\sigma}(u)$. Suppose $e = (u, v) \in E$ is one of the edges in I that are violated by $\boldsymbol{\sigma}$. Let e' be the corresponding inter-cloud edge in E'. The key observation is that either σ' violates the edge e' or one of the endpoints of

e' belongs to $\boldsymbol{S}^u$ or $\boldsymbol{S}^v$. Thus we have

$$\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m \le \mathbf{E}\left[\mathsf{cost}_{I'}(\sigma')m' + \sum_{u\in V} |\boldsymbol{S}^u|\right].$$

It follows that either

a) $\mathbf{E}\,\mathsf{cost}_{I'}(\sigma') \cdot m' \ge \mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m/2$, or
b) $\sum_{u\in V} \mathbf{E}\,|\boldsymbol{S}^u| \ge \mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m/2$.

In case a), we obtain

$$\mathsf{cost}_{I'}(\sigma') \cdot m' \ge \frac{\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m}{2} = \frac{\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m'}{2(d_0+1)}.$$

Hence, we have $\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \le C \cdot \mathbf{E}\,\mathsf{cost}_{I'}(\sigma')$ by setting $C \ge 2(d_0+1)$.

To handle case b), for each label $a \in \Sigma$, let $C_a^u = (\sigma')^{-1}(a) \cap \mathrm{cloud}(u)$ be the set of vertices in $\mathrm{cloud}(u)$ that are labelled a by σ' and $p_a^u = \frac{|C_a^u|}{|\mathrm{cloud}(u)|}$ be its fraction. Then, note that

$$\mathbf{E}\,|\boldsymbol{S}^u| = \sum_{a\in\Sigma} p_a^u |\mathrm{cloud}(u) \setminus C_a^u| = |\mathrm{cloud}(u)| \cdot \sum_{a\in\Sigma} p_a^u(1-p_a^u),$$

where the first equality follows as p_a^u is the probability that we choose label a as our label for $\boldsymbol{\sigma}$, and hence every vertex in $\mathrm{cloud}(u) \setminus C_a^u$ differs from the label given to $\boldsymbol{\sigma}$. Note that every edge between C_a^u and C_b^u for $a \ne b$ is violated by σ' because they are all labelled with "equality" constraints. Then by the fact that the cloud is an expander, there exists a constant $\phi > 0$ that is determined by d_0 and λ_0 such that the number of edges in $\mathrm{cloud}(u)$ violated by σ' is at least

$$\begin{aligned}
&\frac{\phi}{2}\sum_{a\in\Sigma} \min\{|C_a^u|, |\mathrm{cloud}(u) \setminus C_a^u|\} = \frac{\phi|\mathrm{cloud}(u)|}{2} \cdot \sum_{a\in\Sigma} \min\{p_a^u, 1-p_a^u\} \\
&\ge \frac{\phi|\mathrm{cloud}(u)|}{2} \cdot \sum_{a\in\Sigma} p_a^u(1-p_a^u) = \frac{\phi}{2}\,\mathbf{E}\,|\boldsymbol{S}^u|.
\end{aligned}$$

Therefore σ' violates at least the following number of edges:

$$\begin{aligned}
\mathsf{cost}_{I'}(\sigma') \cdot m' &\ge \frac{\phi}{2}\sum_{u\in V} \mathbf{E}\,|\boldsymbol{S}^u| \\
&\ge \frac{\phi\,\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m}{4} && \text{(since we are in case (b))} \\
&= \frac{\phi\,\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m'}{4(d_0+1)}.
\end{aligned}$$

Hence, we have $\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \le C \cdot \mathsf{cost}_{I'}(\sigma')$ by setting $C \ge 4(d_0+1)/\phi$.

Next, we analyze the value of C_σ. Let $I, \tilde{I} \in \mathcal{I}$ be CSP instances and let $I' = \textsc{DegreeReduction}(I)$ and $\tilde{I}' = \textsc{DegreeReduction}(\tilde{I})$. Let $\sigma', \tilde{\sigma}' : V' \to \Sigma$ be assignments for $\mathcal{I}', \tilde{\mathcal{I}}'$, respectively. Let $\pi \in \Pi(\boldsymbol{\sigma}', \tilde{\boldsymbol{\sigma}}')$ such that $\mathrm{EMD}(\boldsymbol{\sigma}', \tilde{\boldsymbol{\sigma}}') = \mathbf{E}_{(\sigma',\tilde{\sigma}')\sim\pi}\,\mathrm{Ham}(\sigma', \tilde{\sigma}')$. Then, we have

$$\mathsf{EMD}(\boldsymbol{\sigma}, \tilde{\boldsymbol{\sigma}}) \le \mathop{\mathbf{E}}_{(\sigma',\tilde{\sigma}')\sim\pi} \sum_{u\in V} \mathsf{TV}(\boldsymbol{\sigma}(u) \mid \sigma', \tilde{\boldsymbol{\sigma}}(u) \mid \tilde{\sigma}')$$

$$
\begin{aligned}
&= \mathop{\mathbf{E}}_{(\sigma',\tilde{\sigma}')\sim\pi} \sum_{u\in V} \frac{1}{2} \sum_{a\in\Sigma} \left| \frac{|(\sigma')^{-1}(a)\cap \mathrm{cloud}(u)|}{|\mathrm{cloud}(u)|} - \frac{|(\tilde{\sigma}')^{-1}(a)\cap \mathrm{cloud}(u)|}{|\mathrm{cloud}(u)|} \right| \\
&= \frac{1}{d} \mathop{\mathbf{E}}_{(\sigma',\tilde{\sigma}')\sim\pi} \sum_{u\in V'} \mathbf{1}[\sigma'(u) \neq \tilde{\sigma}'(u)] \\
&= \frac{\mathsf{EMD}(\boldsymbol{\sigma}', \tilde{\boldsymbol{\sigma}}')}{d}.
\end{aligned}
$$

□

Hence, the choice $C_\sigma = 1/d$ satisfies Item 4 of Definition 3.1.

We will need a variant of Lemma 5.3 in the alphabet reduction step, which we discuss below. First, we note that DEGREEREDUCTION can be extended to non-regular instances by introducing $\deg(v)$ copies of each vertex $v \in V$. Note that the resulting graph is (d_0+1)-regular.

Marked CSPs. A *marked CSP instance* $\hat{I} = (V, E, \Sigma, \mathcal{R}, S)$ consists of a CSP instance $(V, E, \Sigma, \mathcal{R})$ and a set of *marked vertices* $S \subseteq V$.

For a marked CSP instance $\hat{I}$, we measure the sensitivity of an algorithm A using marked vertices only. Specifically, we define

$$\mathsf{Sens}(A, \hat{I}) = \max_{e\in E} \mathsf{Ham}(A(I)\mid_S, A(I-e)\mid_S),$$

where for an assignment $\sigma : V \to \Sigma$, $\sigma|_S : S \to \Sigma$ is a restriction of σ on S. Then, we can define other sensitivity-related notions for marked CSP instances using the definition above.

For a marked instance $\hat{I}$, we define DEGREEREDUCTION$(\hat{I})$ as a marked instance $(V', E', \Sigma, \mathcal{R}', S')$, where $(V', E', \Sigma, \mathcal{R}') = \text{DEGREEREDUCTION}(I)$ and $S' = \bigcup_{v\in S} \mathrm{cloud}(v)$. We can define the swap-closed property and the swap-closure for a family of marked instances naturally. The proof of Lemma 5.3 gives the following:

Corollary 5.4. Let $\hat{\mathcal{I}}$ be a swap-closed family of binary marked CSP instances over Σ, where every marked vertex has degree at least d, and let $\hat{\mathcal{I}}' = \text{DEGREEREDUCTION}(\hat{\mathcal{I}})$. Then for any $\varepsilon > 0$, we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\hat{\mathcal{I}}')) \geq d \cdot \mathsf{SwapSens}_{1,1-\varepsilon}(\hat{\mathcal{I}}),$$

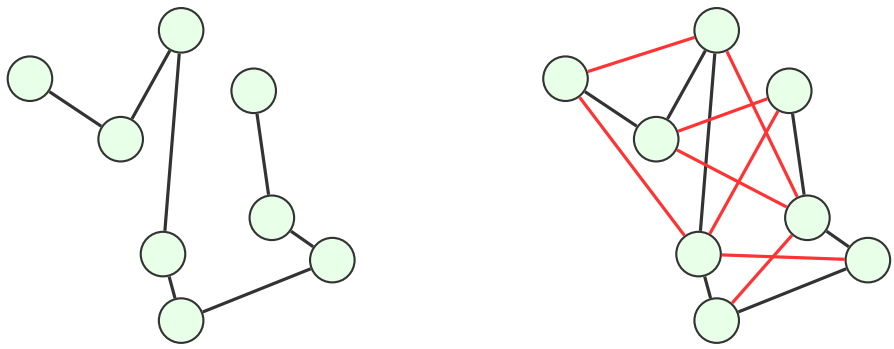
for $\varepsilon' := \varepsilon/C$, where $C > 0$ is a universal constant.

5.2 Expanderization

In this section, we introduce a transformation that makes the underlying graph of a CSP instance into an expander. This subroutine, called EXPANDERIZATION, is simple. Given a binary d-regular instance I on n variables, we just superimpose an (n, d_0, λ_0)-expander given by Lemma 2.1. (This may lead to multiple edges.) On each edge of the expander we simply put a trivial constraint, i.e., a constraint that is always satisfied. A depiction can be seen in Figure 2.

Let us now record some parameters of G'. The new graph is regular with degree $d + d_0$ and the new number of edges is $n(d+d_0)/2$. Also, the new constraint graph is indeed a constant degree expander because the new λ' is at most $d + \lambda_0 < d + d_0$ by Lemma 2.2. Note that domain Σ does not change and so $\Sigma' = \Sigma$.

We now show how the sensitivity bound translates through the transformation.



(a) G before EXPANDERIZATION **(b)** G after EXPANDERIZATION

Figure 2: A graph G before and after EXPANDERIZATION. The edges of the exapnder are marked in red.

Lemma 5.5. Let $\mathcal{I}$ be a swap-closed family of binary d-regular instances of a CSP over Σ, and let $\mathcal{I}' = \text{EXPANDERIZE}(\mathcal{I})$. Then for any $\varepsilon > 0$, we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})$$

for $\varepsilon' := \varepsilon/C$, where $C > 0$ is a universal constant.

Proof. Let $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$ and $I' = (V, E', \Sigma, \mathcal{R}') = \text{EXPANDERIZE}(\mathcal{I})$. We consider a trivial algorithm T_σ that, given an assignment $\boldsymbol{\sigma}'$ for I', outputs $\boldsymbol{\sigma} = \boldsymbol{\sigma}'$ as an assignment for I.

We show that the pair $(\text{EXPANDERIZE}, T_\sigma)$ is a $(1, 1-\varepsilon, c' = 1, s' = 1-\varepsilon', C_I = 1, C_\sigma = 1)$-sensitivity-preserving reduction. Then the claim follows by Lemma 3.2. As the analysis for $c' = 1$, $C_I = 1$, and $C_\sigma = 1$ is trivial, we analyze s' below. Let $\boldsymbol{\sigma}'$ be an assignment for I' with $\mathbf{E}\,\mathsf{val}_{I'}(\boldsymbol{\sigma}') \geq 1 - \varepsilon' = s'$. Then we have

$$\varepsilon' m' \geq \mathbf{E}\,\mathsf{cost}_{I'}(\boldsymbol{\sigma}) \cdot m' = \mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot m = \mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \cdot \frac{dm'}{d + d_0}.$$

Hence, we get $\mathbf{E}\,\mathsf{cost}_I(\boldsymbol{\sigma}) \leq \varepsilon m$ by setting $C \geq d/(d + d_0)$. □

5.3 Gap Amplification

In this section, we consider amplifying the gap between the completeness and soundness. Here, we use the transformation based on graph powering, introduced by Dinur [Din07].

5.3.1 Graph Powering

Let t be an integer and $I := (V, E, \Sigma, \mathcal{R} = \{R_e\}_{e \in E})$ be a CSP instance, where the graph $G = (V, E)$ is an (n, d, λ)-expander. We construct a new instance $I' := (V, E', \Sigma, \mathcal{R}' = \{R'_e\}_{e \in E})$, where the constraint R'_e for $e \in E$ is a subset of $\Sigma^{1+d+\cdots+d^t} \times \Sigma^{1+d+\cdots+d^t}$. As G is d-regular, for any vertex $v \in V$, the number of vertices at distance at most t from v is at most $1 + d + \cdots + d^t$. In any assignment $\sigma' : V \to \Sigma^{1+d+\cdots+d^t}$ for I', we will think of each vertex $v \in V$ as having an "opinion" about what the value of each vertex $w \in V$ at distance $\leq t$ should be; which is $\sigma'(v)_w$. Roughly, the constraints will state that if the edge (a, b) is of distance $\leq t$ from u and v then the assignment $(\sigma'(v)_a, \sigma'(w)_b)$ satisfies the constraint R_{ab}. As each vertex now is required to answer correctly on all constraints within radius t, this should blow up the gap by $O(t)$.

It will be convenient to define the edges E' and the constraints $\{R'_e\}_{e \in E'}$ by the following process:

1. Describe a distribution over edges $e \in E'$ and corresponding constraints R'_e.
2. View this distribution as a weighted graph, where edges have rational weights w_e depending on d, $|\Sigma|$, and t.
3. Introduce copies of the constraint R_e, where the number of copies is proportional to w_e.

In particular, the distribution will be useful for analyzing the change in the gap and the sensitivity. In order to describe this distribution, we will make use of following two types of random walks.

- *Before Stopping Random Walk (BSRW).* Pick a random vertex $\boldsymbol{v} \in V$ to be the start vertex. Repeat the following: Choose a random neighbour of the current vertex, and move to that vertex, and halt with probability $1/t$.
- *After Stopping Random Walk (ASRW).* Let v be a given start vertex. Repeat the following: With probability $1/t$ halt. If we did not halt, pick a random neighbour of the current vertex, and move to that vertex.

We now describe the gap amplification step, which is by a reduction known as POWERING. Given an instance $I = (V, E, \Sigma, \mathcal{R} = \{R_e\}_{e \in E})$, the set E' and the constraints $\{R'_e\}_{e \in E'}$ of the instance I' that we are constructing are defined by the following distribution:

- E': Pick a random vertex $\boldsymbol{v}$. Perform an ASRW from $\boldsymbol{v}$, ending at some vertex $\boldsymbol{w}$. If the length of this walk is greater than $B := 10t \log |\Sigma|$, then do nothing. Otherwise, add an edge $(\boldsymbol{v}, \boldsymbol{w})$ to E'.
- R'_e: Let $e = (v, w)$ be the edge defined in the previous step. We describe the constraint R'_e: for each edge (a, b) traversed in the ASRW used to define e, if $d_G(v, a) \leq t$ and $d_G(w, b) \leq t$, then add the constraint $(\sigma'(v)_a, \sigma'(w)_b) \in R_{ab}$ to R'_{uv}.

We note that the probability that an edge $e = (v, w)$ is chosen in the above distribution is a multiple of $1/(n(dt)^B)$ and we can view the distribution as a weighted CSP instance, where each weight w_e is a multiple of $1/(n(dt)^B)$. Then, we output the CSP instance I' obtained by copying each constraint $n(dt)^B w_e$-many times. Note that the underlying graph of I' is $D := (dt)^B$-regular.

It remains to show that the gap increases by $O(t)$. Although the sensitivity lower bound decreases by D here, we will later offset this by applying degree reduction.

Lemma 5.6. Let $\mathcal{I}$ be a swap-closed family of binary instances of a CSP over Σ such that the underlying graph is an (n, d, λ)-expander, and let $\mathcal{I}' = \text{POWERING}(\mathcal{I})$. Then for any $\varepsilon = O(1/t)$, we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \frac{1}{8D} \cdot \mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I}),$$

for

$$D := (dt)^B, \quad B := 10t \log |\Sigma|, \quad \text{and} \quad \varepsilon' := \frac{t}{C|\Sigma|^4} \cdot \varepsilon,$$

where $C > 0$ is a universal constant.

Let $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$ and $I' = (V, E', \Sigma, \mathcal{R}') = \text{POWERING}(\mathcal{I})$. We design an algorithm T_σ that extracts a solution $\boldsymbol{\sigma}$ for the original instance I from an assignment $\boldsymbol{\sigma}' : V' \to \Sigma$ for I'. This will be done by another probability distribution, using a random walk and truncating probabilities. Fix a vertex $v \in V$ and consider the following: perform a BSRW starting from v, conditioned on this walk ending within t steps. Let $\boldsymbol{w}$ be the final vertex. This gives a probability distribution $\mu_v : \Sigma \to \mathbb{R}$ over opinions $\sigma'(\boldsymbol{w})_v$ of what v's value should be. That is, $\boldsymbol{w}$ contributes the value $\sigma'(\boldsymbol{w})_v$ towards what v's value should be, weighted by the

probability of ending at $\boldsymbol{w}$ in t steps given that we started at v. Then, we consider the *truncated* distribution μ_v^*, which is defined as

$$\mu_v^*(a) = \frac{\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+}{\sum_{b\in\Sigma}\left[\mu_v(b) - \frac{1}{10|\Sigma|}\right]_+}.$$

In particular $\mu_v^*(a) > 0$ only if $\mu_v(a) > 1/(10|\Sigma|)$. The value of $\boldsymbol{\sigma}(v)$ is drawn from the distribution μ_v^*. Our goal is to show that the pair $(\textsc{Powering}, T_\sigma)$ is $(1, 1-\varepsilon, c' = 1, s' = 1-\varepsilon', C_I = D, C_\sigma = 8)$-sensitivity-preserving reduction, where ε' is as in Lemma 5.6. Then, Lemma 5.6 follows by Lemma 3.2. The analysis for c' and C_I is clear, and hence we focus on analyzing s' and C_σ and discuss them in Sections 5.3.2 and 5.3.3, respectively.

5.3.2 Sensitivity Increase in the Recovery Procedure

In this we section, we show the following:

Lemma 5.7. The choice $C_\sigma = 8$ satisfies Item 4 of Definition 3.1.

Proof. Let $I \in \mathcal{I}$ be an instance and I' be the new instance after $\textsc{Powering}$. Let $\sigma', \tilde{\sigma}'$ be two assignments for I' with $\mathsf{Ham}(\sigma', \tilde{\sigma}') = 1$. For $v \in V$, let $\mu_v, \tilde{\mu}_v$ be the distributions constructed as before from σ' and $\tilde{\sigma}'$, and similarly for $\mu_v^*, \tilde{\mu}_v^*$. Let $\boldsymbol{\sigma}, \tilde{\boldsymbol{\sigma}}$ be assignments for I constructed from $\sigma', \tilde{\sigma}'$, respectively. We have

$$\mathsf{EMD}(\boldsymbol{\sigma}, \tilde{\boldsymbol{\sigma}}) \le \sum_{v\in V} \mathsf{TV}(\boldsymbol{\sigma}(v), \tilde{\boldsymbol{\sigma}}(v)) \le \sum_{v\in V} \|\mu_v^* - \tilde{\mu}_v^*\|_1.$$

Let $A_v = \sum_{a\in\Sigma}\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+$ and $\tilde{A}_v = \sum_{a\in\Sigma}\left[\tilde{\mu}_v(a) - \frac{1}{10|\Sigma|}\right]_+$. Let $\delta_{v,a} := \tilde{\mu}_v(a) - \mu_v(a)$. Note that $A_v \ge 9/10$, $\tilde{A}_v \ge 9/10$ and

$$\begin{aligned}
|A_v - \tilde{A}_v| &\le \sum_{a\in\Sigma}\left|\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+ - \left[\tilde{\mu}_v(a) - \frac{1}{10|\Sigma|}\right]_+\right| \\
&\le \sum_{a\in\Sigma}\left|\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+ - \left[\mu_v(a) + \delta_{v,a} - \frac{1}{10|\Sigma|}\right]_+\right| \\
&\le \sum_{a\in\Sigma}|\delta_{v,a}| = \|\mu_v - \tilde{\mu}_v\|_1. \qquad (2)
\end{aligned}$$

Then, we have

$$\begin{aligned}
\|\mu_v^* - \tilde{\mu}_v^*\|_1 &= \sum_{a\in\Sigma}\left|\frac{\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+}{A_v} - \frac{\left[\tilde{\mu}_v(a) - \frac{1}{10|\Sigma|}\right]_+}{\tilde{A}_v}\right| \\
&= \sum_{a\in\Sigma}\left|\frac{\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+}{A_v} - \frac{\left[\mu_v(a) + \delta_{v,a} - \frac{1}{10|\Sigma|}\right]_+}{\tilde{A}_v}\right| \\
&= \sum_{a\in\Sigma}\left|\frac{\tilde{A}_v \cdot \left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+ - A_v \cdot \left[\mu_v(a) + \delta_{v,a} - \frac{1}{10|\Sigma|}\right]_+}{A_v\tilde{A}_v}\right|
\end{aligned}$$

$$
\begin{aligned}
&\le \sum_{a\in\Sigma} \left| \frac{(A_v \pm \|\mu_v - \tilde{\mu}_v\|_1) \cdot \left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+ - A_v \cdot \left[\mu_v(a) + \delta_{v,a} - \frac{1}{10|\Sigma|}\right]_+}{A_v \tilde{A}_v} \right| && \text{(by (2))} \\
&\le \sum_{a\in\Sigma} \left| \frac{\left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+ - \left[\mu_v(a) + \delta_{v,a} - \frac{1}{10|\Sigma|}\right]_+}{\tilde{A}_v} \right| + \sum_{a\in\Sigma} \left| \frac{\|\mu_v - \tilde{\mu}_v\|_1 \cdot \left[\mu_v(a) - \frac{1}{10|\Sigma|}\right]_+}{A_v \tilde{A}_v} \right| \\
&\le \frac{\sum_{a\in\Sigma} |\delta_{v,a}|}{\tilde{A}_v} + \frac{\|\mu_v - \tilde{\mu}_v\|_1}{\tilde{A}_v} \\
&\le 4\|\mu_v - \tilde{\mu}_v\|_1. && \text{(since } \tilde{A}_v \ge 9/10.)
\end{aligned}
$$

Let $w \in V$ be such that $\sigma'(w) \neq \tilde{\sigma}'(w)$ and let $p_{v\to w}$ be the probability that we reach w from a vertex $v \in V$ in the BSRW. Note that w contributes by $2p_{v\to w}$ to the value of $\|\mu_v - \tilde{\mu}_v\|_1$. Then, we have

$$
\sum_{v\in V} \|\mu_v^* - \tilde{\mu}_v^*\|_1 \le 4 \sum_{v\in V} \|\mu_v - \tilde{\mu}_v\|_1 \le 8 \sum_{v\in V} p_{v\to w} = 8. \qquad \square
$$

5.3.3 Gap Analysis

We now prove that the gap increases significantly when t is large (but still constant).

Lemma 5.8. Suppose $\varepsilon = O(1/t)$. Then, the choice $s' = 1 - \varepsilon'$ satisfies Item 2 of Definition 3.1 for

$$
\varepsilon' = \frac{t}{C|\Sigma|^4} \cdot \varepsilon,
$$

where $C > 0$ is a universal constant.

Suppose $\sigma' : V \to \Sigma^{1+d+\cdots+d^t}$ satisfies $\mathsf{val}_{I'}(\sigma') \ge 1 - \varepsilon' = s'$. Our goal is to show that $\mathbf{E}\,\mathsf{val}_I(\boldsymbol{\sigma}) \ge 1 - \varepsilon$. We can handle the situation that we have a random assignment $\boldsymbol{\sigma}'$ for I' by the conditioning argument as in the proof of Lemma 5.3.

Let $\boldsymbol{\sigma}$ be the extracted assignment and let $F_\sigma \subseteq E$ be the set of edges in $G = (V, E)$ whose constraints are violated when $\boldsymbol{\sigma} = \sigma$. We relate the expected number of constraints violated by $\boldsymbol{\sigma}$ to the number constraints violated by σ' using the notion of a "faulty" step in the ASRW.

Definition 5.9 (Faulty step)**.** For an assignment $\sigma : V \to \Sigma$, a σ-faulty step in the ASRW defining an edge $e' = (x, y) \in E'$ is an edge $(u, v) \in E$ along this path satisfying

(i) $(u, v) \in F_\sigma$
(ii) $d_G(x, u) \le t$ and $\sigma'(x)_u = \sigma(u)$
(iii) $d_G(y, v) \le t$ and $\sigma'(y)_v = \sigma(v)$

We further define a step to be σ-faulty* if

1. the step is faulty,
2. the number of steps in the overall walk is at most B.

Let $\boldsymbol{N}_\sigma$ and $\boldsymbol{N}_\sigma^*$ be the numbers of σ-faulty and σ-faulty* steps, respectively, in the ASRW with respect. Let $\boldsymbol{S}$ be the total number of steps. By definition, we have $\boldsymbol{N}_\sigma^* = \boldsymbol{N}_\sigma \cdot \mathbf{1}[\boldsymbol{S} \le B]$. We will use this to bound ε' as follows

$$\varepsilon' \ge \frac{|\{(u,v) \in E' : (\sigma'(u), \sigma'(v)) \notin R'_{uv}\}|}{|E'|} = \Pr_{e \sim E'}[\sigma' \text{ violates } R'_e] \ge \Pr[\boldsymbol{N}_{\boldsymbol{\sigma}}^* > 0] \ge \frac{\mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}}^*]^2}{\mathbf{E}[(\boldsymbol{N}_{\boldsymbol{\sigma}}^*)^2]}, \quad (3)$$

where the final inequality follows by the second moment method. Let $\bar{F} := \mathbf{E}\, F_{\boldsymbol{\sigma}}$. We will later show the following two lemmas.

Lemma 5.10. The following holds:

$$\mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}}^*] \ge \frac{t\bar{F}}{1600|\Sigma|^2 m}.$$

Lemma 5.11. The following holds:

$$\mathbf{E}[(\boldsymbol{N}_{\boldsymbol{\sigma}}^*)^2] \le \left(1 + \frac{1}{1-\lambda/d}\right) \cdot \frac{t\bar{F}}{m} + \frac{2t^2}{m^2}\big((2d+1)\bar{F} + \bar{F}^2\big).$$

Proof of Lemma 5.8. Suppose the first term in Lemma 5.11 is smaller than the second term. Then, we have

$$\begin{aligned} &\left(1 + \frac{1}{1-\lambda/d}\right) \cdot \frac{t\bar{F}}{m} < \frac{2t^2}{m^2}\big((2d+1)\bar{F} + \bar{F}^2\big) \\ \Leftrightarrow &\left(1 + \frac{1}{1-\lambda/d}\right) < \frac{2t}{m}\big((2d+1) + \bar{F}\big) \\ \Leftrightarrow &\, \mathbf{E}\, \mathsf{cost}_I(\boldsymbol{\sigma}) > \frac{1}{2t}\left(1 + \frac{1}{1-\lambda/d}\right) - \frac{2d+1}{m}. \end{aligned}$$

This is a contradiction from the condition that $\mathbf{E}\, \mathsf{cost}_I(\boldsymbol{\sigma}) = O(1/t)$ (by choosing the hidden constant to be small enough).

Now, the first term in Lemma 5.11 is larger than or equal to the second term. Then combining Lemmas 5.10 and 5.11, we obtain from (3)

$$\varepsilon' \ge \mathop{\mathbf{E}}_{\sigma \sim \boldsymbol{\sigma}} \left[\frac{1}{2 \cdot 1600^2 |\Sigma|^4 \left(1 + \frac{1}{1-\lambda/d}\right)} \cdot \frac{t\, \mathbf{E}\, |F_{\boldsymbol{\sigma}}|}{m} \right] \ge \frac{t\, \mathbf{E}\, \mathsf{cost}_I(\boldsymbol{\sigma})}{C|\Sigma|^4},$$

where $C > 0$ is a large enough constant. By setting $\varepsilon' = t/(C|\Sigma|^4) \cdot \varepsilon$, we obtain $\mathbf{E}\, \mathsf{cost}_I(\boldsymbol{\sigma}) \le \varepsilon$. □

We will use the following fact in order to analyze $N_{\boldsymbol{\sigma}}$.

Fact 5.12. Consider an ASRW in a graph G, conditioned on there being exactly k $u \to v$ steps. Let $\boldsymbol{x}, \boldsymbol{y}$ be the initial and final vertices of the walk. Then $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent random variables, where $\boldsymbol{x}$ (resp., $\boldsymbol{y}$) is distributed as a BSRW from u (resp., v).

Proof of Lemma 5.10. We first bound $\boldsymbol{N}_{\boldsymbol{\sigma}}$. Consider conditioning $\boldsymbol{\sigma} = \sigma$ and let $(u,v) \in F_\sigma$ be a violated edge. Then, we have

$$\mathbf{E}[\boldsymbol{N}_\sigma] = \mathbf{E}[\# \ \sigma\text{-faulty } u \to v \text{ steps}]$$

$$= \sum_{k \geq 1} \mathbf{E}[\# \ \sigma\text{-faulty } u \to v \text{ steps} \mid \text{exactly } k \ u \to v \text{ steps}] \cdot \Pr[\text{exactly } k \ u \to v \text{ steps}]$$
$$= \sum_{k \geq 1} k \Pr[u \to v \text{ step is } \sigma\text{-faulty} \mid \text{exactly } k \ u \to v \text{ steps}] \cdot \Pr[\text{exactly } k \ u \to v \text{ steps}], \tag{4}$$

where the last equality holds because either all $u \to v$ steps are σ-faulty or not.

Claim 5.13. We have

$$\Pr[u \to v \text{ step is } \sigma\text{-faulty} \mid \text{exactly } k \ u \to v \text{ steps}] \geq \frac{1}{400|\Sigma|^2}.$$

Proof. Suppose that the ASRW makes exactly k $u \to v$ steps. As $(u, v) \in F_\sigma$, this step is faulty if (ii) and (iii) hold. As $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent by Fact 5.12, we have $\Pr[(ii) \text{ and } (iii)] = \Pr[(ii)] \cdot \Pr[(iii)]$ and as $\Pr[(ii)] = \Pr[(iii)]$, it is enough to show that $\Pr[(ii)] \geq \frac{1}{20|\Sigma|}$. Let $\boldsymbol{x}$ be a random vertex generated by taking a BSRW from u, and let $\boldsymbol{\ell}$ be the number of steps of the walk. Then by Fact 5.12,

$$\begin{aligned}
&\Pr\big[d_G(u, \boldsymbol{x}) \leq t \text{ and } \sigma'(\boldsymbol{x})_u = \sigma(u)\big] \\
&= \Pr\big[d_G(u, \boldsymbol{x}) \leq t \text{ and } \sigma'(\boldsymbol{x})_u = \sigma(u) \mid \boldsymbol{\ell} \leq t\big] \cdot \Pr[\boldsymbol{\ell} \leq t] \\
&= \Pr\big[\sigma'(\boldsymbol{x})_u = \sigma(u) \mid \boldsymbol{\ell} \leq t\big] \cdot \Pr[\boldsymbol{\ell} \leq t] \\
&\geq \frac{1}{2} \cdot \Pr\big[\sigma'(\boldsymbol{x})_u = \sigma(u) \mid \boldsymbol{\ell} \leq t\big] \\
&= \frac{\mu_u(\sigma(u))}{2} \\
&\geq \frac{1}{20|\Sigma|},
\end{aligned}$$

where the first inequality holds because the BSRW halts within t steps with probability $1-(1-1/t)^t \geq 1/2$, the last equality holds because the distribution on $\boldsymbol{x}$ is precisely μ_u — take a BSRW from u, conditioned on stopping within t steps, and the last inequality holds because $\sigma(u)$ is in the support of μ_u^*, which implies that $\mu_u(\sigma(u)) \geq \frac{1}{10|\Sigma|}$. □

By the claim above, we have

$$(4) \geq \sum_{k \geq 1} \frac{k}{400|\Sigma|^2} \cdot \Pr[\text{exactly } k \ u \to v \text{ steps}] = \frac{t}{800|\Sigma|^2 m},$$

where the last equality follows because each step is equally likely to be one of the $2m$ possibilities and the expected total number of steps is t. Hence, we have

$$\mathbf{E}[\boldsymbol{N}_\sigma] \geq \frac{t|F_\sigma|}{800|\Sigma|^2 m}$$

and uncoditioning $\boldsymbol{\sigma}$, we have

$$\mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}}] \geq \frac{t\bar{F}}{800|\Sigma|^2 m}.$$

Next, we bound $\boldsymbol{N}_{\boldsymbol{\sigma}}^*$. Note that

$$\mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}}^*] = \mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}} \cdot \mathbf{1}[\boldsymbol{S} \leq B]] = \mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}} \cdot (1 - \mathbf{1}[\boldsymbol{S} > B)]]$$

$$
\begin{aligned}
&= \mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}}] - \mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}} \cdot \mathbf{1}[\boldsymbol{S} > B]] \\
&\geq \frac{t\bar{F}}{800|\Sigma|^2 m} - \mathbf{E}\left[\boldsymbol{N}_{\boldsymbol{\sigma}} \cdot \mathbf{1}[\boldsymbol{S} > B]\right], \qquad (5)
\end{aligned}
$$

We can bound the second term as

$$
\begin{aligned}
&\mathbf{E}\left[\boldsymbol{N}_{\boldsymbol{\sigma}} \cdot \mathbf{1}[\boldsymbol{S} > B]\right] = \Pr[\boldsymbol{S} > B] \cdot \mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}} \mid \boldsymbol{S} > B] = \left(1 - \frac{1}{t}\right)^B \cdot \mathbf{E}[\boldsymbol{N}_{\boldsymbol{\sigma}} \mid \boldsymbol{S} > B] \\
&\geq \exp\left(-\frac{eB}{(e-1)t}\right) \cdot \mathbf{E}[\boldsymbol{S} \mid \boldsymbol{S} > B] \cdot \frac{\bar{F}}{m} && \text{(by } e^{-x} \leq 1 - \tfrac{e-1}{e}x \text{ for } x \in [0,1]\text{)} \\
&= \exp\left(-\frac{eB}{(e-1)t}\right) \cdot (B + t) \cdot \frac{\bar{F}}{m} \\
&\geq \frac{1}{|\Sigma|^{10e/(e-1)}} \cdot (20t \log|\Sigma|) \cdot \frac{\bar{F}}{m} \\
&\geq \frac{t\bar{F}}{1600|\Sigma|^2 m}, && \text{(assuming } |\Sigma| \text{ is large enough)}
\end{aligned}
$$

Finally, by (5), we conclude that

$$
\mathbf{E}[\boldsymbol{N}^*_{\boldsymbol{\sigma}}] \geq \frac{t\bar{F}}{1600|\Sigma|^2 m}. \qquad \square
$$

To prove Lemma 5.11, we use the following result, which states that the second moment of $F_{\boldsymbol{\sigma}}$ is not too large compared to its first moment.

Lemma 5.14. We have

$$
\mathbf{E}\,|F_{\boldsymbol{\sigma}}|^2 \leq (2d+1)\bar{F} + \bar{F}^2.
$$

Proof. For an edge $e \in E$, let $\boldsymbol{X}_e$ be the indicator of the event that $\boldsymbol{\sigma}$ violates the constraint (e, R_e), and let $p_e = \mathbf{E}\,X_e$ be the probability of the event. For two edges $e, f \in E$, we write $e \sim f$ to denote that they are incident. Note that $\boldsymbol{X}_e$ and $\boldsymbol{X}_f$ are independent if $e \not\sim f$. Then, we have

$$
\begin{aligned}
&\mathbf{E}\,|F_{\boldsymbol{\sigma}}|^2 = \mathbf{E}\left(\sum_{e \in E} \boldsymbol{X}_e\right)^2 \\
&= \mathbf{E}\sum_{e \in E} \boldsymbol{X}_e^2 + \mathbf{E}\sum_{e,f \in E: e \not\sim f} \boldsymbol{X}_e \boldsymbol{X}_f + \mathbf{E}\sum_{e,f \in E: e \sim f} \boldsymbol{X}_e \boldsymbol{X}_f \\
&\leq \mathbf{E}\,|F_{\boldsymbol{\sigma}}| + \left(\mathbf{E}\sum_{e \in E} \boldsymbol{X}_e\right)^2 + \sum_{e,f \in E: e \sim f} \max\{p_e, p_f\} \\
&\leq \mathbf{E}\,|F_{\boldsymbol{\sigma}}| + (\mathbf{E}\,|F_{\boldsymbol{\sigma}}|)^2 + 2d\sum_{e \in E} p_e \\
&\leq (2d+1)\,\mathbf{E}\,|F_{\boldsymbol{\sigma}}| + (\mathbf{E}\,|F_{\boldsymbol{\sigma}}|)^2 \\
&\leq (2d+1)\bar{F} + \bar{F}^2. \qquad \square
\end{aligned}
$$

Proof of Lemma 5.11. Although the proof is similar to that of Lemma 5.2 in [Din07], we need to account for the fact that $\boldsymbol{\sigma}$ is a random variable.

Consider conditioning on $\boldsymbol{\sigma} = \sigma$. Let $\boldsymbol{S}_i$ be the indicator random variable which is 1 iff the i-th step of the ASRW is in F_σ, and let $\boldsymbol{M} := \sum_{i=1}^{\infty} \boldsymbol{S}_i$ be the number of steps of the ASRW that were within F_σ. Then, we have

$$\begin{aligned}
\mathbf{E}[(\boldsymbol{N}_\sigma^*)^2] \le \mathbf{E}[\boldsymbol{M}^2] &= \sum_{i,j=1}^{\infty} \mathbf{E}[\boldsymbol{S}_i \boldsymbol{S}_j] \\
&\le 2 \sum_{i=1}^{\infty} \Pr[\boldsymbol{S}_i = 1] \cdot \sum_{j \ge i} \Pr[\boldsymbol{S}_j = 1 \mid \boldsymbol{S}_i = 1] \\
&= 2 \sum_{i=1}^{\infty} \Pr[\boldsymbol{S}_i = 1] \left(\Pr[\boldsymbol{S}_i = 1 \mid \boldsymbol{S}_i = 1] + \sum_{j > i} \Pr[\boldsymbol{S}_j = 1 \mid \boldsymbol{S}_i = 1] \right) \\
&= 2 \sum_{i=1}^{\infty} \Pr[\boldsymbol{S}_i = 1] \left(1 + \sum_{j > i} \Pr[\boldsymbol{S}_j = 1 \mid \boldsymbol{S}_i = 1] \right).
\end{aligned}$$

Now,

$$\begin{aligned}
\Pr[\boldsymbol{S}_j = 1 \mid \boldsymbol{S}_i = 1] &= \Pr[\text{The ASRW takes } j - i \text{ more steps}] \cdot \Pr[(j-i)\text{-th step is in } F_\sigma] \\
&\le \left(1 - \frac{1}{t}\right)^{j-i} \left(\frac{|F_\sigma|}{m} + \left(\frac{\lambda}{d}\right)^{\ell - 1} \right),
\end{aligned}$$

by Proposition 5.4 of [Din07]), as G is a (n, d, λ)-expander.

Substituting this into the previous bound, we have

$$\begin{aligned}
\mathbf{E}[(\boldsymbol{N}_\sigma^*)^2] &\le 2 \sum_{i=1}^{\infty} \Pr[\boldsymbol{S}_i = 1] \left(1 + \sum_{\ell=1}^{\infty} \left(1 - \frac{1}{t}\right)^{\ell} \left(\frac{|F_\sigma|}{m} + \left(\frac{\lambda}{d}\right)^{\ell-1} \right) \right) \\
&\le 2 \sum_{i=1}^{\infty} \Pr[\boldsymbol{S}_i = 1] \left(1 + (t-1)\frac{|F_\sigma|}{m} + \sum_{\ell=1}^{\infty} \left(\frac{\lambda}{d}\right)^{\ell-1} \right) && \text{(Since } 1 - 1/t < 1) \\
&\le 2 \sum_{i=1}^{\infty} \Pr[\boldsymbol{S}_i = 1] \left(1 + \frac{t|F_\sigma|}{m} + \frac{1}{1 - \lambda/d} \right) && \text{(Since } \lambda < d) \\
&= 2 \left(1 + \frac{t|F_\sigma|}{m} + \frac{1}{1 - \lambda/d} \right) \mathbf{E}[\boldsymbol{M}] \\
&\le 2 \left(1 + \frac{t|F_\sigma|}{m} + \frac{1}{1 - \lambda/d} \right) \cdot \frac{t|F_\sigma|}{m}.
\end{aligned}$$

By unconditioning $\boldsymbol{\sigma}$, we obtain

$$\begin{aligned}
\mathbf{E}[(\boldsymbol{N}_{\boldsymbol{\sigma}}^*)^2] &\le 2 \left(1 + \frac{1}{1 - \lambda/d} \right) \cdot \frac{t\bar{F}}{m} + \frac{2t^2}{m^2} \mathbf{E}\, |F_{\boldsymbol{\sigma}}|^2 \\
&\le \left(1 + \frac{1}{1 - \lambda/d} \right) \cdot \frac{t\bar{F}}{m} + \frac{2t^2}{m^2} \big((2d+1)\bar{F} + \bar{F}^2 \big), && \text{(by Lemma 5.14)}
\end{aligned}$$

which completes the proof. □

5.3.4 Sensitivity Recovery

The gap amplification step decreases the sensitivity bound by $D = (dt)^B$, but we can offset it by combining it with degree reduction:

Lemma 5.15. Let $\mathcal{I}$ be a swap-closed family of instances of a binary CSP such that the underlying graph is an (n, d, λ)-expander, and let $\mathcal{I}' = \textsc{DegreeReduction}(\textsc{Powering}(\mathcal{I}))$. Then for any $\varepsilon = O(1/t)$, we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \Omega\big(\mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})\big),$$

for

$$\varepsilon' := \frac{t}{C|\Sigma|^4} \cdot \varepsilon,$$

where $C > 0$ is a universal constant.

Proof. The claim follows by combining Lemmas 5.3 and 5.6 and noting that every instance in $\textsc{Powering}(\mathcal{I})$ is D-regular, where D is as in the statement of Lemma 5.6. □

5.4 Alphabet Reduction

The previous step leaves us with a set of constraints over an alphabet of size $|\Sigma|^{1+d+\ldots+d^t}$. In this section, we introduce a procedure $\textsc{AlphabetReduction}$ which decreases the alphabet size while preserving the gap, and not decreasing the sensitivity by too much. This procedure is identical to the construction of Dinur [Din07]. Our contribution is a new procedure T_σ which recovers an assignment to the instance prior to $\textsc{AlphabetReduction}$ from the instance after. This new procedure will allow us to show that $(\textsc{AlphabetReduction}, T_\sigma)$ is a sensitivity-preserving reduction.

We begin by recalling the $\textsc{AlphabetReduction}$ of Dinur. The heart of which is the sub-routine $\textsc{AssignmentTester}$, which will be ran on each edge of the original instance.

5.4.1 Assignment Tester

We say that two assignments $x \in \{0,1\}^k$ and $y \in \{0,1\}^k$ are δ-*far* if $\mathsf{Ham}(x, y) \geq \delta \cdot k$.

Lemma 5.16 (Assignment Tester [Din07])**.** There is a reduction which takes as input a Boolean circuit $C : \{0,1\}^X \to \{0,1\}$ over Boolean variables X and outputs a binary CSP instance $I := (X \cup T, E, \Sigma_0, \mathcal{R})$ over Boolean variables X and an additional set of variables T over an alphabet of size $|\Sigma_0| \leq 2^6$. Furthermore, for every assignment $\alpha \in \{0,1\}^X$:

- If $C(\alpha) = 1$, then there exists $\beta \in \Sigma_0^T$ such that (α, β) satisfies every constraint in I, otherwise
- If α is δ-far from every $\alpha^* \in \{0,1\}^X$ for which $C(\alpha^*) = 1$, then for every $\beta \in \Sigma_0^T$, at least a $\delta/48$ fraction of the constraints of I are falsified by (α, β).

As it will be pertinent to our analysis, we recall the construction of the $\textsc{AssignmentTester}$, run on a Boolean circuit C with input variables X.

Variable Introduction. We construct the sets of additional Boolean variables Y and Z as follows:

1. Arithmetize the circuit C as a set of quadratic equations $\mathcal{P}$ such that $C(x) = 1$ if and only if $P(x) = 1$ for every $P \in \mathcal{P}$. To do so, we introduce, for each gate g in C, a new Boolean variable y_g. Let Y be the collection of all y_g variables introduced. As well, we add the following quadratic constraints to $\mathcal{P}$: for every gate g in C with children g_1, g_2, we add one of the following constraints:
   - If $g = \wedge$: $y_{g_1} y_{g_2} - y_g = 0$ if g_1, g_2 are the children to g.
   - If $g = \vee$: $y_{g_1} + y_{g_2} + y_{g_1} y_{g_2} - y_g = 0$ if g_1, g_2 are the children to g.
   - If $g = \neg$: $y_{g_1} + y_g - 1 = 0$ if g_1 is the child to g.

   For the remaining pairs (g_i, g_j) and triples (g_i, g_j, g_k), with $i \geq j, k$, for which we have not introduced a constraint, we add a trivial constraint which is always satisfied. Let $k := |X| + |Y| = |X| + |C|$, where $|C|$ denotes the number of gates in the circuit C.
2. Introduce sets of variables $L \in \{0,1\}^k$ and $Q \in \{0,1\}^{k^2}$, and let $Z = L \cup Q$. These variables will purportedly represent the Hadamard encoding L of an assignment $\alpha \in \{0,1\}^{X \cup Y}$ and the Hadamard encoding of $\alpha \otimes \alpha$. We will think of L as a table indexed by strings $s \in \{0,1\}^k$ such that $L(s) = \sum_{i=1}^{k} s_i \alpha_i$, and similarly for Q; that is, $Q(t) = \sum_{1 \leq i,j \leq k} t_{ij} \alpha_i \alpha_j$ for $t \in \{0,1\}^{k^2}$.

In total, $Y \cup Z$ contains $O(k) + 2^{k(k+1)}$-many Boolean variables.

Constraint Introduction. The purpose of the set of constraints that will be introduced is to verify that an assignment to the variables $\{0,1\}^{X \cup Y \cup Z}$ encodes a satisfying assignment to the circuit C. We will describe the constraints using a probabilistic language, where each edge e that is introduced will have an associated weight. Furthermore, the edges will initially be hyperedges (involving up to 6 variables); they will be reduced to regular edges in the *sparsification* step below. We will construct a *weighted* instance $I := (X \cup Y \cup Z, E, \{0,1\}, \mathcal{R})$, by populating the edge set E and constraint set $\mathcal{R}$. Every time we "introduce a constraint" we add an edge to E and a corresponding constraint R_e to $\mathcal{R}$. We will then normalize the instance by adding copies of each edge $e \in E$ proportional to their weight to obtain the final instance.

1. Verify that L is a Hadamard code (of some assignment in $\{0,1\}^k$): Sample $x, y \sim \{0,1\}^k$ and check that $L(x) + L(y) = L(x+y)$; that is, we are running the BLR Linearity test [BLR93] on L. That is, for every $x, y \in \{0,1\}^k$ we introduce a constraint which accepts iff $L(x) + L(y) = L(x+y)$, with associated weight $w_1 := 2^{-2k}$.
2. Verify that Q is a Hadamard code (of some assignment in $\{0,1\}^{k^2}$): For every $x, y \in \{0,1\}^{k^2}$, introduce a constraint which accepts iff $Q(x) + Q(y) = Q(x+y)$, with associated weight $w_2 := 2^{-2k^2}$.
3. Verify that L and Q encode the same assignment: Let $\textsc{SelfCorrect}(L, x)$ be the procedure which samples $x' \sim \{0,1\}^k$ and outputs $L(x+x') - L(x')$. In this step we sample $x, y \sim \{0,1\}^k$ and check whether
$$\textsc{SelfCorrect}(L, x)\textsc{SelfCorrect}(L, y) = \textsc{SelfCorrect}(Q, x \otimes y).$$
   That is, for every $x, y, x', y' \in \{0,1\}^k$ and $q \in \{0,1\}^{k^2}$, we introduce the constraint
$$\big(L(x+x') - L(x')\big)\big(L(y+y') - L(y')\big) = Q(x \otimes y + q) - Q(q),$$
   with weight $w_3 := 2^{-(4k+k^2)}$.

4. Verify that the assignment that is referred to by L and Q satisfies the circuit C: Sample $r \sim \{0,1\}^{|\mathcal{P}|}$ and let $s_0 \in \{0,1\}, s \in \{0,1\}^k, t \in \{0,1\}^{k^2}$ be such that

$$\sum_{P \in \mathcal{P}} r_P P(x) = s_0 + \sum_{i=1}^{k} s_i x_i + \sum_{1 \le i,j \le k} t_{ij} x_i x_j,$$

and check whether $s_0 + \textsc{SelfCorrect}(L, s) + \textsc{SelfCorrect}(Q, t) = 0$. That is, for every $r \in \{0,1\}^{|\mathcal{P}|}, x \in \{0,1\}^k, q \in \{0,1\}^{k^2}$ introduce the constraint

$$s_0 + L(s + x) - L(x) + Q(t + q) - Q(q) = 0,$$

with weight $w_4 := 2^{-(|C|+k+k^2)}$, noting that $|\mathcal{P}| = |C|$.
5. Verify that the assignment referred to by L and Q is the same assignment as encoded by the variables X. Let e_i denote the ith standard basis vector. Sample $i \sim [|X|]$ and check whether

$$\textsc{SelfCorrect}(L, e_i) = X_i.$$

That is, for every $x \in \{0,1\}^k$, and every $1 \le i \le |X|$, introduce $L(e_i + x) - L(x) = X_i$. Weight each of these constraints by $w_5 := 2^{-k}/|X|$.

We convert this instance I to an unweighted instance by replacing each constraint edge pair (e, R_e) of type (i), for $i \in [5]$, with $w_i N$-many copies of it, where N is the least common multiple of $1/w_1, \ldots, 1/w_5$. Observe that each of these constraints depend on at most 6 variables in $X \cup Y \cup Z$. Finally, as we would like to measure swap sensitivity (rather than sensitivity) we would like to be able to recover any instance obtained by swapping C for another circuit C of the same size (encoding a constraint in our instance).[2] To accommodate this, let α be the maximum number of times that a specific $L(s)$ or $Q(q)$ appears in the constraints obtained from step (4). For every $s, x \in \{0,1\}^k$ and $t, q \in \{0,1\}^{k^2}$ for which a constraint of type (4) does not exist, we introduce a trivial (always satisfied) constraint on the variables $L(s+x), L(x), Q(t+q), Q(q)$, with multiplicity $\alpha w_4 N$.

It remains to reduce these to binary constraints.

Sparsification. Let $I = (X \cup Y \cup Z, E, \Sigma_0, \mathcal{R})$ be the instance constructed so far. We reduce these 6-ary constraints to binary constraints by introducing an additional set of 2^6-ary variables $W := \{w_e : e \in E\}$. We replace each pair of edge and constraint (e, R_e) belonging to E and $\mathcal{R}$ as follows: Let $v_1, \ldots, v_t$ for $2 \le t \le 6$ be the set of variables on which R_e depends. Replace R_e by constraints $R_{e,i}$ on (w_e, v_i) for $i \in [t]$, where $(a, b) \in R_{e,i}$ if b is consistent with a and a satisfies R_e. Let $I = (X \cup T, E, \Sigma_0, \mathcal{R})$, where $T := X \cup Y \cup Z \cup W$, be the resulting instance that we have constructed. This completes our description of the $\textsc{AssignmentTester}$.

We record some useful observations.

Observation 5.17. Let C be any circuit on variables X and consider $I = \textsc{AssignmentTester}(C)$. Then following hold:

1. The parameters N and k depend only on the number of gates of C.

[2] When we apply the Assignment Tester to the constraints of our instance, we will assume that every constraint is encoded by a circuit of the same size, by taking the maximum.

2. Replacing C by any other circuit of the same size on the input variables X will only modify $O(N)$-many constraints from (4).
3. Each X-variable has degree $D := 3N/|X|$, as each constraint in step (5) depends on exactly three variables and by the sparsification step we replace each such constraint by three binary constraints.

5.4.2 Reduction Procedure

Now, we describe the procedure ALPHABETREDUCTION, building on ASSIGNMENTTESTER. Let $I = (V, E, \Sigma, \mathcal{R}) \in \mathcal{I}$ be a binary CSP instance with n variables. First, we encode the elements of Σ in binary by applying the Hadamard code $\mathsf{Had} : \Sigma \to \{0,1\}^\ell$, where $\ell := 4\log|\Sigma|$; hence each vertex $v \in V$ is encoded by a set X_v of ℓ-many Boolean variables. For each edge $e = (u, v) \in E$ we will replace the constraint $R_e \subseteq \Sigma^2$ by a Boolean circuit $C_e : \{0,1\}^{2\ell} \to \{0,1\}$ such that $C_e(\alpha, \beta) = 1$ iff there are $a, b \in \Sigma$ such that $\mathsf{Had}(a) = \alpha, \mathsf{Had}(b) = \beta$, and $(a, b) \in R_e$. Let $X_e = X_u \cup X_v$ be the set of Boolean variables (encoding the two endpoints of e) on which C_e depends. Finally, we will assume without loss of generality that the size of the circuits C_e is the same for all $e \in E$ by padding. In particular, by Observation 5.17, the parameters N and k will be the same for every $e \in E$.

To generate the instance $I' := \text{ALPHABETREDUCTION}(I)$, we will feed the circuit C_e that simulates each constraint R_e to the ASSIGNMENTTESTER. Let $I_e = (X_e \cup T_e, E_e, \Sigma_0, \mathcal{R}_e) := \text{ASSIGNMENTTESTER}(C_e)$. Define the instance $I' := (X' \cup T', E', \Sigma' = \Sigma_0, \mathcal{R}')$ as $X' = \bigcup_{e\in E} X_e$, $T' = \bigcup_{e\in E} T_e$, $E' := \bigcup_{e\in E} E_e$, and $\mathcal{R}' := \bigcup_{e\in E} \mathcal{R}_e$.

The goal of this section is to show the following:

Lemma 5.18. Let $\mathcal{I}$ be a swap-closed family of CSP instances and let $\mathcal{I}' = \text{ALPHABETREDUCTION}(\mathcal{I})$. Then we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \frac{\mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})}{C_I \cdot C_\sigma},$$

where $C_I = N$, $C_\sigma = O(1/\log|\Sigma|)$, and $\varepsilon' \geq \varepsilon/C$ for some universal constant $C > 0$, and parameter N from the ASSIGNMENTTESTER.

Consider the following algorithm T_σ, which given an assignment $\sigma' : V' \to \Sigma'$ to I', returns an assignment $\boldsymbol{\sigma} : V \to \Sigma$ to I as follows. For each $u \in V$, denote by $\sigma'[u]$ the restriction of σ' to the block of variables which represent u's label. Then $\sigma'[u]$ is a purported Hadamard codeword on $\ell = 4\log|\Sigma|$-many bits. Let c_u be the closest codeword to $\sigma'[u]$. Consider the "Swiss cheese" $S \subseteq \{0,1\}^\ell$ obtained from the hypercube $\{0,1\}^\ell$ by removing any point which is within the unique decoding radius (the Hamming ball of radius $\ell/4$) of any codeword; an illustration is given in Figure 3. Let $\delta_u = \mathsf{Ham}(\sigma'[u], c_u)/\ell$. Note that $\delta_u \in [0, 1/4]$. Then, we choose a threshold $\tau \in [0, 1]$ uniformly at random (we use this threshold for all $u \in V$). Then, we set $\boldsymbol{\sigma}(u) = a$ if $4\delta_u \leq \tau$, where $a \in \Sigma$ is such that $\mathsf{Had}(a) = c_u$. Otherwise, we set $\boldsymbol{\sigma}(u) = r$, where r is an arbitrary but fixed label in Σ. Let μ_u be this distribution, where $\mu_u(1)$ denotes the first event and $\mu_u(2)$ denotes the second.

We claim that $(\text{ALPHABETREDUCTION}, T_\sigma)$ is a $(1, 1-\varepsilon, c' = 1, s' = 1-\varepsilon', C_I = N, C_\sigma = 8/\ell)$-sensitivity-preserving reduction, where $\varepsilon' = \Theta(\varepsilon)$. The analysis for c' is obvious. Also, $C_I = N$ suffices by Observation 5.17. Hence, we focus on analyzing C_σ and s' in the rest of this section.

5.4.3 Sensitivity Increase in the Recovery Procedure

We now analyze the sensitivity increase in the recovery procedure T_σ.

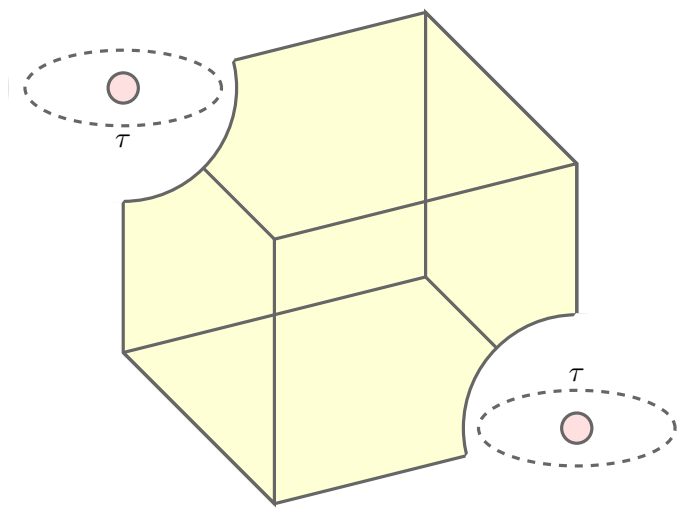


Figure 3: Conceptual illustration of the "Swiss cheese" $S \subseteq \{0,1\}^\ell$. Vertices of $\{0,1\}^\ell$ which are codewords are indicated by red circles, and the Hamming balls of radius $\ell/4$ around each one of them has been removed from $\{0,1\}^\ell$ to form S. The random threshold τ chosen in the recovery procedure is indicated by the dashed circle.

Lemma 5.19. The choice $C_\sigma = 8/\ell$ satisfies Item 4 of Definition 3.1.

Proof. Consider any pair of assignments $\sigma', \tilde{\sigma}' : V' \to \Sigma'$ such that $\mathsf{Ham}(\sigma', \tilde{\sigma}') = 1$. As well, let $\boldsymbol{\sigma} = T_\sigma(\sigma')$ and $\tilde{\boldsymbol{\sigma}} = T_\sigma(\tilde{\sigma}')$. Then,

$$\mathsf{EMD}(\boldsymbol{\sigma}, \tilde{\boldsymbol{\sigma}}) \leq \sum_{u \in V} \mathsf{TV}\Big(\boldsymbol{\sigma}(u),\ \tilde{\boldsymbol{\sigma}}(u)\Big).$$

Suppose $\mathsf{Ham}(\sigma'[u], \tilde{\sigma}'[u]) = 1$ for a vertex $u \in V$. Let $c_u, \tilde{c}_u \in \{0,1\}^\ell$ be the closest codewords to $\sigma'[u]$ and $\tilde{\sigma}'[u]$, respectively. we consider two cases based on whether these codewords are the same.

Case 1. If $c_u = \tilde{c}_u$: Then,

$$\begin{aligned}
&\mathsf{TV}(\boldsymbol{\sigma}(u),\ \tilde{\boldsymbol{\sigma}}(u)) = \mathsf{TV}(\mu_u, \tilde{\mu}_u) = \frac{1}{2}\Big(|\mu_u(1) - \tilde{\mu}_u(1)| + |\mu_u(2) - \tilde{\mu}_u(2)|\Big) \\
&= \frac{1}{2 \cdot (\ell/4)}\Big(|\mathsf{Ham}(\sigma'[u], c_u) - \mathsf{Ham}(\tilde{\sigma}'[u], c_u)| + |\mathsf{Ham}(\sigma'[u], S) - \mathsf{Ham}(\tilde{\sigma}'[u], S)|\Big) \\
&\leq \frac{1}{\ell/4} \cdot \mathsf{Ham}(\sigma'[u], \tilde{\sigma}'[u]). && \text{(By the triangle inequality)} \\
&= \frac{4}{\ell}.
\end{aligned}$$

Case 2. If $c_u \neq \tilde{c}_u$: As $\mathsf{Ham}(\sigma', \tilde{\sigma}') = 1$, this can only occur if $\sigma'[u], \tilde{\sigma}'[u]$ lie on the unique decoding radius of their respective codewords. Let $a, b \in \Sigma$ be such that $\mathsf{Had}(a) = c_u, \mathsf{Had}(b) = \tilde{c}_u$. Then $\boldsymbol{\sigma}(u), \tilde{\boldsymbol{\sigma}}(u)$ take on values a, b, respectively, with probability at most $1/(\ell/4)$. Then, we have

$$\mathsf{TV}(\boldsymbol{\sigma}(u),\ \tilde{\boldsymbol{\sigma}}(u)) \leq \Big|\Pr\left[\boldsymbol{\sigma}(u) = a\right] - \Pr\left[\tilde{\boldsymbol{\sigma}}(u) = b\right]\Big| \leq \frac{2}{\ell/4} = \frac{8}{\ell}.$$

Hence, the claim follows. □

5.4.4 Gap Analysis

Now we analyze the gap decrease of ALPHABETREDUCTION.

Lemma 5.20. The choice $s' = 1 - \varepsilon'$ for $\varepsilon' = \varepsilon/C$ satisfies Item 2 of Definition 3.1, where $C > 0$ is a universal constant.

Proof. Let σ' be an assignment to I' with $\mathsf{val}_{I'}(\sigma') \geq 1 - \varepsilon'$. Let $\boldsymbol{\sigma}$ be the assignment recovered from $\boldsymbol{\sigma}'$ by T_σ. Our goal is to show that $\mathbf{E}\,\mathsf{val}_I(\boldsymbol{\sigma}) \geq 1 - \varepsilon$. We can handle the situation that we have a random assignment $\boldsymbol{\sigma}'$ for I' by the conditioning argument as in the proof of Lemma 5.3.

(Generalizing this argument to the case where σ' is a random assignment is straightforward.)

For $e \in E$, let ε_e be the probability that $\boldsymbol{\sigma}$ violates the constraint on e. Also, let ε'_e be the fraction of the constraints in I_e violated by σ'. Note that $\sum_{e \in E} \varepsilon'_e/m \leq \varepsilon'$. For $u \in V$, let $c_u \in \{0,1\}^\ell$ be the closest codeword to $\sigma'[u]$. Let $a_u \in \Sigma$ be such that $\mathsf{Had}(a_u) = c_u$. Let $\delta_u = \mathsf{Ham}(\sigma'[u], c_u)/\ell$. Then, the probability that $\boldsymbol{\sigma}(u) = a_u$ is

$$p_u := \frac{\ell/4 - \mathsf{Ham}(\sigma'[u], c_u)}{\ell/4} = \frac{1/4 - \delta_u}{1/4} = 1 - 4\delta_u.$$

Fix an edge $e = (u, v) \in E$ of I. Let (c^*_u, c^*_v) be the satisfying assignment to I_e which is closest to $(\sigma'[u], \sigma'[v])$. As (c^*_u, c^*_v) is satisfying, they must be Hadamard codewords, and hence there are $a^*_u, a^*_v \in \Sigma$ such that $\mathsf{Had}(a^*_u) = c^*_u$, $\mathsf{Had}(a^*_v) = c^*_v$, and $(a^*_u, a^*_v) \in R_e$. Let p_e be the probability that both $\boldsymbol{\sigma}(u) = a_u$ and $\boldsymbol{\sigma}(v) = a_v$ hold. Because we use threshold rounding, we have $p_e = 1 - \max\{4\delta_u, 4\delta_v\} = \min\{p_u, p_v\}$.

We consider two cases.

Case 1. $(a_u, a_v) \notin R_e$. Then at least one of $a^*_u \neq a_u$ or $a^*_v \neq a_v$. Suppose without loss of generality that the first holds. Then,

$$\begin{aligned}
\frac{\ell}{2} &\leq \mathsf{Ham}\Big(\mathsf{Had}(a^*_u),\ \mathsf{Had}(a_u)\Big) && \text{(As Hadamard codewords have distance } 1/2.) \\
&= \mathsf{Ham}\Big(c^*_u,\ c_u\Big) && (\mathsf{Had}(a_u) = c_u) \\
&\leq \mathsf{Ham}\Big(c^*_u, \sigma'[u]\Big) + \mathsf{Ham}\Big(\sigma'[u], c_u\Big) \\
&\leq 2\mathsf{Ham}\Big(c^*_u, \sigma'[u]\Big),
\end{aligned}$$

where the final inequality follows as c_u is the closest codeword to $\sigma'[u]$. Hence, $(\sigma'[u], \sigma'[v])$ is $(1/8)$-far from (c^*_u, c^*_v), and hence $(1/(8 \cdot 48))$-far from any satisfying assignment to I_e by Lemma 5.16. This means $\varepsilon'_e \geq 1/(8 \cdot 48) \geq \varepsilon_e/384$, where the last inequality is by $\varepsilon_e \leq 1$.

Case 2. $(a_u, a_v) \in R_e$. Suppose without loss of generality that $\delta_u \geq \delta_v$ holds. Then, $\mathsf{Ham}(\sigma'[u], c^*_u) \geq \mathsf{Ham}(\sigma'[u], c_u) \geq \delta_u \ell$ holds because c_u is the closest codeword to $\sigma'[u]$. Hence, $\sigma'([u], [v])$ is at least $(\delta_u/2)$-far from (c^*_u, c^*_v), and it follows that $(\delta_u/2)$-far from any satisfying assignment to I_e by Lemma 5.16. This implies $\varepsilon'_e \geq \delta_u/(2 \cdot 48)$.

On the other hand, the probability that we have both $\boldsymbol{\sigma}(u) = a_u$ and $\boldsymbol{\sigma}(v) = a_v$ is p_u. Hence $\epsilon_e \leq 1 - p_u = 4\delta_u$ and we have $\varepsilon'_e \geq \varepsilon_e/(2 \cdot 48 \cdot 4) = \varepsilon_e/384$.

Combining the two cases, we have

$$\varepsilon' = \frac{1}{m} \sum_{e \in E} \varepsilon'_e \geq \frac{1}{m} \sum_{e \in E} \frac{\varepsilon_e}{384} = \frac{\mathsf{cost}_I(\boldsymbol{\sigma})}{384}.$$

Hence, we have $\mathsf{cost}_I(\boldsymbol{\sigma}) \leq \epsilon$ by setting $\varepsilon' = \varepsilon/384$. □

5.4.5 Sensitivity Recovery

Finally, we combine the ALPHABETREDUCTION procedure with DEGREEREDUCTION in order to restore the our sensitivity bound.

Lemma 5.21. Let $\mathcal{I}$ be a swap-closed family of instances of a binary CSP and consider the family of instances $\mathcal{I}' = \text{DEGREEREDUCTION}(\text{ALPHABETREDUCTION}(\mathcal{I}))$. Then for any $\varepsilon > 0$, we have

$$\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) \geq \Omega\big(\mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})\big),$$

for $\varepsilon' = \varepsilon/C$ for some universal constant $C > 0$.

Proof. Let $I' = (X' \cup T', E', \Sigma' = \Sigma_0, \mathcal{R}') = \text{ALPHABETREDUCTION}(I)$ for an instance $I \in \mathcal{I}$, where X' are the vertices which represent the Hadamard encoding of the vertices of I. Let D be the maximum degree of any X'-vertex among all $I \in \mathcal{I}'$. Our aim is to apply Corollary 5.4 to I'. Define the marked CSP $\hat{I}'$ which is I' with marked set of vertices $S = X'$. Then by Lemma 5.18 and Corollary 5.4, we have

$$\begin{aligned}\mathsf{SwapSens}_{1,1-\varepsilon'}(\mathsf{SwapClo}(\mathcal{I}')) &\geq \frac{D \cdot \mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})}{C_I C_\sigma} \\ &= \Omega\left(\frac{N/\log|\Sigma| \cdot \mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})}{N \cdot (1/\log|\Sigma|)}\right) = \Omega\big(\mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})\big).\end{aligned}$$

□

5.5 Proof of Theorem 5.2

Proof of Theorem 5.2. Starting with the $\Omega(1/n)$ gap between completeness and soundness from Lemma 4.1, we gradually increase this gap by applying the reductions developed in this section.

Each round consists of Lemma 5.5, Lemma 5.15 with a sufficiently large constant t, and Lemma 5.21, in this order, and we apply the reductions for $r := O(\log_t n)$ rounds, where $\delta > 0$ is a sufficiently small constant. The gap amplification and subsequent degree reduction steps increase the number of vertices by a factor of $O(d^t)$, where d is the degree of the instance at the beginning of the round, and the alphabet reduction and subsequent degree reduction steps increase the number of vertices by a factor of $2^{O(k^2)}$, where k is from ASSIGNMENTTESTER. The instance size after r rounds is

$$N := n \cdot \left(O\big(d^t\big) \cdot 2^{O(k^2)}\right)^r = n^{O(t/\log t)}.$$

This implies $n = N^{\Omega(\log t/t)}$. Each round decreases the sensitivity lower bound by a constant. Then after r rounds, the sensitivity lower bound becomes

$$\frac{n}{O(1)^r} = n^{1-O(1/\log t)} = N^{\Omega(\log t/t)}.$$

Each round increases the gap between the completeness and soundness by t/C for some universal constant $C > 0$. Hence by choosing the hidden constant in r to be large enough, the gap after r rounds is

$$\Omega\left(\frac{1}{n}\right) \cdot \left(\frac{t}{C}\right)^r = \Omega(1).$$

To obtain label cover instances, observe that the sparsification process in the alphabet reduction step generates a label cover instance. Therefore, we can obtain label cover instances simply by omitting the degree reduction step at the end. Finally, Lemma 2.3 yields the theorem. □

6 Maximum Clique and Related Problems

A vertex set $S \subseteq V$ in a graph $G = (V, E)$ is called a *clique* if every pair of vertices in S is adjacent in G. In the *maximum clique problem*, given a graph $G = (V, E)$, the goal is to find a clique of maximum size. We first show a sensitivity lower bound for $(1-\varepsilon)$-approximation algorithm for some $\varepsilon > 0$ in Section 6.1. As a corollary, we also obtain a lower bound for $(1+\varepsilon)$-approximation algorithm for the minimum vertex cover problem. Then, we show a sensitivity lower bound for $n^{-\varepsilon}$-approximation algorithm for some $\varepsilon > 0$ in Section 6.2. For a graph family $\mathcal{G}$, let $\mathsf{Clo}(\mathcal{G})$ denote the *deletion closure* of $\mathcal{G}$, i.e., the family of graphs obtained from a graph in $\mathcal{G}$ by deleting edges.

6.1 Lower Bounds for $(1-\varepsilon)$-Approximation Algorithms

In this section, we show the following:

Theorem 6.1. There exist universal constants $\varepsilon, \delta > 0$ such that any $(1-\varepsilon)$-approximation algorithm for the maximum clique problem has sensitivity $\Omega(n^{\delta})$.

For $1 \geq c \geq s \geq 0$, we define $\mathsf{MaxClique}_{c,s}$ as the problem that, given a graph $G = (V, E)$ on n vertices with a clique of size at least cn, the goal is to find a clique of size at least sn. We use the standard reduction from $\mathsf{LabelCover}$ to $\mathsf{MaxClique}$ [FGL+96], which we call FGLSS. Let $I = (U, V, E, \Sigma_U, \Sigma_V, \mathcal{R} = \{R_e\}_{e \in E})$ be a satisfiable label cover instance. We construct a graph $G' = (V', E')$, where $V' = \{v_{e,a,b} : e \in E, (a,b) \in R_e\}$. As for the edges in G', we connect $v_{e_1,a_1,b_1} \in V'$ and $v_{e_2,a_2,b_2} \in V'$ by an edge if (e_1, a_1, b_1) and (e_2, a_2, b_2) are consistent. For each $e \in E$, we define $\mathrm{cloud}(e) = \{v_{e,a,b} : (a,b) \in R_e\}$. We note that $n' := |V'| = |\Sigma_U| m$. As well, it is easy to see that G' has a clique of size m: For any satisfying assignment σ for I, the vertex set $\{v_{e,\sigma(u),\sigma(v)} : e = (u,v) \in E\}$ forms a clique in G'.

Proof of Theorem 6.1. We basically follow the argument in Section 3, but we need to slightly modify it because the target problem is not a CSP.

Let $\varepsilon, \delta > 0$ and $d, k \geq 1$ to be the ones given in Theorem 5.2. Let $\mathcal{I}$ be the family of satisfiable label cover instances with n variables on finite domains Σ_U, Σ_V with $|\Sigma_V| \leq |\Sigma_U| = k$, where each variable is incident to at most d constraints. Note that $\mathsf{Sens}_{1,1-\varepsilon}(\mathcal{I}) = \Omega(n^{\delta})$.

Let $c := 1/k$ and let A' be an algorithm for $\mathsf{MaxClique}_{c,(1-\varepsilon)c}$. Using A', we design an algorithm A for $\mathsf{LabelCover}$. Specifically, given a satisfiable instance $I = (U, V, E, \Sigma_U, \Sigma_V, \mathcal{R}) \in \mathcal{I}$, we first apply FGLSS to obtain a graph $G' = (V', E')$ of n' vertices. Note that G' contains a clique of size $|V'|/|\Sigma_U| = cn'$. Then, we apply the algorithm A' to obtain a clique $\boldsymbol{C}$ of size at least $(1-\varepsilon)cn'$.

Next, we describe a procedure T_σ that recovers an assignment $\boldsymbol{\sigma}$ for I from a clique C for G'. Let $U(C) = \{u \in U : v_{uv,a,b} \in C\}$ and $V(C) = \{v \in V : v_{uv,a,b} \in C\}$. It is easy to observe that for each $u \in U(C)$, there is a value $a_u \in \Sigma_U$ such that, if C has a vertex of the form $v_{e,a,b}$ with u being an endpoint of e, then $a = a_u$. Thus, we can define $\sigma(u) = a_u$. For a variable $u \in U \setminus U(C)$, we set $\sigma(u)$ to be an arbitrary but fixed value in Σ_U. Similarly, for each $v \in V(C)$, there is a value $b_v \in \Sigma_V$ such that, if C has a vertex of the form $v_{e,a,b}$ with v being an endpoint of e, then $b = b_v$. Thus, we can define $\sigma(v) = b_v$. For a variable $v \in V \setminus V(C)$, we set $\sigma(v)$ to be an arbitrary but fixed value in Σ_V.

Now, we show that the expected value of $\boldsymbol{\sigma} := T_\sigma(\boldsymbol{C})$ is at least $1-\varepsilon$. By the observation that $\boldsymbol{C}$ may contain at most a single vertex from the cloud of each $e \in E$, it follows that there are more than $(1-\varepsilon)cn' = (1-\varepsilon)m$ clouds in expectation from which $\boldsymbol{C}$ contains a vertex. Let $\boldsymbol{F} \subseteq E$ be the set of edges $e \in E$ such that $\boldsymbol{C}$ contains some vertex from the cloud of e. We argue that $\boldsymbol{\sigma}$ satisfies all edges in $\boldsymbol{F}$, hence they satisfy at least $|\boldsymbol{F}|$ constraints, which is a $(1-\varepsilon)$-fraction of the constraints in expectation.

Indeed, if $e = (u, v) \in \boldsymbol{F}$ then there is a vertex of the form $v_{e,a,b}$ in $\boldsymbol{C}$. Then, (a, b) satisfies the constraint R_e by construction, and we have $\sigma(u) = a$ and $\sigma(v) = b$ by the choice of $\boldsymbol{\sigma}$.

Next, we discuss sensitivity. Let $I, \tilde{I} \in \mathcal{I}$ be two instances, where $\tilde{I}$ is obtained from I by deleting a constraints. Let $G', \tilde{G}'$ denote the graphs obtained from $I, \tilde{I}$, respectively, by applying FGLSS. Then, it is easy to observe that $|E' \triangle \tilde{E}'| \leq 2d|\Sigma_U|$. Let $\boldsymbol{C}, \tilde{\boldsymbol{C}}$ be the outputs of A' on G' and $\tilde{G}'$, respectively. Then, we have

$$\mathsf{EMD}(\boldsymbol{C}, \tilde{\boldsymbol{C}}) \leq 2dk \cdot \mathsf{Sens}(A', \mathcal{G}'),$$

where $\mathcal{G}' = \mathsf{Clo}(\mathrm{FGLSS}(\mathcal{I}))$. Let $\boldsymbol{\sigma}, \tilde{\boldsymbol{\sigma}}$ be the assignments constructed from $\boldsymbol{C}, \tilde{\boldsymbol{C}}$, respectively, by applying T_σ above. Then, we have

$$\begin{aligned}\mathsf{EMD}(\boldsymbol{\sigma}, \tilde{\boldsymbol{\sigma}}) &= \mathsf{EMD}(U(\boldsymbol{C}), U(\tilde{\boldsymbol{C}})) + \mathsf{EMD}(V(\boldsymbol{C}), V(\tilde{\boldsymbol{C}})) \\ &\leq 2\mathsf{EMD}(\boldsymbol{C}, \tilde{\boldsymbol{C}}) = 4dk \cdot \mathsf{Sens}(A', \mathcal{G}'))\end{aligned}$$

As the choice of the algorithm A' was arbitrary, any algorithm for $\mathsf{MaxClique}_{c,(1-\varepsilon)c}$ must have sensitivity $\Omega(n^\delta/dk) = \Omega((n')^\delta)$. □

For a graph $G = (V, E)$, a vertex set $S \subseteq V$ is called a *vertex cover* if every edge in E is incident to some vertex in S. In the *minimum vertex cover problem*, given a graph $G = (V, E)$, the goal is to find a vertex cover of the minimum size.

Corollary 6.2. There exist universal constants $\varepsilon, \delta > 0$ such that any $(1 + \varepsilon)$-approximation algorithm for the minimum vertex cover problem has sensitivity at least $\Omega(n^\delta)$.

Proof. First, note that the proof of Theorem 6.1 shows that there exists a universal constants $c > 0$ and $\varepsilon > 0$ such that any algorithm for $\mathsf{MaxClique}_{c,(1-\varepsilon)c}$ has sensitivity at least $n^{1-o(1)}$. Also note that a vertex set C is a clique of a graph $G = (V, E)$ if and only if its complement $\bar{C} := V \setminus C$ is a vertex cover of the complement graph $\bar{G} = (V, \binom{V}{2} \setminus E)$.

Let A' be an arbitrary $(1 + \varepsilon c)$-approximation algorithm for the minimum vertex cover problem. Then, we consider an algorithm A for the maximum clique problem that, given a graph $G = (V, E)$, first applies A' to $\bar{G}$ to obtain a vertex cover $S \subseteq V$, and then returns a vertex set $\bar{S}$. When there is a clique of size cn, the algorithm A returns a clique of size at least

$$n - (1 + \varepsilon c)(1 - c)n = -\varepsilon cn + cn + \varepsilon c^2 n \geq (1 - \varepsilon)cn.$$

Hence, A is a $(1 - \varepsilon)$-approximation algorithm and hence its sensitivity must be $\Omega(n^\delta)$.

Moreover deleting an edge from G corresponds to adding an edge to $\bar{G}$, and hence the sensitivity bound applies to A'. □

6.2 Lower Bounds for $n^{-\varepsilon}$-Approximation Algorithms

In this section, we show a sensitivity lower bound for $n^{-\varepsilon}$-approximation algorithms for the maximum clique problem for some constant $\varepsilon > 0$. To this end, we first apply serial repetition to get a lower bound for ε-approximation algorithms for $\mathsf{LabelCover}$ for any $\varepsilon > 0$.

Lemma 6.3. There exists a universal constant $\delta > 0$ such that for any $\varepsilon > 0$, any algorithm for $\mathsf{MaxCSP}_{1,\varepsilon}$ on t-ary instances of degree at most $O(\log n + \varepsilon^{-1} \log \varepsilon^{-1})$ that succeeds with probability at least $1 - O(1/n)$ requires sensitivity $\Omega(n^\delta/(\varepsilon^{-1} \log \varepsilon^{-1}))$, where $t = O(\log \varepsilon^{-1})$.

Let $\varepsilon' > 0$ be an arbitrary parameter and we set

$$t = O\left(\frac{\log(1/\varepsilon')}{\varepsilon}\right) \text{ and } M = \max\left\{\log\frac{3k^n}{\varepsilon}\cdot\frac{(1-\varepsilon/3)^{-t}}{3}, \frac{3m\log(nm)}{t}, \frac{3m\log n^2}{dt}\right\}.$$

We consider a randomized transformation, called SERIALREPETITION, from label cover instances to CSP instances. Specifically, given an instance $I = (U, V, E, \Sigma_U, \Sigma_V, \mathcal{R} = \{R_e\}_{e\in E}) \in \mathcal{I}$, we construct a (random) instance $\boldsymbol{I}' = (U, V, \boldsymbol{E}', \Sigma_U^t, \Sigma_V^t, \boldsymbol{\mathcal{R}}' = \{R'_e\}_{e\in \boldsymbol{E}'})$ as follows: For $i = 1, \ldots, M$, sample t edges $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_t \in E$ uniformly at random, and let $\boldsymbol{e}^{(i)}$ be the multiset $\bigcup_{j=1}^t \boldsymbol{e}_j$. Let $R'_{\boldsymbol{e}^{(i)}}$ be the relation obtained by combining $R_{\boldsymbol{e}_1}, \ldots, R_{\boldsymbol{e}_t}$, i.e.,

$$\boldsymbol{R}'_{\boldsymbol{e}^{(i)}} = \left\{(a_1, \ldots, a_t, b_1, \ldots, b_t) \in \Sigma_U^t \times \Sigma_V^t : (a_j, b_j) \in R_{\boldsymbol{e}_j}\ \forall j \in [t]\right\}.$$

Note that $|\boldsymbol{E}'| = M$ and each relation has arity at most $2t$.

We show several key properties of $\boldsymbol{I}'$.

Claim 6.4. With probability at least $1 - \varepsilon/3$, we have $\mathsf{val}_{\boldsymbol{I}'}(\sigma) < \varepsilon'$ for any assignment σ for $I \in \mathcal{I}$ with $\mathsf{val}_I(\sigma) < 1 - \varepsilon/3$.

Proof. Let σ be an arbitrary assignment that satisfies at most a $(1-\varepsilon/3)$-fraction of the constraints of I. Let $\boldsymbol{X}_i^\sigma$ be the indicator random variable that σ satisfies the ith constraint of $\boldsymbol{I}'$. Then by the Chernoff bound, we have

$$\Pr\left[\sum_{i=1}^M \boldsymbol{X}_i^\sigma \geq 2\left(1-\frac{\varepsilon}{3}\right)^t M\right] \leq \exp\left(-\frac{1}{3}\left(1-\frac{\varepsilon}{3}\right)^t M\right) \leq \frac{\varepsilon}{3k^n},$$

by our choice of M. Finally, by a union bound, with probability at least $1 - \varepsilon/3$ we have $\mathsf{val}_{\boldsymbol{I}'}(\sigma) \leq 2(1-\varepsilon/3)^t < \varepsilon'$ for any σ with $\mathsf{val}_I(\sigma) < 1 - \varepsilon/3$. □

Claim 6.5. With probability at least $1 - 1/n$, every edge $e \in E$ is used to construct at most $2tM/m$ constraints in $\boldsymbol{I}'$.

Proof. For an edge $e \in E$, $i \in [M]$, and $j \in [t]$, let $\boldsymbol{Y}_{i,j}^e$ be the indicator that e is used to construct $\boldsymbol{e}^{(i)}$ as the j-th edge. By Chernoff's bound, we have

$$\Pr\left[\sum_{i=1}^M\sum_{j=1}^t \boldsymbol{Y}_{i,j}^e \geq \frac{2tM}{m}\right] \leq \exp\left(-\frac{tM}{3m}\right) \leq \frac{1}{nm}.$$

By a union bound, with probability at most $1/n$, every edge $e \in E$ is used to construct at most $2tM/m$ constraints in $\boldsymbol{I}'$. □

Claim 6.6. With probability at least $1 - 1/n$, the instance $\boldsymbol{I}'$ has degree at most dtM/m.

Proof. For a variable $v \in U \cup V$, $i \in [M]$, and $j \in [M]$, let $\boldsymbol{Y}_{i,j}^v$ be the indicator that an edge incident to v is used to construct $\boldsymbol{e}^{(i)}$ as the j-th edge. Note that $\mathbf{E}\,\boldsymbol{Y}_{i,j}^v \leq d/2m$. Hence by Chernoff's bound, we have

$$\Pr\left[\sum_{i=1}^M\sum_{j=1}^t \boldsymbol{Y}_{i,j}^v \geq \frac{dtM}{m}\right] \leq \exp\left(-\frac{dtM}{3m}\right) \leq \frac{1}{n^2}.$$

By a union bound, with probability at most $1/n$, every vertex $v \in U \cup V$ is incident to at most dtM/m constraints in $\boldsymbol{I}'$. □

Proof of Lemma 6.3. Let $\varepsilon, \delta > 0$ and $d, k \geq 1$ to be the ones given in Theorem 5.2. Let $\mathcal{I}$ be the swap-closed family of satisfiable label cover instances with m edges and n variables on finite domains Σ_U, Σ_V with $|\Sigma_V| \leq |\Sigma_U| = k$, where each variable is incident to at most d other variables. Note that we have $\mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I}) = \Omega(n^\delta)$ (see the proof of Theorem 5.2).

Let $\mathcal{I}'$ be the swap closure of instances in the support of $\textsc{SerialRepetition}(\mathcal{I})$ such that the degree is at most dtM/m. Note that $\boldsymbol{I}'$ belongs to $\mathcal{I}'$ with probability at least $1 - 1/n$ by Claim 6.6. Let A' be an arbitrary algorithm for $\mathcal{I}'$ that attains $\mathsf{SwapSens}_{1,\varepsilon'}(\mathcal{I}')$. We design an algorithm A for $\mathcal{I}$ which operates as follows: Given an instance of $I \in \mathcal{I}$, let $\boldsymbol{I}' = \textsc{SerialRepetition}(I)$. Then, we apply A' on $\boldsymbol{I}'$ to compute an assignment $\boldsymbol{\sigma}'$, and simply output $\boldsymbol{\sigma} := \boldsymbol{\sigma}'$.

We first show that A is an algorithm for $\mathsf{LabelCover}_{1,1-\varepsilon}$ on $\mathcal{I}$. Recall that $\mathsf{val}_{I'}(A'(I')) \geq \varepsilon'$ with probability at least $1 - O(1/n)$ for any $I' \in \mathcal{I}'$. Let $\boldsymbol{X}_{I'}$ be the indicator of the event that this happens. Let $\boldsymbol{X}_{6.4}$ and $\boldsymbol{X}_{6.6}$ be the indicators of the events that the events of Claim 6.4 and Claim 6.6 hold, respectively. Then, we have

$$\begin{aligned}
&\mathbf{E}\,\mathsf{val}_I(A(I)) \geq \Pr[\boldsymbol{X}_{I'} \wedge \boldsymbol{X}_{6.4} \wedge \boldsymbol{X}_{6.6}] \cdot \mathbf{E}[\mathsf{val}_I(A(I)) \mid \boldsymbol{X}_{I'} \wedge \boldsymbol{X}_{6.4} \wedge \boldsymbol{X}_{6.6}] \\
&\geq \left(1 - \frac{\varepsilon}{3} - \frac{1}{n} - \frac{1}{n}\right) \mathbf{E}[\mathsf{val}_I(A(I)) \mid \boldsymbol{X}_{I'} \wedge \boldsymbol{X}_{6.4} \wedge \boldsymbol{X}_{6.6}] && \text{(by Claim 6.4 and Claim 6.6)} \\
&\geq \left(1 - \frac{\varepsilon}{3} - \frac{1}{n} - \frac{1}{n}\right)\left(1 - \frac{\varepsilon}{3}\right) && \text{(by } \mathsf{val}_{I'}(A'(I')) \geq \varepsilon' \text{ for } I' \in \mathcal{I}') \\
&\geq 1 - \varepsilon.
\end{aligned}$$

Next, we analyze the swap sensitivity of A. Let $I, \tilde{I} \in \mathcal{I}$ be two instances with swap distance one. We consider a natural coupling between $\boldsymbol{I}'$ and $\tilde{\boldsymbol{I}}'$. Note that whenever $\boldsymbol{I}'$ satisfies the claims in Claim 6.5 and Claim 6.6, so does $\tilde{\boldsymbol{I}}'$. Let $\boldsymbol{X}_{6.5}$ be the indicator of the event that the claim of Claim 6.5 holds. Then, the swap sensitivity of A is bounded from above by

$$\begin{aligned}
&\Pr[\boldsymbol{X}_{6.5} \wedge \boldsymbol{X}_{6.6}] \cdot \frac{2tM}{m} \cdot \mathsf{SwapSens}(A', \mathcal{I}') + \Pr\left[\bar{\boldsymbol{X}}_{6.5} \vee \bar{\boldsymbol{X}}_{6.6}\right] \cdot n \\
&\leq \left(1 - \frac{2}{n}\right) \cdot \frac{2tM}{m} \cdot \mathsf{SwapSens}(A', \mathcal{I}') + \frac{2}{n} \cdot n \\
&\leq O_{\varepsilon,d,k}\left(\max\left\{\frac{\log(1/\varepsilon')}{\varepsilon'}, \log n\right\}\right) \cdot \mathsf{SwapSens}_{1,\varepsilon}(\mathcal{I}').
\end{aligned}$$

Hence, we have

$$\mathsf{SwapSens}_{1,\varepsilon'}(\mathcal{I}') = \Omega_{\varepsilon,d,k}\left(\frac{\mathsf{SwapSens}_{1,1-\varepsilon}(\mathcal{I})}{\max\left\{\frac{\log(1/\varepsilon')}{\varepsilon'}, \log n\right\}}\right) = \Omega_{\varepsilon,d,k}\left(\frac{n^\delta}{\log n + \frac{\log(1/\varepsilon')}{\varepsilon'}}\right) \qquad \square$$

We obtain the claimed lower bound for swap sensitivity by replacing ε' with ε and slightly decreasing δ. Finally, Lemma 2.3 gives a lower bound on sensitivity.

Theorem 6.7. There exist universal constants $\varepsilon, \delta > 0$ such that any algorithm for the maximum clique problem that outputs an $n^{-\varepsilon}$-approximate clique with probability $1 - O(1/n)$ has sensitivity $\Omega(n^\delta)$.

Proof. Let $\varepsilon > 0$ be a parameter which will be determined later. Let $\delta > 0$ and $t \geq 1$ be the ones given in Lemma 6.3 with ε in the statement being replaced with $n^{-\varepsilon}$; in particular, $t = O(\varepsilon \log n)$. Let $\mathcal{I}$ be the set

of satisfiable label cover instances with at most n variables, each relation having arity at most t, and each variable being incident to at most $d = O(n^{\varepsilon})$ constraints, given by Lemma 6.3. Note that any algorithm for $\mathcal{I}$ that outputs $n^{-\varepsilon}$-approximate clique with probability $1 - O(1/n)$ has sensitivity $\Omega(n^{\delta})$.

We consider a slight modification of the FGLSS reduction described in Section 6.1. Given a CSP instance $I = (V, E, \Sigma, \mathcal{R} = \{R_e\}_{e \in E}) \in \mathcal{I}$, we construct a graph $G' = (V', E')$, where $V' = E \times \Sigma^t$, as follows: For each hyperedge $e \in E$ and every assignment $\alpha \in \Sigma^t$, add a node $v_{e,\alpha}$ to V'. Then, we connect v_{e_1,α_1} and v_{e_2,α_2} if $\alpha_1 \in R_{e_1}$, $\alpha_2 \in R_{e_2}$, and α_1 and α_2 are consistent with all the variables shared between e_1 and e_2. Note that $n' := |V'| = m|\Sigma|^t = mn^{O(\varepsilon \log |\Sigma|)}$ and that G' has a clique of size m.

Let A' be an $n^{-\varepsilon}$-approximation algorithm for the maximum clique problem. Given an instance $I \in \mathcal{I}$, we first construct a graph G' as above, and then compute a clique $\boldsymbol{C}$ in G' using A'. We construct an assignment $\boldsymbol{\sigma}$ for I as follows. For every variable $v_{e,\alpha}$ in $\boldsymbol{C}$, we set $\boldsymbol{\sigma}(u) = \alpha(u)$ for every $u \in e$. By the construction of the graph, $\boldsymbol{\sigma}$ is well defined. For $u \in V$ with $\boldsymbol{\sigma}(u)$ undefined, we set $\boldsymbol{\sigma}(u)$ to be an arbitrarily but fixed label.

We note that $\boldsymbol{\sigma}$ satisfies all the constraints corresponding to edges $e \in E'$ such that $v_{e,\alpha}$ is included in the clique $\boldsymbol{C}$ for some $\alpha \in R_e$. This implies that the assignment $\boldsymbol{\sigma}$ satisfies $|\boldsymbol{C}| \geq m/n^{\varepsilon}$ constraints with probability $1 - O(1/n)$, and hence A outputs $n^{-\varepsilon}$-approximate clique with probability $1 - O(1/n)$.

Next, we analyze the sensitivity of A. When we delete a constraint in I, we change at most $d2^t = O(n^{2\varepsilon})$ edges in G'. Hence, the sensitivity of A is at most $O(n^{2\varepsilon})$ times the sensitivity of A'. By choosing $\varepsilon \ll \delta$, the sensitivity of A' is

$$\Omega\left(\frac{n^{\delta}}{n^{2\varepsilon}}\right) = \Omega\left(n^{\delta/2}\right) = \Omega\left((n')^{\delta/O(\varepsilon \log |\Sigma|+1)}\right),$$

which implies the claim. □

A vertex set $S \subseteq V$ is called an *independent set* if no two vertices $u, v \in S$ have an edge between them. In the *maximum independent set problem*, given a graph $G = (V, E)$, the goal is to find an independent set of maximum size. Because a vertex set is an independent set if and only it is a clique in the complement graph, we obtain the following:

Corollary 6.8. There exist universal constants $\varepsilon, \delta > 0$ such that any algorithm for the maximum independent set that outputs an $n^{-\varepsilon}$-approximate solution with probability $1 - O(1/n)$ has sensitivity $\Omega(n^{\delta})$.

7 Max CSPs

In this seciton, we show lower bounds for various Max CSPs. First, we consider E3SAT, which is a special case of Boolean CSPs, where each constraint is a disjunctions of exactly 3 literals.

Theorem 7.1. There exist universal constants $\varepsilon, \delta > 0$ such that any algorithm for $\mathsf{E3SAT}_{1,1-\varepsilon}$ has sensitivity $\Omega(n^{\delta})$.

Proof. Let $\varepsilon, \delta > 0$ and $d, k \geq 1$ be as in Theorem 5.2. That is, any algorithm for $\mathsf{LabelCover}_{1,1-\varepsilon}$ over a bipartite graph of maximum degree d and over a domain of size k has sensitivity of $\Omega(n^{\delta})$.

We consider the following transformation T_I from label cover instances to SAT instances. Let $I = (U, V, E, \Sigma_U, \Sigma_V, \mathcal{R})$ be a satisfiable label cover instance with $|\Sigma_V| \leq |\Sigma_U| = k$. We encode each label using $O(\log k)$ bits and introduce corresponding Boolean variables. As well, we introduce a constraint simulating R_e for each $e \in E$ over the corresponding Boolean variables. Let the Boolean CSP instance that results from this procedure by denoted by $\hat{I} = (\hat{V}, \hat{E}, \{0, 1\}, \hat{\mathcal{R}})$, and note that each hyperedge $e \in \hat{E}$

is of size at most $O(\log k)$. Then, we further transform it to a SAT instance $I' = (V', E', \{0,1\}, \mathcal{R}')$ by converting each constraint in $\hat{\mathcal{R}}$ to an E3CNF formula. Note that we may need to add some auxiliary variables in this step and hence $\hat{V}$ is a subset of V'. Without loss of generality we may assume that each E3CNF formula has exactly K clauses, where $K = O(\log k) \cdot 2^{O(\log k)} = O(k \log k)$.

Suppose $\sigma' : V' \to \{0,1\}$ is an assignment for I'. Then, there is a natural transformation T_σ that decodes an assignment σ for I from σ'. We now show that the pair (T_I, T_σ) is a $(1, 1-\varepsilon, c' = 1, s' = 1 - \varepsilon/K, C_I = K, C_\sigma = 1)$-sensitivity-preserving reduction. Then, the claim follows by Lemma 3.2. The analysis for c' and C_σ is immediate.

First, we analyze s'. Suppose that a (possibly random) assignment $\boldsymbol{\sigma}' : V' \to \{0,1\}$ satisfies $\mathbf{E}\,\mathsf{val}_{I'}(\boldsymbol{\sigma}') \geq 1 - \varepsilon'$. This implies that $\boldsymbol{\sigma}'$ violates at most $\varepsilon' m'$ constraints in I' in expectation, and hence $\boldsymbol{\sigma}'$ (restricted to $\hat{V}$) violates at most $\varepsilon' K \hat{m}$ constraints in $\hat{I}$ in expectation. It follows that $\boldsymbol{\sigma}$ violates at most $\varepsilon' K \hat{m} = \varepsilon' K m$ constraints, and hence $\mathbf{E}\,\mathsf{val}_I(\boldsymbol{\sigma}) \geq 1 - \varepsilon' K$. Then, $s' = 1 - \varepsilon/K$ satisfies Item 2 of Definition 3.1.

Next, we analyze C_I. Let I and $\tilde{I}$ be two instances of $\mathsf{LabelCover}_{1,1-\varepsilon}$, where $\tilde{I}$ is obtained from I by deleting one constraint. Then, I' and $\tilde{I}'$ differ by at most K constraints, and hence we can set $C_I = K = O(k \log k)$. □

Next, we consider 3LIN, which is a special type of a Boolean CSP, where we identify the domain $\{0,1\}$ with the group $\mathbb{Z}_2$ and each constraint is of the form $x = 0$, $x = 1$, $x + y = 0$, $x + y = 1$, $x + y + z = 0$, or $x + y + z = 1$. For $1 \geq c \geq s \geq 0$, we define $\mathsf{Max3LIN}_{c,s}$ denote the problem that, given a c-satisfiable instance of 3LIN, the goal is to compute an s-satisfying assignment.

Corollary 7.2. There exist universal constants $\varepsilon, \delta > 0$ such that any algorithm for $\mathsf{Max3LIN}_{4/7,(1-\varepsilon)4/7}$ has sensitivity $\Omega(n^\delta)$.

Proof. Consider the following transformation T_I from an instance $I = (V, E, \{0,1\}, \mathcal{R})$ of E3SAT to an instance $I' = (V', E', \mathbb{Z}_2, \mathcal{R}')$ of 3LIN: For each constraint of the form $(l_1 \vee l_2 \vee l_3)$, where l_1, l_2, l_3 are literals, we introduce seven constraints of the form $l_1 = 1$, $l_2 = 1$, $l_3 = 1$, $l_1 + l_2 = 1$, $l_2 + l_3 = 1$, $l_1 + l_3 = 1$, and $l_1 + l_2 + l_3 = 1$, where we identify a negative literal $\bar{x}$ with $1 + x \pmod 2$. In particular, $|V| = |V'|$ holds.

Given an assignment $\sigma' : V' \to \mathbb{Z}_2$ for I', we simply output the corresponding assignment $\sigma : V \to \{0,1\}$. It is easy to confirm that the pair (T_I, T_σ) is a $(1, 1-\varepsilon, 4/7, (1-\varepsilon)\cdot 4/7, 7, 1)$-sensitivity-preserving reduction. Hence, the claim follows by Lemma 3.2. □

Finally, we consider the maximum cut problem, where we are given a graph $G = (V, E)$, and the goal is to find a set $S \subseteq V$ that maximizes the cut size, i.e., the number of edges between S and $V \setminus S$. Using the reduction in [TSSW00], we obtain the following sensitivity lower bound for the maximum cut problem.

Corollary 7.3. There exist universal constants $c, \varepsilon, \delta > 0$ such that any algorithm for $\mathsf{MaxCut}_{c,c(1-\varepsilon)}$ has sensitivity $\Omega(n^\delta)$.

8 Distributed Algorithms

We consider the LOCAL model of distributed computing [Lin92], where a network is modeled as a graph $G = (V, E)$ in such a way that each vertex $v \in V$ corresponds to an agent and each edge $e \in E$ corresponds to a communication link. The communication proceeds in synchronous rounds. In each round, each vertex $v \in V$ receives the messages sent from its neighbors, performs some arbitrary local computation, and sends a message of arbitrary size to each of its neighbors. We also assume that the vertices are anonymous, —-

i.e., they do not have identifiers — and that they each have access to an infinite string of local random bits. The goal of a distributed algorithm is for the outputs of the vertices to form a feasible solution to the problem that they are trying to solve. The *round complexity* of a distributed algorithm is the number of rounds until it terminates. The following observation shows that if there is a distributed algorithm with small round complexity, then we can use it to design an algorithm with low sensitivity.

Theorem 8.1. Let $\mathcal{P}$ be a graph problem, where the output is a vertex set. If there is a distributed algorithm for $\mathcal{P}$ in the LOCAL model that runs in t rounds, then there exists an algorithm for $\mathcal{P}$ with sensitivity $O(n^t)$.

Proof. Let A' be a (possibly randomized) distributed algorithm for $\mathcal{P}$ with round complexity t. Then, we consider the following (centralized) algorithm A. Given a graph $G = (V, E)$, for every vertex $v \in V$, we simulate the process of A' on v for t rounds, and then we collect the outputs of vertices and output them as a solution. Clearly, the output of A has the same distribution as that of A'.

We now analyze the sensitivity of A. Let $G = (V, E)$ be a graph and consider $\tilde{G} = G - e$ for some $e \in E$. For a vertex $v \in V$, let π_v denote the internal randomness of the distributed algorithm A' on v. For $\pi = \{\pi_v\}_{v \in V}$, let A'_π denote the deterministic distributed algorithm such that the internal randomness of $v \in V$ is fixed to π_v.

Now, we consider bounding the expected Hamming distance $\mathbf{E}_\pi \, \mathsf{Ham}(A'_\pi(G), A'_\pi(\tilde{G}))$. Suppose a vertex $v \in V$ is in the outputted vertex set of exactly one of $A'_\pi(G)$ and $A'_\pi(\tilde{G})$. Because we use the same internal randomness between G and $\tilde{G}$, the edge e must belong to the t-hop neighborhood of v. However, the number of vertices $v \in V$ such that e belongs to the t-hop neighborhood is bounded by n^t, and hence we have $\mathsf{Ham}(A'_\pi(G), A'_\pi(\tilde{G})) \le n^t$. Hence, we have

$$\mathsf{EMD}(A(G), A(\tilde{G})) = \mathsf{EMD}(A'(G), A'(\tilde{G})) \le \mathop{\mathbf{E}}_{\pi} \mathsf{Ham}(A'_\pi(G), A'_\pi(\tilde{G})) \le n^t. \qquad \square$$

Corollary 8.2. Let $\mathcal{P}$ be a graph problem, where the output is a vertex set. If any algorithm for $\mathcal{P}$ on graphs with maximum degree at most Δ has sensitivity at least $f(n)$, then any distributed algorithm for $\mathcal{P}$ must have round complexity $\log_\Delta f(n)$.

Using Corollary 8.2, we can recover various known lower bounds for the LOCAL model.

Maximum independent set. Corollary 6.8 states that there exist constants $\varepsilon, \delta > 0$ such that any algorithm for the maximum independent set problem on graphs with maximum degree $n^{O(\varepsilon)}$ that outputs an $n^{-\varepsilon}$-approximate solution with probability $1 - O(1/n)$ has sensitivity $\Omega(n^\delta)$. Hence by Corollary 8.2, any distributed algorithm for the maximum independent set problem that outputs an $n^{-\varepsilon}$-approximation solution with probability $1 - O(1/n)$ has round complexity

$$\Omega\left(\log_{n^{O(\varepsilon)}} n^\delta\right) = \Omega\left(\frac{1}{\varepsilon}\right).$$

This matches the lower bound of [BHKK16].

Minimum vertex cover. Corollary 6.2 states that there exist $\varepsilon, \delta > 0$ such that any $(1+\varepsilon)$-approximation algorithm for the minimum vertex cover problem on bounded-degree graphs has sensitivity $\Omega(n^\delta)$. Hence by Corollary 8.2, the round complexity of a $(1+\varepsilon)$-approximation algorithm for the minimum vertex cover problem must be $\Omega(\log n^\delta) = \Omega(\log n)$. This matches the known lower bound of [GS14, FFK22].

Maximum cut. Corollary 7.3 states that there exist $\varepsilon, \delta > 0$ such that any $(1+\varepsilon)$-approximation algorithm for the maximum cut problem on bounded-degree graphs has sensitivity $\Omega(n^\delta)$. Hence by Corollary 8.2,

the round complexity of a $(1-\varepsilon)$-approximation algorithm for the maximum cut problem must be $\Omega(\log n^\delta) = \Omega(\log n)$. This matches the known lower bound of [CL23].

Next, we consider distributed constraint satisfaction problems (DCSPs) [YIDK92]. In DCSPs, each variable and constraint is associated with an agent. The agent for a variable can communicate with the agent for any constraint involving that variable, and vice versa. The goal of a distributed algorithm is for the assignment produced by the variables to maximize the number of satisfied constraints.

Given a CSP instance $G = (V, E, \Sigma, \mathcal{R})$, its *constraint graph* is a bipartite graph on the vertex set $V \cup E$ where $v \in V$ and $e \in E$ is connected if $v \in e$. We can specialize Corollary 8.2 for DCSPs:

Corollary 8.3. Let $\mathcal{I}$ be a family of CSP instances. If any algorithm for $\mathcal{I}$ on constraint graphs with maximum degree at most Δ has sensitivity at least $f(n)$, then any distributed algorithm for $\mathcal{P}$ must have round complexity $\log_\Delta f(n)$.

Proof of Theorem 1.4. Lemma 6.3 claims that for any $\varepsilon > 0$, there exists a universal constant $\delta > 0$ such that any algorithm for $\mathsf{MaxCSP}_{1,\varepsilon}$ on t-ary instances of degree at most $O(\log n + \varepsilon^{-1}\log\varepsilon^{-1})$ with success probability $1 - O(1/n)$ has sensitivity $\Omega(n^\delta/(\varepsilon^{-1}\log\varepsilon^{-1}))$, where $t = O(\log\varepsilon^{-1})$. Hence, by Corollary 8.3, any distributed algorithms for such CSP instances must have round complexity

$$\Omega\left(\log_{\log n+\varepsilon^{-1}\log\varepsilon^{-1}} \frac{n^\delta}{\varepsilon^{-1}\log\varepsilon^{-1}}\right) = \Omega\left(\frac{\log n^\delta - \varepsilon^{-1}\log\varepsilon^{-1}}{\log(\log n + \varepsilon^{-1}\log\varepsilon^{-1})}\right).$$

By replacing ϵ with $1/\log n^\epsilon$ for $\epsilon \ll \delta$, we obtain that any algorithm that outputs $1/\log n^\epsilon$-approximate solution with probability $1 - O(1/n)$ must have round complexity $\Omega(\log n/\log\log n)$. □

References

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.

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

[BD24] Silvia Butti and Victor Dalmau. The complexity of the distributed constraint satisfaction problem. *Theory of Computing Systems*, 68(4):838–867, 2024.

[BHKK16] Marijke HL Bodlaender, Magnús M Halldórsson, Christian Konrad, and Fabian Kuhn. Brief announcement: Local independent set approximation. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC)*, pages 93–95, 2016.

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993.

[BMV24] Mitali Bafna, Dor Minzer, and Nikhil Vyas. Quasi-linear size PCPs with small soundness from hdx. *arXiv preprint arXiv:2407.12762*, 2024.

[BSGH+04] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust PCPs of proximity, shorter PCPs and applications to coding. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–10, 2004.

[CAFL+22] Vincent Cohen-Addad, Chenglin Fan, Silvio Lattanzi, Slobodan Mitrovic, Ashkan Norouzi-Fard, Nikos Parotsidis, and Jakub M Tarnawski. Near-optimal correlation clustering with privacy. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:33702–33715, 2022.

[CHHK16] Keren Censor-Hillel, Elad Haramaty, and Zohar Karnin. Optimal dynamic distributed MIS. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC)*, pages 217–226, 2016.

[CL23] Yi-Jun Chang and Zeyong Li. The complexity of distributed approximation of packing and covering integer linear programs. In *Proceedings of the 2023 ACM Symposium on Principles of Distributed Computing (PODC)*, pages 32–43, 2023.

[DHP+12] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science (ITCS)*, pages 214–226, 2012.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3):12–es, 2007.

[DLR+22] Laxman Dhulipala, Quanquan C Liu, Sofya Raskhodnikova, Jessica Shi, Julian Shun, and Shangdi Yu. Differential privacy from locally adjustable graph algorithms: k-core decomposition, low out-degree ordering, and densest subgraphs. In *Proceedings of the IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 754–765, 2022.

[DS14] Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 624–633, 2014.

[Dwo06] Cynthia Dwork. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 1–12, 2006.

[DY24] Jing Dong and Yuichi Yoshida. A batch-to-online transformation under random-order model. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024.

[EKKL20] Marek Eliáš, Michael Kapralov, Janardhan Kulkarni, and Yin Tat Lee. Differentially private release of synthetic graphs. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 560–578, 2020.

[Fei98] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[FFK22] Salwa Faour, Marc Fuchs, and Fabian Kuhn. Distributed CONGEST approximation of weighted vertex covers and matchings. In *25th International Conference on Principles of Distributed Systems (OPODIS)*, 2022.

[FGL+96] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996.

[FPY18] Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research*, 61:623–698, 2018.

[GLM+10] Anupam Gupta, Katrina Ligett, Frank McSherry, Aaron Roth, and Kunal Talwar. Differentially private combinatorial optimization. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1106–1125, 2010.

[GS14] Mika Göös and Jukka Suomela. No sublogarithmic-time approximation scheme for bipartite vertex cover. *Distributed Computing*, 27:435–443, 2014.

[Hås01] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

[HLW06] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

[HY23] Satoshi Hara and Yuichi Yoshida. Average sensitivity of decision tree learning. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

[KHMT20] Jian Kang, Jingrui He, Ross Maciejewski, and Hanghang Tong. InFoRM: Individual fairness on graph mining. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 379–389, 2020.

[Kho02] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 767–775, 2002.

[KY22a] Soh Kumabe and Yuichi Yoshida. Average sensitivity of dynamic programming. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1925–1961, 2022.

[KY22b] Soh Kumabe and Yuichi Yoshida. Average sensitivity of the knapsack problem. In *30th Annual European Symposium on Algorithms (ESA)*, pages 75:1–75:14, 2022.

[KY23] Soh Kumabe and Yuichi Yoshida. Lipschitz continuous algorithms for graph problems. In *Proceedings of the IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 762–797, 2023.

[Lin92] Nathan Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992.

[Mos12] Dana Moshkovitz. The projection games conjecture and the NP-hardness of $\ln n$-approximating set-cover. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 276–287. Springer, 2012.

[MY19] Shogo Murai and Yuichi Yoshida. Sensitivity analysis of centralities on unweighted networks. In *The World Wide Web Conference (WWW)*, pages 1332–1342, 2019.

[PY20] Pan Peng and Yuichi Yoshida. Average sensitivity of spectral clustering. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 1132–1140, 2020.

[Raz95] Ran Raz. A parallel repetition theorem. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing (STOC)*, pages 447–456, 1995.

[Sch78] Thomas J Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–226, 1978.

[SMS18] Khot Subhash, Dor Minzer, and Muli Safra. Pseudorandom sets in Grassmann graph have near-perfect expansion. In *Proceedings of the IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 592–601, 2018.

[TSSW00] Luca Trevisan, Gregory B Sorkin, Madhu Sudan, and David P Williamson. Gadgets, approximation, and linear programming. *SIAM Journal on Computing*, 29(6):2074–2097, 2000.

[VY23] Nithin Varma and Yuichi Yoshida. Average sensitivity of graph algorithms. *SIAM Journal on Computing*, 52(4):1039–1081, 2023.

[YH00] Makoto Yokoo and Katsutoshi Hirayama. Algorithms for distributed constraint satisfaction: A review. *Autonomous Agents and Multi-Agent Systems*, 3:185–207, 2000.

[YI22] Yuichi Yoshida and Shinji Ito. Average sensitivity of Euclidean k-clustering. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:32487–32498, 2022.

[YIDK92] Makoto Yokoo, Toru Ishida, Edmund H Durfee, and Kazuhiro Kuwabara. Distributed constraint satisfaction for formalizing distributed problem solving. In *Proceedings of the 12th International Conference on Distributed Computing System (ICDCS)*, pages 614–615, 1992.

[YZ21] Yuichi Yoshida and Samson Zhou. Sensitivity analysis of the maximum matching problem. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2021.

[YZ24] Yuichi Yoshida and Zihan Zhang. Personal communication, 2024.