Gap Preserving Reductions Between Reconfiguration Problems

Naoto Ohsaka
CyberAgent, Inc., Tokyo, Japan

Abstract

Combinatorial reconfiguration is a growing research field studying problems on the transformability between a pair of solutions for a search problem. For example, in SAT Reconfiguration, for a Boolean formula $\varphi$ and two satisfying truth assignments $\sigma_s$ and $\sigma_t$ for $\varphi$, we are asked to determine whether there is a sequence of satisfying truth assignments for $\varphi$ starting from $\sigma_s$ and ending with $\sigma_t$, each resulting from the previous one by flipping a single variable assignment. We consider the approximability of optimization variants of reconfiguration problems; e.g., Maxmin SAT Reconfiguration requires to maximize the minimum fraction of satisfied clauses of $\varphi$ during transformation from $\sigma_s$ to $\sigma_t$. Solving such optimization variants approximately, we may be able to obtain a reasonable reconfiguration sequence comprising almost-satisfying truth assignments.

In this study, we prove a series of gap-preserving reductions to give evidence that a host of reconfiguration problems are PSPACE-hard to approximate, under some plausible assumption. Our starting point is a new working hypothesis called the Reconfiguration Inapproximability Hypothesis (RIH), which asserts that a gap version of Maxmin CSP Reconfiguration is PSPACE-hard. This hypothesis may be thought of as a reconfiguration analogue of the PCP theorem [3,4]. Our main result is PSPACE-hardness of approximating Maxmin 3-SAT Reconfiguration of bounded occurrence under RIH. The crux of its proof is a gap-preserving reduction from Maxmin Binary CSP Reconfiguration to itself of bounded degree. Because a simple application of the degree reduction technique using expander graphs due to Papadimitriou and Yannakakis (J. Comput. Syst. Sci., 1991) [40] does not preserve the perfect completeness, we modify the alphabet as if each vertex could take a pair of values simultaneously. To accomplish the soundness requirement, we further apply an explicit family of near-Ramanujan graphs and the expander mixing lemma. As an application of the main result, we demonstrate that under RIH, optimization variants of popular reconfiguration problems are PSPACE-hard to approximate, including Nondeterministic Constraint Logic due to Hearn and Demaine (Theor. Comput. Sci., 2005) [24,25], Independent Set Reconfiguration, Clique Reconfiguration, and Vertex Cover Reconfiguration.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases combinatorial reconfiguration, hardness of approximation, gap reduction

Digital Object Identifier 10.4230/LIPIcs.STACS.2023.49

Related Version Full Version: https://arxiv.org/abs/2212.04207 [38]

Acknowledgements I wish to thank the anonymous referees for their suggestions which help improve the presentation of this paper.

1 Introduction

Combinatorial reconfiguration is a growing research field studying the following problem over the solution space: Given a pair of feasible solutions for a particular search problem, find a step-by-step transformation from one to the other, called a reconfiguration sequence.

Since the establishment of the unified framework of reconfiguration due to Ito, Demaine, Harvey, Papadimitriou, Sideri, Uehara, and Uno [28], numerous reconfiguration problems have been derived from search problems; e.g., in SAT Reconfiguration [23], for a Boolean formula $\varphi$ and two satisfying truth assignments $\sigma_s$ and $\sigma_t$ for $\varphi$, we seek a reconfiguration

© Naoto Ohsaka; licensed under Creative Commons License CC-BY 4.0
40th International Symposium on Theoretical Aspects of Computer Science (STACS 2024).
Editors: Petra Berenbrink, Patricia Bouyer, Anuj Dawar, and Mamadou Moustapha Kanté;
Article No. 49; pp. 49:1-49:18
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
sequence of satisfying truth assignments from $\sigma_s$ to $\sigma_t$, each resulting from the previous one by flipping a single variable assignment. Of particular importance is to reveal their computational complexity. Most reconfiguration problems are classified as either P (e.g., 3-Coloring Reconfiguration [14]), NP-complete, or PSPACE-complete (e.g., Independent Set Reconfiguration [24]), and recent studies dig into the fine-grained analysis using restricted graph classes and parameterized complexity [20,21]. See surveys by van den Heuvel [41] and Nishimura [37]. One promising aspect has, however, been still less explored: approximability.

Just like an NP optimization problem derived from an NP search problem (e.g., Max SAT is a generalization of SAT), an optimization variant can be defined for a reconfiguration problem. For instance, in Maxmin SAT Reconfiguration [28] – an optimization variant of SAT Reconfiguration – we wish to maximize the minimum fraction of clauses of $\varphi$ satisfied by any truth assignment during transformation from $\sigma_s$ to $\sigma_t$. Such optimization variants naturally arise when we are faced with the nonexistence of a reconfiguration sequence for the original decision version, or when we already know a problem of interest to be PSPACE-complete. Solving them approximately, we may be able to obtain a reasonable reconfiguration sequence, e.g., that comprising almost-satisfying truth assignments, each violating at most 1% of the clauses.

Indeed, in their seminal work, Ito et al. [28] proved inapproximability results of Maxmin SAT Reconfiguration and Maxmin Clique Reconfiguration, and posed PSPACE-hardness of approximation as an open problem. Their results rely on NP-hardness of the corresponding search problem, which, however, does not bring us PSPACE-hardness. The significance of showing PSPACE-hardness is that it not only refutes a polynomial-time algorithm under P $\neq$ PSPACE, but further disproves the existence of a witness (especially a reconfiguration sequence) of polynomial length under NP $\neq$ PSPACE. The present study aims to reboot the study on PSPACE-hardness of approximation for reconfiguration problems, assuming some plausible hypothesis.

Our Approach. Since no PSPACE-hardness of approximation for natural reconfiguration problems are known (to the best of our knowledge), we assert a new working hypothesis called the Reconfiguration Inapproximability Hypothesis (RIH), concerning a gap version of Maxmin $q$-CSP Reconfiguration, and use it as a starting point.

▶ Hypothesis 1.1 (informal; see Hypothesis 2.4). Given a constraint graph $G$ and two satisfying assignments $\psi_s$ and $\psi_t$ for $G$, it is PSPACE-hard to distinguish between YES instances, in which $\psi_s$ can be transformed into $\psi_t$ by repeatedly changing the value of a single vertex at a time, while ensuring every intermediate assignment satisfying $G$, and NO instances, in which any such transformation induces an assignment violating $\varepsilon$-fraction of the constraints.

This hypothesis may be thought of as a reconfiguration analogue of the PCP theorem [3,4], and it already holds as long as “PSPACE-hard” is replaced by “NP-hard” [28]. Moreover, if the gap version of some reconfiguration problem, e.g., Maxmin SAT Reconfiguration, is PSPACE-hard, RIH directly follows. Our contribution is to prove that the converse is also true: Starting from RIH, we present a series of (polynomial-time) gap-preserving reductions to give evidence that a host of reconfiguration problems are PSPACE-hard to approximate.

Our Results. Figure 1 presents an overall picture of the gap-preserving reductions introduced in this paper, all of which preserve the perfect completeness; i.e., YES instances have a solution to the decision version. Our main result is PSPACE-hardness of approximating Maxmin E3-SAT Reconfiguration of bounded occurrence under RIH (Theorem 3.1). Here, “bounded
occurrence” is critical to further reduce to Nondeterministic Constraint Logic, which requires the number of clauses to be proportional to the number of variables. Toward that end, we first reduce from Maxmin \( q \)-CSP Reconfiguration to Maxmin Binary CSP Reconfiguration in a gap-preserving manner via Maxmin \( E_3 \)-SAT Reconfiguration (Lemmas 3.2 and 3.4), which employs a reconfigurable SAT encoding.

We then proceed to a gap-preserving reduction from Maxmin Binary CSP Reconfiguration to itself of bounded degree (Lemma 3.5), which is the most technical step in this paper. Recall shortly the degree reduction technique due to Papadimitriou and Yannakakis [40], also used by Dinur [19] to prove the PCP theorem [3,4]: Each (high-degree) vertex is replaced by an expander graph called a cloud, and equality constraints are imposed on the intra-cloud edges so that the assignments in the cloud behave like a single assignment. Observe easily that a simple application of this technique to Binary CSP Reconfiguration fails to preserve the perfect completeness. This is because we have to change the value of the vertices in the cloud one by one, breaking many equality constraints. To bypass this issue, we modify the alphabet as if each vertex could take a pair of values simultaneously; e.g., if the original alphabet is \( \Sigma = \{a, b, c\} \), the new one is \( \Sigma' = \{a, b, ab, bc, ca\} \). Having a vertex to be assigned \( ab \) represents that it has value \( a \) and \( b \). With this interpretation in mind, we redefine equality-like constraints for the intra-cloud edges so as to preserve the perfect completeness.

Unfortunately, this modification causes another issue, which renders the proof of soundness nontrivial. Example 3.9 illustrated in Figure 2 tells us that our reduction is neither a Karp reduction of Binary CSP Reconfiguration nor a PTAS reduction [16,17] of Maxmin Binary CSP Reconfiguration. One particular reason is that assigning conflicting values to vertices in a cloud may not break any equality-like constraints. Thankfully, we are “promised” that at least \( \varepsilon \)-fraction of constraints are violated for some \( \varepsilon \in (0, 1) \). We therefore use the following machinery to accomplish the soundness requirement:

- Use an explicit family of near-Ramanujan graphs [1,36] so that the second largest eigenvalue \( \lambda \) is \( O(\sqrt{d}) \); the degree \( d \) is determined based on the value of \( \varepsilon \).
- Apply the expander mixing lemma [2] to bound the number of violated edges, whereas Papadimitriou and Yannakakis [40] used the vertex expansion property, which is not applicable as shown in Example 3.9.

By applying this degree reduction step, we come back to Maxmin \( E_3 \)-SAT Reconfiguration, wherein, but this time, each variable appears in a constant number of clauses, completing the proof of the main result.
Once we have established gap-preserving reducibility from RIH to Maxmin E3-SAT Reconfiguration of bounded occurrence, we can apply it to devise conditional \textbf{PSPACE}-hardness of approximation for an optimization variant of \textit{Nondeterministic Constraint Logic} (Proposition 4.1). \textbf{Nondeterministic Constraint Logic} is a \textbf{PSPACE}-complete problem proposed by Hearn and Demaine [24, 25] that has been used to prove \textbf{PSPACE}-hardness of many games, puzzles, and other reconfiguration problems [5, 9, 11, 30]. We show that under RIH, it is \textbf{PSPACE}-hard to distinguish whether an input is a \textbf{yes} instance, or has a property that every transformation must violate more than $\varepsilon$-fraction of nodes. The proof makes a modification to the existing gadgets [24, 25]. As a consequence of Proposition 4.1, we demonstrate that assuming RIH, optimization variants of popular reconfiguration problems on a graph are \textbf{PSPACE}-hard to approximate, including \textit{Independent Set Reconfiguration}, \textit{Clique Reconfiguration}, and \textit{Vertex Cover Reconfiguration} (Corollaries 4.2–4.4), whose proofs are almost immediate from existing work [9, 24, 25].

Owing to space limitations, proofs marked with $\ast$ are omitted and can be found in the full version of this paper [38].

\textbf{Additional Related Work.} Other reconfiguration problems whose approximability was analyzed include \textit{Set Cover Reconfiguration} [28], which is 2-factor approximable, \textit{Subset Sum Reconfiguration} [27], which admits a PTAS, \textit{Shortest Path Reconfiguration} [22], and \textit{Submodular Reconfiguration} [39]. We note that approximability of reconfiguration problems frequently refers to that of \textit{the shortest sequence} [7, 8, 10, 12, 13, 26, 29, 35, 42], which is of independent interest. The objective value of optimization variants is sometimes called the \textit{reconfiguration index} [32] or \textit{reconfiguration threshold} [18]. A different type of optimization variant, called \textit{incremental optimization under the reconfiguration framework} [6, 31, 43] has recently been studied; e.g., starting from an initial independent set, we want to transform into a maximum possible independent set without touching those smaller than the specified size.

\section{Preliminaries}

\textbf{Notations.} For two integers $m, n \in \mathbb{N}$ with $m \leq n$, let $[n] \triangleq \{1, 2, \ldots, n\}$ and $[m..n] \triangleq \{m, m+1, \ldots, n-1, n\}$. A \textit{sequence} $\mathcal{E}$ of a finite number of elements $e^{(0)}, e^{(1)}, \ldots, e^{(\ell)}$ is denoted by $(e^{(0)}, e^{(1)}, \ldots, e^{(\ell)})$, and we write $e^{(i)} \in \mathcal{E}$ to indicate that $e^{(i)}$ appears in $\mathcal{E}$. We briefly recapitulate Ito et al.’s reconfiguration framework [28]. Suppose we are given a “definition” of feasible solutions for some combinatorial search problem and a symmetric
An adjacency relation can also be defined in terms of a “reconfiguration step,” which specifies how a solution can be transformed, e.g., a flip of a single variable assignment.

**Boolean Satisfiability.** We use the standard terminology and notation of Boolean satisfiability. Truth values are denoted by T or F. A Boolean formula \( \varphi \) consists of variables \( x_1, \ldots, x_n \) and the logical operators, AND (\( \land \)), OR (\( \lor \)), and NOT (\( \lnot \)). A truth assignment \( \sigma : \{x_1, \ldots, x_n\} \to \{T, F\} \) for \( \varphi \) is a mapping that assigns a truth value to each variable. A Boolean formula \( \varphi \) is said to be satisfiable if there exists some assignment \( \sigma \) such that \( \varphi \) evaluates to T when each variable \( x_i \) is assigned the truth value specified by \( \sigma(x_i) \). A literal is either a variable or its negation; a clause is a disjunction of literals. A Boolean formula is said to be in conjunctive normal form (CNF) if it is a conjunction of clauses. A k-CNF formula is a CNF formula in which every clause contains at most \( k \) literals. Hereafter, the prefix “Ek-” means that every clause has exactly \( k \) distinct literals, while the suffix “(B)” indicates that the number of occurrences of each variable is bounded by \( B \in \mathbb{N} \). We say that two truth assignments for a Boolean formula are adjacent if one is obtained from the other by flipping a single variable assignment; i.e., they differ in exactly one variable. In the \textsc{PSPACE}-complete \( k \)-SAT Reconfiguration problem [23], for a \( k \)-CNF formula \( \varphi \) and two satisfying truth assignments \( \sigma_0 \) and \( \sigma_1 \) for \( \varphi \), we are asked to decide if there exists a reconfiguration sequence of satisfying truth assignments for \( \varphi \) from \( \sigma_0 \) to \( \sigma_1 \). Since we are concerned with approximability of \( k \)-SAT Reconfiguration, we formulate its optimization variant [28], where we are allowed to go through non-satisfying truth assignments. For a CNF formula \( \varphi \) consisting of \( m \) clauses \( C_1, \ldots, C_m \) and a truth assignment \( \sigma \) for \( \varphi \), let \( \text{val}_\varphi(\sigma) \) denote the fraction of clauses in \( \varphi \) satisfied by \( \sigma \); namely,

\[
\text{val}_\varphi(\sigma) \triangleq \frac{|\{j \in [m] : \sigma \text{ satisfies } C_j\}|}{m}.
\]

For a reconfiguration sequence \( \sigma = (\sigma(0), \ldots, \sigma(t)) \) of truth assignments, let \( \text{val}_\varphi(\sigma) \) denote the minimum fraction of satisfied clauses over all \( \sigma(i) \)'s in \( \sigma \); i.e.,

\[
\text{val}_\varphi(\sigma) \triangleq \min_{\sigma(i) \in \sigma} \text{val}_\varphi(\sigma(i)).
\]

Then, Maxmin \( k \)-SAT Reconfiguration is defined as a problem of maximizing \( \text{val}_\varphi(\sigma) \) subject to \( \sigma = (\sigma_0, \ldots, \sigma_t) \), where \( \sigma_0 \) and \( \sigma_t \) are not necessarily satisfying. For two truth assignments \( \sigma_0 \) and \( \sigma_1 \) for \( \varphi \), let \( \text{val}_\varphi(\sigma_0 \leadsto \sigma_1) \) denote the maximum value of \( \text{val}_\varphi(\sigma) \) over all possible reconfiguration sequences \( \sigma \) from \( \sigma_0 \) to \( \sigma_1 \); namely,

\[
\text{val}_\varphi(\sigma_0 \leadsto \sigma_1) \triangleq \max_{\sigma=(\sigma_0, \ldots, \sigma_t)} \text{val}_\varphi(\sigma) = \max_{\sigma=(\sigma_0, \ldots, \sigma_t)} \min_{\sigma(i) \in \sigma} \text{val}_\varphi(\sigma(i)).
\]

Note that \( \text{val}_\varphi(\sigma_0 \leadsto \sigma_1) \leq \min\{\text{val}_\varphi(\sigma_0), \text{val}_\varphi(\sigma_1)\} \). If \( \text{val}_\varphi(\sigma_0 \leadsto \sigma_1) \geq \rho \) for some \( \rho \), we can transform \( \sigma_0 \) into \( \sigma_1 \) while ensuring that every intermediate truth assignment satisfies at least \( \rho \)-fraction of the clauses of \( \varphi \). We finally define the “gap version” of Maxmin \( k \)-SAT Reconfiguration as follows.

---

1 An adjacency relation can also be defined in terms of a “reconfiguration step,” which specifies how a solution can be transformed, e.g., a flip of a single variable assignment.
Problem 2.1. For every \(k \in \mathbb{N}\) and \(0 \leq s \leq c \leq 1\), \(\text{Gap}_{c,s} k\text{-SAT}\) Reconfiguration requests to determine for a \(k\)-CNF formula \(\varphi\) and two truth assignments \(\sigma_s\) and \(\sigma_t\) for \(\varphi\), whether \(\text{val}_\varphi(\sigma_s \leadsto \sigma_t) \geq c\) (the input is a yes instance) or \(\text{val}_\varphi(\sigma_s \leadsto \sigma_t) < s\) (the input is a no instance). Here, \(c\) and \(s\) denote completeness and soundness, respectively.

Problem 2.1 is a promise problem, in which we are allowed to output anything when \(s \leq \text{val}_\varphi(\sigma_s \leadsto \sigma_t) < c\). The present problem definition does not request an actual reconfiguration sequence. Note that we can assume \(\sigma_s\) and \(\sigma_t\) to be satisfying ones whenever \(c = 1\), and the case of \(s = c = 1\) particularly reduces to \(k\)-SAT Reconfiguration.

Constraint Satisfaction Problem. Subsequently, we introduce reconfiguration problems on constraint satisfaction. First, we define the notion of constraint graphs.

Definition 2.2 (Constraint graph). A \(q\)-ary constraint graph is defined as a tuple \(G = (V, E, \Sigma, \Pi)\) such that \((V, E)\) is a \(q\)-uniform hypergraph called the underlying graph of \(G\), \(\Sigma\) is a finite set called the alphabet, and \(\Pi = (\pi_e)_{e \in E}\) is a collection of \(q\)-ary constraints, where each \(\pi_e \subseteq \Sigma^q\) is a set of \(q\)-tuples of acceptable values that \(q\) vertices in \(e\) can take. The degree \(d_G(v)\) of each vertex \(v\) in \(G\) is defined as the number of hyperedges including \(v\).

An assignment is a mapping \(\psi : V \to \Sigma\) that assigns a value of \(\Sigma\) to each vertex of \(V\). We say that \(\psi\) satisfies hyperedge \(e = \{v_1, \ldots, v_q\} \in E\) (or constraint \(\pi_e\)) if \(\psi(e) \triangleq (\psi(v_1), \ldots, \psi(v_q)) \in \pi_e\), and \(\psi\) satisfies \(G\) if it satisfies all hyperedges of \(G\). We say that \(G\) is satisfiable if some assignment that satisfies \(G\) exists. Two assignments are said to be adjacent if they differ in exactly one vertex. In \(q\)-CSP Reconfiguration, for a \(q\)-ary constraint graph \(G\) and two satisfying assignments \(\psi_s\) and \(\psi_t\) for \(G\), we are asked to decide if there is a reconfiguration sequence of satisfying assignments for \(G\) from \(\psi_s\) to \(\psi_t\). Hereafter, BCSP stands for \(2\)-CSP, \(q\)-CSP\(_W\) designates the restricted case that the alphabet size \(|\Sigma|\) is some \(W \in \mathbb{N}\), and \(q\)-CSP(\(\Delta\)) for some \(\Delta \in \mathbb{N}\) means that the maximum degree of the constraint graph is bounded by \(\Delta\).

In an analogous way to the case of Boolean satisfiability, we define \(\text{val}_G(\psi) \triangleq \frac{\#(e \in E : \psi\text{satisfies } e)}{|E|}\) for assignment \(\psi : V \to \Sigma\), \(\text{val}_G(\Psi) \triangleq \min_{\psi \in \Psi} \text{val}_G(\psi)\) for reconfiguration sequence \(\Psi = (\psi^{(i)})_{i \in [0, \ell]}\), and \(\text{val}_G(\psi_s \leadsto \psi_t) \triangleq \max_{\Psi = (\psi_s, \ldots, \psi_t)} \text{val}_G(\Psi)\) for \(\psi_s, \psi_t : V \to \Sigma\). In Maxmin \(q\)-CSP Reconfiguration, we wish to maximize \(\text{val}_G(\psi)\) subject to \(\Psi = (\psi_s, \ldots, \psi_t)\). The corresponding gap version is defined as follows.

Problem 2.3. For every \(q \in \mathbb{N}\) and \(0 \leq s \leq c \leq 1\), \(\text{Gap}_{c,s} q\text{-CSP}\) Reconfiguration requests to determine for a \(q\)-ary constraint graph \(G\) and two (not necessarily satisfying) assignments \(\psi_s\) and \(\psi_t\) for \(G\), whether \(\text{val}_G(\psi_s \leadsto \psi_t) \geq c\) or \(\text{val}_G(\psi_s \leadsto \psi_t) < s\).

Reconfiguration Inapproximability Hypothesis. We now present a formal description of our working hypothesis, which serves as a starting point for \(\text{PSPACE}\)-hardness of approximation.

Hypothesis 2.4 (Reconfiguration Inapproximability Hypothesis, RIH). There exist universal constants \(q, W \in \mathbb{N}, \epsilon \in (0, 1)\) such that \(\text{Gap}_{1,1-\epsilon} q\text{-CSP}\(_W\) Reconfiguration is \(\text{PSPACE}\)-hard.

3 Hardness of Approximation for Maxmin \(E^3\)-SAT \((B)\) Reconfiguration

In this section, we prove the main result of this paper; that is, Maxmin \(E^3\)-SAT Reconfiguration of bounded occurrence is \(\text{PSPACE}\)-hard to approximate under RIH.
Theorem 3.1. Under Hypothesis 2.4, there exist universal constants \( c \in (0, 1) \) and \( B \in \mathbb{N} \) such that \( \text{Gap}_{1,1-c^c} \text{-E3-SAT}(B) \) Reconfiguration is PSPACE-hard.

The remainder of this section is devoted to the proof of Theorem 3.1 and organized as follows. In Section 3.1, we reduce Maxmin \( q \)-CSP\(_W\) Reconfiguration to Maxmin BCSP\(_3\) Reconfiguration; Section 3.2 presents the degree reduction of Maxmin BCSP Reconfiguration.

### 3.1 Maxmin \( q \)-CSP\(_W\) Reconfiguration to Maxmin BCSP\(_3\) Reconfiguration

We first reduce from Maxmin \( q \)-CSP\(_W\) Reconfiguration to Maxmin E3-SAT Reconfiguration.

Lemma 3.2 (*). For every \( q, W \geq 2 \) and \( \epsilon \in (0, 1) \), there exists a gap-preserving reduction from \( \text{Gap}_{1,1-\epsilon q \text{-CSP}\(_W\)} \) Reconfiguration to \( \text{Gap}_{1,1-\epsilon q-W} \text{-E3-SAT} \). Moreover, if the maximum degree of the constraint graph in the former problem is \( \Delta \), then the number of occurrences of each variable in the latter problem is bounded by \( W^q \cdot 2^{W \Delta} \).

The proof of Lemma 3.2 consists of a reduction from Maxmin \( q \)-CSP\(_W\) Reconfiguration to Maxmin E3-SAT Reconfiguration, where the clause size \( k \) depends solely on \( q \) and \( W \), and that from Maxmin E3-SAT Reconfiguration to Maxmin E3-SAT Reconfiguration. In the first reduction, we apply a slightly sophisticated SAT encoding, described below. In the second reduction, we use an established Karp reduction from \( k \)-SAT to 3-SAT, previously used by Gopalan, Kolaitis, Maneva, and Papadimitriou [23] in the context of reconfiguration.

**Reconfigurable SAT Encoding.** For the proof of the first reduction, we introduce an encoding of the alphabet of a constraint graph into a string of truth values. Hereafter, we denote \( \Sigma \triangleq [W] \) for some integer \( W \in \mathbb{N} \). Consider an encoding \( \text{enc}: \{T,F\}^\Sigma \rightarrow \Sigma \) of a binary string \( s \in \{T,F\}^\Sigma \) to \( \Sigma \) defined as follows:

\[
\text{enc}(s) \triangleq \begin{cases} 
1 & \text{if } s_\alpha = F \text{ for all } \alpha \in \Sigma, \\
\alpha & \text{if } s_\alpha = T \text{ and } s_\beta = F \text{ for all } \beta > \alpha.
\end{cases}
\]

The encoding \( \text{enc} \) exhibits the following property concerning reconfigurability:

Claim 3.3 (*). For any two strings \( s \) and \( t \) in \( \{T,F\}^\Sigma \) with \( \alpha \triangleq \text{enc}(s) \) and \( \beta \triangleq \text{enc}(t) \), we can transform \( s \) into \( t \) by repeatedly flipping one entry at a time while preserving every intermediate string mapped to \( \alpha \) or \( \beta \) by \( \text{enc} \).

We use \( \text{enc} \) to encode each \( q \)-tuple of unacceptable values \( (\alpha_1, \ldots, \alpha_q) \in \Sigma^q \setminus \pi_{\epsilon} \) for hyperedge \( e = \{v_1, \ldots, v_q\} \in E \).

We then reduce Maxmin E3-SAT Reconfiguration to Maxmin BCSP\(_3\) Reconfiguration in a gap-preserving manner, whose proof uses the place encoding due to Järvsalo and Niemelä [33].

Lemma 3.4 (*). For every \( \epsilon \in (0, 1) \), there exists a gap-preserving reduction from \( \text{Gap}_{1,1-\epsilon} \text{-E3-SAT} \) to \( \text{Gap}_{1,1-\epsilon} \text{-BCSP}\(_3\) \). Moreover, if the number of occurrences of each variable in the former problem is \( B \), then the maximum degree of the constraint graph in the latter problem is bounded by \( \max\{B, 3\} \).
Reduction. We here describe a reduction from Maxmin 3-SAT Reconfiguration to Maxmin BCSP$_3$ Reconfiguration. Let $(\varphi, \sigma_0, \sigma_1)$ be an instance of Maxmin 3-SAT Reconfiguration, where $\varphi$ is a 3-CNF formula consisting of $m$ clauses $C_1, \ldots, C_m$ over $n$ variables $x_1, \ldots, x_n$ and $\sigma_0$ and $\sigma_1$ satisfy $\varphi$. Using the place encoding due to Järvisalo and Niemelä [33], we construct a binary constraint graph $G = (V, E, \Sigma, \Pi)$ as follows. The underlying graph of $G$ is a bipartite graph with a bipartition $\{(x_1, \ldots, x_n), \{C_1, \ldots, C_m\}\}$, and there is an edge between variable $x_i$ and clause $C_j$ in $E$ if $x_i$ or $\overline{x_i}$ appears in $C_j$. For the sake of notation, we use $\Sigma_0$ to denote the alphabet assigned to vertex $v \in V$. We then express $\Sigma_{x_i} \triangleq \{T, F\}$ for each variable $x_i$, and $\Sigma_{C_j} \triangleq \{\ell_1, \ell_2, \ell_3\}$ for each clause $C_j = (\ell_1 \lor \ell_2 \lor \ell_3)$. For each edge $(x_i, C_j) \in E$ with $C_j = (\ell_1 \lor \ell_2 \lor \ell_3)$, the constraint $\pi_{(x_i, C_j)} \subseteq \Sigma_{x_i} \times \Sigma_{C_j}$ is defined as follows:

$$\pi_{(x_i, C_j)} \triangleq \begin{cases} (\Sigma_{x_i} \times \Sigma_{C_j}) \setminus \{(F, x_i)\} & \text{if } x_i \text{ appears in } C_j, \\ (\Sigma_{x_i} \times \Sigma_{C_j}) \setminus \{(T, \overline{x_i})\} & \text{if } \overline{x_i} \text{ appears in } C_j. \end{cases}$$

(5)

For an assignment $\psi$ for $G$, $\psi(x_i)$ represents the truth value assigned to $x_i$, and $\psi(C_j)$ specifies which literal should evaluate to $T$. Note that $|V| = n + m$, $|E| = 3m$, and the maximum degree of $(V, E)$ is $\max \{B, 3\}$. For a satisfying truth assignment $\sigma$ for $\varphi$, let $\psi_\sigma : V \to \Sigma$ be an assignment for $G$ defined as follows: $\psi_\sigma(x_i) \triangleq \sigma(x_i)$ for each variable $x_i$, and $\psi_\sigma(C_j) \triangleq \ell_i$ for each clause $C_j$ whenever $\ell_i$ appears in $C_j$ and evaluates to $T$ by $\sigma$.2 Obviously, $\psi_\sigma$ satisfies $G$. Constructing $\psi_\sigma$ from $\sigma_0$ and $\psi_1$ from $\sigma_1$ according to this procedure, we obtain an instance $(G, \psi_\sigma, \psi_\tau)$ of Maxmin BCSP$_3$ Reconfiguration, which completes the reduction. It is not hard to see that the above reduction is a gap-preserving reduction, whose proof can be found in the full version [38].

3.2 Degree Reduction of Maxmin BCSP Reconfiguration

We now present a gap-preserving reduction from Maxmin BCSP Reconfiguration to itself of bounded degree, which is the most technical step in this paper.

Lemma 3.5. For every $\varepsilon \in (0, 1)$, there exists a gap-preserving reduction from Gap$_{1,1-\varepsilon}$ BCSP$_3$ Reconfiguration to Gap$_{1,1-\varepsilon}$ BCSP$_6(\Delta)$ Reconfiguration, where $\tau \in (0, 1)$ and $\Delta \in \mathbb{N}$ are some computable functions dependent only on the value of $\varepsilon$. In particular, the constraint graph in the latter problem is of bounded degree for fixed $\varepsilon$.

We first introduce the notion of expander graphs.

Definition 3.6 (Expander graph). For every $n \in \mathbb{N}$, $d \in \mathbb{N}$, and $\lambda > 0$, an $(n, d, \lambda)$-expander graph is a $d$-regular graph $G$ on $n$ vertices such that $\max\{\lambda_2(G), |\lambda_n(G)|\} \leq \lambda < d$, where $\lambda_i(G)$ is the $i$-th largest (real-valued) eigenvalue of the adjacency matrix of $G$.

An $(n, d, \lambda)$-expander graph is called Ramanujan if $\lambda \leq 2\sqrt{d-1}$. There exists an explicit construction (i.e., a polynomial-time algorithm) for near-Ramanujan graphs.

Theorem 3.7 (Explicit construction of near-Ramanujan graphs [1,36]). For every constant $d \geq 3$, $\varepsilon > 0$, and all sufficiently large $n \geq n_0(d, \varepsilon)$, where $n$ is even, there is a deterministic $n^{O(1)}$-time algorithm that outputs an $(n, d, \lambda)$-expander graph with $\lambda \leq 2\sqrt{d-1} + \varepsilon$.

In this paper, we rely only on the special case of $\varepsilon = 2\sqrt{d} - 2\sqrt{d-1}$ so that $\lambda \leq 2\sqrt{d}$; thus, we let $n_0(d) \triangleq n_0(d, 2\sqrt{d} - 2\sqrt{d-1})$. We can assume $n_0(\cdot)$ to be computable as $2\sqrt{d} - 2\sqrt{d-1} \geq \frac{1}{2\sqrt{d}}$. The crucial property of expander graphs that we use in the proof of Lemma 3.5 is the following expander mixing lemma [2].

2 Such $\ell_i$ always exists as $\sigma$ satisfies $C_j$, and ties are broken arbitrarily.
Lemma 3.8 (Expander mixing lemma; e.g., Alon and Chung [2]). Let \( G \) be an \((n, d, \lambda)\)-expander graph. Then, for any two sets \( S, T \) of vertices, it holds that

\[
|e(S, T) - \frac{d|S| \cdot |T|}{n}| \leq \lambda \sqrt{|S| \cdot |T|},
\]

where \( e(S, T) \) counts the number of edges between \( S \) and \( T \).

This Lemma states that \( e(S, T) \) of an expander graph \( G \) is concentrated around its expectation if \( G \) were a random \( d \)-regular graph. The use of near-Ramanujan graphs enables us to make an additive error (i.e., \( \lambda \sqrt{|S| \cdot |T|} \)) acceptably small.

**Reduction.** We then explain our gap-preserving reduction, which does depend on \( \varepsilon \). Redefine \( \varepsilon \leftarrow [1/\varepsilon]^{-1} \) so that \( 1/\varepsilon \) is a positive integer, which does not increase the value of \( \varepsilon \); i.e., \( \text{val}_G(\psi_\varepsilon \leftrightarrow \psi_\varepsilon) < 1 - \varepsilon \) implies \( \text{val}_G(\psi_\varepsilon \leftrightarrow \psi_\varepsilon) < 1 - [1/\varepsilon]^{-1} \). Let \( (G = (V, E, \Sigma, \Pi = (\pi_\varepsilon)_{e \in E}), \psi_\varepsilon, \psi_\varepsilon) \) be an instance of \( \text{Gap}_{1,-\varepsilon} \) \( \text{BCSP}_3 \) \( \text{Reconfiguration} \), where \( \psi_\varepsilon \) and \( \psi_\varepsilon \) satisfy \( G \). For the sake of notation, we denote \( \Sigma \triangleq \{a, b, c\} \). We then create a new instance \((G' = (V', E', \Sigma', \Pi'(\pi'_{\varepsilon'})_{e \in E'}), \psi'_{\varepsilon}, \psi'_{\varepsilon'})\) of \( \text{Maxmin BCSP}_3 \) \( \text{Reconfiguration} \), which turns out to meet the requirement of completeness and soundness. The constraint graph \( G' \) is defined as follows:

**Vertex set:** For each vertex \( v \) of \( V \), let

\[
\text{cloud}(v) \triangleq \{(v, e) : e \in E \text{ is incident on } v\}.
\]

Define \( V' \triangleq \bigcup_{v \in V} \text{cloud}(v) \).

**Edge set:** For each vertex \( v \) of \( V \), let \( X_v \) be a \((d_G(v), d_0, \lambda)\)-expander graph on \( \text{cloud}(v) \) using Theorem 3.7 if \( d_G(v) \geq n_0(d_0) \), or a complete graph on \( \text{cloud}(v) \) if \( d_G(v) < n_0(d_0) \). Here, \( \lambda \leq 2\sqrt{d_0} \) and \( d_0 = \Theta(\varepsilon^{-2}) \), whose precise value will be determined later. Define

\[
E' \triangleq \bigcup_{v \in V} E(X_v) \cup \left\{((v, e), (w, e)) : e = (v, w) \in E\right\}.
\]

**Alphabet:** Define \( \Sigma' \triangleq \{\{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{c, a\}\} \). By abuse of notation, we write each value of \( \Sigma' \) as if it were an element (e.g., \( ab \in \Sigma', a \subset ab, \) and \( b \not\subset ca \)).

**Constraints:** The constraint \( \pi'_{\varepsilon'} \subseteq \Sigma' \) for each edge \( e' \in E' \) is defined as follows:

- If \( e' \in E(X_v) \) for some \( v \in V \) (i.e., \( e' \) is an intra-cloud edge), define \(^3\)

\[
\pi'_{\varepsilon'} \triangleq \{(\alpha, \beta) : \alpha, \beta \in \Sigma', \alpha \subseteq \beta \text{ or } \beta \subseteq \alpha\}.
\]

- If \( e' = ((v, e), (w, e)) \) such that \( e = (v, w) \in E \) (i.e., \( e' \) is an inter-cloud edge), define

\[
\pi'_{\varepsilon'} \triangleq \{(\alpha, \beta) : \alpha \times \beta \subseteq \pi_\varepsilon\}.
\]

Although the underlying graph \((V', E')\) is the same as that in [19] (except for the use of Theorem 3.7), the definitions of \( \Sigma' \) and \( \Pi' \) are somewhat different, which is essential to ensure the perfect completeness, while making the proof of the soundness nontrivial. Intuitively, having vertex \( v' \in V' \) be \( \psi(v') = ab \) represents that \( v' \) has values \( a \) and \( b \) simultaneously;

\(^3\) Eq. (9) can be explicitly expanded as \( \pi'_{\varepsilon'} = \{a, a\}, \{b, b\}, \{c, c\}, \{ab, a\}, \{ab, b\}, \{bc, b\}, \{bc, c\}, \{ca, c\}, \{ca, a\}, \{ab, ab\}, \{bc, bc\}, \{ca, ca\}\).
e.g., if $\psi'(v') = ab$ and $\psi'(w') = c$ for some $v' \in \text{cloud}(v)$ and $w' \in \text{cloud}(w)$ with $v \neq w$, then $\psi'$ satisfies $\pi'_{v',w'}$ if both $(a, b)$ and $(a, c)$ are in $\pi_{v,w}$ owing to Eq. (10). Construct two assignments $\psi_1^G : V' \to \Sigma'$ from $\psi_2$ and $\psi_2' : V' \to \Sigma'$ from $\psi_1$ such that $\psi_1'(v, e) \triangleq \{ \psi_2(v) \}$ and $\psi_2'(v, e) \triangleq \{ \psi_2(v) \}$ for all $(v, e) \in V'$. Observe that both $\psi'_1$ and $\psi'_2$ satisfy $\Sigma'$, thereby completing the reduction. Note that $|V'| = 2|E|$, $|E'| \leq n_0(d_0) \cdot |E|$, $|\Sigma'| = 6$, and the maximum degree of $G'$ is $\Delta \leq n_0(d_0)$, which is constant for fixed $\varepsilon$.

Using an example illustrated in Figure 2, we demonstrate that our reduction may map a NO instance of BCSP Reconfiguration to a YES instance; namely, it is neither a Karp reduction of BCSP Reconfiguration nor a PTAS reduction of Maxmin BCSP Reconfiguration.

Example 3.9. We construct a constraint graph $G = (V, E; \Sigma, \Pi = (\pi_x)_{x \in E})$ such that $V \triangleq \{ \varepsilon, v, x, y, z_1, \ldots, z_n \}$ for some large integer $n$, $E \triangleq \{(v, w), (v, x), (v, y), (v, z_1), \ldots, (v, z_n)\}$, $\Sigma \triangleq \{ a, b, c \}$, and each $\pi_x$ is defined as follows: $\pi_{(v, w)} \triangleq \{ \{ a, a \}, \{ a, b \}, \{ b, b \} \}$, $\pi_{(v, x)} \triangleq \{ \{ a, a \}, \{ b, a \}, \{ b, b \} \}$, $\pi_{(v, y)} \triangleq \{ \{ a, a \}, \{ a, b \}, \{ b, b \} \}$, and $\pi_{(v, z_i)} = \cdots = \pi_{(v, z_n)} \triangleq \Sigma 	imes \Sigma$. Define $\psi_3, \psi_4 : V \to \Sigma$ such that $\psi_3(x) \triangleq a$ for all $u \in V$, $\psi_4(x) \triangleq c$, and $\psi_4(u) \triangleq a$ for all other $u$. Then, it is impossible to transform $\psi_3$ into $\psi_4$ without any constraint violation: As the values of $w$ and $v$ cannot change from $a$ to $c$, we can only change the value of $x$ to $c$, violating $(x, y)$. In particular, $val_G(\psi_3 \rightsquigarrow \psi_4) < 1$.

Consider applying our reduction to $v$ only. Create cloud$(v) \triangleq \{ v_w, v_x, v_{z_1}, \ldots, v_{z_n} \}$ with the shorthand notation $v_u \triangleq (v, (v, u))$, and let $\text{X}_v$ be an expand connected graph on cloud$(v)$. We then construct a new constraint graph $G' = (V', E', \Sigma', \Pi') = (\pi'_{x \in E'})$, where $V' \triangleq \{ w, x, y, z_1, \ldots, z_n \} \cup \text{cloud}(v)$, $E' \triangleq E(\text{X}_v) \cup \{(w, v_u), (v_x, x), (v_y, y), (v_{z_1}, z_1), \ldots, (v_{z_n}, z_n)\}$, $\Sigma' \triangleq \{ a, b, c, ab, bc, ca \}$, and each $\pi'_x$ is defined according to Eqs. (9) and (10). Construct $\psi'_3, \psi'_4 : V' \to \Sigma'$ from $\psi_3, \psi_4$ using the procedure described above. Suppose now “by chance” $(v_w, v_x) \notin E(\text{X}_v)$. The crucial observation is that we can assign $a$ to $v_w$, $b$ to $v_x$, and $ab$ to $v_{z_1}, \ldots, v_{z_n}$ to do some “cheating.” Consequently, $\psi'_4$ can be transformed into $\psi'_3$ without sacrificing any constraint: Assign $ab$ to $v_{z_1}, \ldots, v_{z_n}$ in arbitrary order; assign $b$ to $v_x$, and $a$ to $v_{z_1}, \ldots, v_{z_n}$ in arbitrary order. In particular, $val_{G'}(\psi'_3 \rightsquigarrow \psi'_4) = 1$.

Correctness. The proof of the completeness is immediate from the definition of $\Sigma'$ and $\Pi'$.

Lemma 3.10 (*). If $val_G(\psi_3 \rightsquigarrow \psi_4) = 1$, then $val_{G'}(\psi'_3 \rightsquigarrow \psi'_4) = 1$.

Then, in the remainder of this subsection, we prove the soundness.

Lemma 3.11. If $val_G(\psi_3 \rightsquigarrow \psi_4) < 1 - \varepsilon$, then $val_{G'}(\psi'_3 \rightsquigarrow \psi'_4) < 1 - \varepsilon$, where $\varepsilon = \varepsilon(\varepsilon)$ is some computable function such that $\varepsilon \in (0, 1)$ if $\varepsilon \in (0, 1)$.

For an assignment $\psi' : V' \to \Sigma'$ for $G'$, let $\text{PLR}(\psi') : V \to \Sigma$ denote an assignment for $G$ such that $\text{PLR}(\psi'(v))$ for $v \in V$ is determined based on the plurality vote of $\psi'(v')$ over $v' \in \text{cloud}(v)$; namely,

$$\text{PLR}(\psi')(v) \triangleq \arg\max_{\alpha \in \Sigma} \left| \left\{ v' \in \text{cloud}(v) : \alpha \in \psi'(v') \right\} \right|,$$

where ties are arbitrarily broken according to any prefixed ordering over $\Sigma$ (e.g., $a < b < c$). Suppose we have a reconfiguration sequence $\Psi = \langle \psi'^{(0)} = \psi'_3, \ldots, \psi'^{(\ell)} = \psi'_4 \rangle$ for $(G', \psi'_3, \psi'_4)$ with the maximum value. Construct then a sequence of assignments, $\Psi \triangleq \langle \psi^{(i)} \rangle_{i \in [0 \ldots \ell]}$, such that $\psi^{(i)} \triangleq \text{PLR}(\psi^{(i)})$ for all $i \in [0 \ldots \ell]$. Observe that $\Psi$ is a valid reconfiguration sequence for $(G, \psi_3, \psi_4)$, and we thus must have $val_G(\Psi) < 1 - \varepsilon$; in particular, there exists
some $\psi^{(i)}$ such that $\text{val}_G(\text{PLR}(\psi^{(i)})) = \text{val}_G(\psi^{(i)}) < 1 - \varepsilon$. We would like to show that $\text{val}_G(\psi^{(i)}) < 1 - \tau$ for some $\tau \in (0, 1)$. Hereafter, we denote $\psi \triangleq \psi^{(i)}$ and $\psi' \triangleq \psi^{(i)}$ for notational simplicity.

For each vertex $v \in V$, we define $D_v$ as the set of vertices in cloud($v$) whose values disagree with the plurality vote $\psi(v)$; namely,

$$D_v \triangleq \left\{ v' \in \text{cloud}(v) : \psi(v) \not\in \psi(v') \right\}. \quad (12)$$

Consider any edge $e = (v, w) \in E$ violated by $\psi$ (i.e., $(\psi(v), \psi(w)) \not\in \pi_\psi$), and let $e' = (v', w') \in E'$ be a unique (inter-cloud) edge such that $v' \in \text{cloud}(v)$ and $w' \in \text{cloud}(w)$. By definition of $\pi'_\psi$, either of the following must hold: (1) edge $e'$ is violated by $\psi'$ (i.e., $(\psi'(v'), \psi'(w')) \not\in \pi'_\psi$), or (2) $\psi(v) \not\in \psi'(v')$ (i.e., $v' \in D_v$) or $\psi(w) \not\in \psi'(w')$ (i.e., $w' \in D_w$). Consequently, the number of edges in $E$ violated by $\psi$ is bounded by the sum of the number of edges in $E'$ violated by $\psi'$ and the number of vertices in $V'$ that disagree with the plurality vote; namely,

$$|E| < (\# \text{ edges violated by } \psi') + \sum_{v \in V} |D_v|. \quad (13)$$

Then, one of the two terms on the right-hand side should be greater than $\frac{\varepsilon}{2}|E|$. If the number of edges violated by $\psi'$ is more than $\frac{\varepsilon}{2}|E|$, then we have done because

$$\text{val}_{G'}(\psi') \leq \frac{|E'| - (\# \text{ edges violated by } \psi')}{|E'|} < 1 - \frac{\varepsilon}{2} \leq 1 - \frac{\varepsilon}{2} \cdot n_0(d_0). \quad (14)$$

We now consider the case that $\sum_{v \in V} |D_v| > \frac{\varepsilon}{4}|E|$. Define $x_v \triangleq |D_v|/d_G(v)$ for each $v \in V$, which is the fraction of vertices in cloud($v$) who disagree with $\psi(v)$. We also define $\delta \triangleq \frac{\varepsilon}{4}$. We first show that the total size of $|D_v|$ conditioned on $x_v \geq \delta$ is $\Theta(\varepsilon|E|)$.

Claim 3.12 ($\triangleright$). $\sum_{v \in V : x_v \geq \delta} |D_v| > \frac{\varepsilon}{4}|E|$, where $\delta = \frac{\varepsilon}{4}$.

We then discover a pair of disjoint subsets of cloud($v$) for every $v \in V$ such that their size is $\Theta(|D_v|)$ and they are mutually conflicting under $\psi'$, where the fact that $|\Sigma| = 3$ somewhat simplifies the proof by cases.

Claim 3.13 ($\triangleright$). For each vertex $v$ of $V$, there exists a pair of disjoint subsets $S, T$ of cloud($v$) such that $|S| \geq \frac{|D_v|}{4}$, $|T| \geq \frac{|D_v|}{4}$, and $\psi'$ violates all constraints between $S$ and $T$.

Consider a vertex $v \in V$ such that $x_v \geq \delta$; that is, at least $\delta$-fraction of vertices in cloud($v$) disagree with $\psi(v)$. Letting $S$ and $T$ be two disjoint subsets of cloud($v$) obtained by Claim 3.13, we wish to bound the number of edges between $S$ and $T$ (i.e., $e(S, T)$) using the expander mixing lemma. Hereafter, we determine the value of $d_0$ by $d_0 \triangleq (\frac{12}{\delta})^2 = \frac{2256}{\delta^2}$, which is a positive even integer (so that Theorem 3.7 is applicable) and depends only on the value of $\varepsilon$. Suppose first $d_G(v) \geq n_0(d_0)$; i.e., $X_v$ is an expander graph.

Lemma 3.14. For a vertex $v$ of $V$ such that $x_v \geq \delta$ and $d_G(v) \geq n_0(d_0)$, let $S$ and $T$ be a pair of disjoint subsets of cloud($v$) obtained by Claim 3.13. Then, $e(S, T) \geq \frac{\varepsilon}{3}|D_v|$.

Proof sketch. Recall that $X_v$ is a $(d_G(v), d_0, \lambda)$-expander graph, where $\lambda \leq 2\sqrt{d_0}$. By applying the expander mixing lemma on $S$ and $T$, we obtain

$$e(S, T) \geq \frac{d_0 |S| \cdot |T|}{d_G(v)} - \lambda \sqrt{|S| \cdot |T|} \geq \frac{|S| \cdot |T|}{d_G(v)} \left( \frac{12}{\delta} \right)^2 - \frac{2 \cdot 12}{\delta} \sqrt{|S| \cdot |T|}. \quad (15)$$
It is easy to see that \( e(S,T) \) is monotonically increasing in \( \sqrt{|S| \cdot |T|} \) when \( \sqrt{|S| \cdot |T|} > \frac{\delta}{d_G(v)} \). Observing that \( \sqrt{|S| \cdot |T|} \geq \frac{\delta}{d_G(v)} \) since \( |S| \geq \frac{\delta}{2} d_G(v) \), \( |T| \geq \frac{\delta}{2} d_G(v) \), and \( x_v \geq \delta \) by assumption, we derive

\[
e(S,T) \geq e(S,T) \geq \frac{1}{d_G(v)} \left( \frac{x_v \cdot d_G(v)}{3} \right)^2 \left( \frac{12}{\delta} \right)^2 - \frac{2 \cdot 12 x_v \cdot d_G(v)}{3} \\
\geq \frac{1}{d_G(v)} \left( \frac{x_v \cdot d_G(v)}{3} \right) \left( \frac{\delta \cdot d_G(v)}{3} \right) \left( \frac{12}{\delta} \right)^2 - \frac{2 \cdot 12 x_v \cdot d_G(v)}{3} = \frac{8}{\delta} |D_v|, \quad \Box
\]

Suppose then \( d_G(v) < n_0(d_0) \). Since \( X_v \) forms a complete graph over \( d_G(v) \) vertices, \( e(S,T) \) is exactly equal to \( |S| \cdot |T| \), which is evaluated as

\[
e(S,T) = |S| \cdot |T| \geq \left( \frac{|D_v|}{3} \right)^2 = \frac{x_v \cdot d_G(v)}{9} |D_v| \geq \frac{\delta}{9} |D_v|. \quad (16)
\]

By Lemma 3.14 and Eq. (16), for every vertex \( v \in V \) such that \( x_v \geq \delta \), the number of violated intra-cloud edges within \( X_v \) is at least \( \min \{ \frac{\delta}{3}, \frac{\delta}{9} \} |D_v| \geq \frac{\delta}{9} |D_v| \). Simple calculation using Claim 3.12 bounds the total number of edges violated from below as

\[
\sum_{v \in V} (\# \text{ violated edges in } X_v) \geq \sum_{v, x_v \geq \delta} \frac{\delta}{9} |D_v| \geq \frac{\delta}{9} |E| \geq \frac{\delta^2 \cdot |E'|}{288 \cdot n_0(d_0)} \cdot \quad (17)
\]

Consequently, from Eqs. (14) and (17), we conclude that

\[
\text{val}^\tau_G(\psi') \leq \text{val}^\tau_G(\psi') < \max \left\{ 1 - \frac{\varepsilon}{2 \cdot n_0(d_0)}, 1 - \frac{\varepsilon^2}{288 \cdot n_0(d_0)} \right\} = 1 - \frac{\varepsilon^2}{288 \cdot n_0 \left( \frac{\delta^2}{9} \right)}.
\]

Setting \( \varepsilon \triangleq \frac{\varepsilon^2}{288 \cdot n_0 \left( \frac{\delta^2}{9} \right)} \) accomplishes the proof of Lemma 3.11 and thus Lemma 3.5. \( \Box \)

### 4 Applications

Here, we apply Theorem 3.1 to devise conditional PSPACE-hardness of approximation for Nondeterministic Constraint Logic and popular reconfiguration problems on graphs.

#### 4.1 Optimization Variant of Nondeterministic Constraint Logic

First, we review Nondeterministic Constraint Logic developed by Hearn and Demaine [24, 25]. An AND/OR graph is defined as an undirected graph \( G = (V,E) \), where each edge of \( E \) is colored red or blue and has weight 1 or 2, respectively, and each node of \( V \) is one of two types:

- **AND node**, which has two incident red edges and one incident blue edge, or
- **OR node**, which has three incident blue edges.

An orientation (i.e., an assignment of direction to each edge) for \( G \) satisfies a particular node of \( G \) if the total weight of incoming edges is at least 2, and satisfies \( G \) if all nodes are satisfied. AND and OR nodes behave like the corresponding logical gates: the blue edge of an AND node can be directed outward if and only if both two red edges are directed inward; a particular

---

4 We refer to vertices of an AND/OR graph as nodes to distinguish from those of a standard graph.
blue edge of an or node can be directed outward if and only if at least one of the other two blue edges is directed inward. Thus, a direction of each edge can be considered a signal. In the PSPACE-complete Nondeterministic Constraint Logic problem, for an AND/OR graph \( G \) and two satisfying orientations \( O_\sigma \) and \( O_t \) for \( G \), we are asked if \( O_\sigma \) can be transformed into \( O_t \) by repeating the reversal of a single edge while ensuring that every intermediate orientation satisfies \( G \).\(^5\)

We now formulate an optimization variant of Nondeterministic Constraint Logic, where we are allowed to use an orientation that does not satisfy some nodes. Once more, we define \( \text{val}_G(\cdot) \) analogously: Let \( \text{val}_G(O) \) denote the fraction of nodes satisfied by orientation \( O \), let

\[
\text{val}_G(O) \triangleq \min_{O^{(i)} \in O} \text{val}_G(O^{(i)})
\]

for reconfiguration sequence \( O = (O^{(i)})_{i \in [0..4]} \), and let

\[
\text{val}_G(O_k \Rightarrow O_l) \triangleq \max_{\sigma=(O_0,...,O_4)} \text{val}_G(O)
\]

for two orientations \( O_k \) and \( O_l \) for \( G \). In Maxmin Nondeterministic Constraint Logic, we wish to maximize \( \text{val}_G(O) \) subject to \( O = (O_k,...,O_l) \). \( \text{Gap}_{c,s} \) Nondeterministic Constraint Logic requests to distinguish whether \( \text{val}_G(O_k \Rightarrow O_l) \geq c \) or \( \text{val}_G(O_2 \Rightarrow O_1) < s \). In what follows, RZH implies PSPACE-hardness of approximation for Maxmin Nondeterministic Constraint Logic.

**Proposition 4.1.** For every \( B \in \mathbb{N} \) and \( \epsilon \in (0, 1) \), there exists a gap-preserving reduction from \( \text{Gap}_{1-\epsilon} \text{E3-SAT}(B) \) Reconfiguration to \( \text{Gap}_{1-\epsilon}(\frac{c}{s}) \) Nondeterministic Constraint Logic.

Our proof makes a modification to the CNF network [24, 25]. To this end, we borrow special nodes that can be simulated by an AND/OR subgraph, including CHOICE, RED–BLUE, FANOUT nodes, and free-edge terminators, which are described below; see also Hearn and Demaine [24, 25] for more details.

- **CHOICE node:** This node has three red edges and is satisfied if at least two edges are directed inward; i.e., only one edge may be directed outward.

- **RED–BLUE node:** This is a degree-2 node incident to one red edge and one blue edge, which acts as transferring a signal between them; i.e., one edge may be directed outward if and only if the other is directed inward.

- **FANOUT node:** This node is equivalent to an AND node from a different interpretation: two red edges may be directed outward if and only if the blue edge is directed inward. Accordingly, a FANOUT node plays a role in splitting a signal.

- **Free-edge terminator:** This is an AND/OR subgraph of constant size used to connect the loose end of an edge. The connected edge is free in a sense that it can be directed inward or outward.

**Reduction.** Given an instance \( (\varphi, \sigma_x, \sigma_t) \) of Maxmin E3-SAT(\( B) \) Reconfiguration, where \( \varphi \) is an E3-CNF formula consisting of \( m \) clauses \( C_1, ..., C_m \) over \( n \) variables \( x_1, ..., x_n \), and \( \sigma_x \) and \( \sigma_t \) satisfy \( \varphi \), we construct an AND/OR graph \( G_\varphi \) as follows. For each variable \( x_i \) of \( \varphi \), we create a CHOICE node, denoted \( v_{x_i} \), called a variable node. Of the three red edges incident

\(^5\) A variant of Nondeterministic Constraint Logic, called configuration-to-edge [24], requires to decide if a specified edge can be eventually reversed by a sequence of edge reversals. From a point of view of approximability, this definition does not seem to make much sense.
Figure 3 An AND/OR graph $G_\varphi$ corresponding to an E3-CNF formula $\varphi = (w \lor x \lor y) \land (w \lor \overline{x} \lor z) \land (x \lor \overline{y} \lor z)$, taken and modified from [25, Figure 5.1]. Here, thicker blue edges have weight 2, thinner red edges have weight 1, and the square node denotes a free edge terminator. The orientation of $G_\varphi$ shown above is given by $O_\psi_0$ such that $\psi_0(w, x, y, z) = (F, T, T, T)$. If $\psi_t$ is defined as $\psi_t(w, x, y, z) = (F, F, T, T)$, we can transform $O_\psi_0$ into $O_\psi_t$; in particular, edges in the subtree rooted at $x$, denoted the gray area, can be made directed downward.

Observe that $G_\varphi$ is satisfiable if and only if $\varphi$ is satisfiable [24, 25]. In fact, a satisfying orientation $O_\sigma$ for $G_\varphi$ can be obtained from any satisfying truth assignment $\sigma$ for $\varphi$. Here, the trick is that if a literal $x_i$ or $\overline{x}_i$ evaluates to $T$ by $\sigma$ and appears in clause $C_j$ of $\varphi$, we can safely orient every edge on the unique path between $v_{x_i}$ and $v_{C_j}$ toward $v_{C_j}$. Constructing $O_\sigma$ from $\sigma_\varphi$ and $O_t$ from $\sigma_t$, according to this procedure, we obtain an instance $(G, O_\varphi, O_t)$ of Maxmin Nondeterministic Constraint Logic, which completes the reduction. The proof of the correctness shown below relies on the fact that for fixed $B \in \mathbb{N}$, the number of nodes $|V(G_\varphi)|$ is proportional to the number of variable nodes $n$ as well as that of clause nodes $m$. 

$C_1 = (w \lor x \lor y)$  
$C_2 = (w \lor \overline{x} \lor z)$  
$C_3 = (x \lor \overline{y} \lor z)$
Proof sketch of Proposition 4.1. We begin with a few remarks on the construction of $G_\varphi$. For each clause $C_j$ that includes $x_i$ or $\overline{x_i}$, there is a unique path between $v_{x_i}$ and $v_{C_j}$ without passing through any other variable or clause node, which takes the following form:

Output signal of a variable node $v_{x_i}$
- a RED–BLUE node
- any number of (a FANOUT node → a RED–BLUE node)
- a clause node $v_{C_j}$.

Therefore, every node except variable and clause nodes is uniquely associated with a particular literal $\ell$ of $\varphi$. Hereafter, the subtree rooted at literal $\ell$ is defined as a subgraph of $G_\varphi$ induced by the unique paths between the corresponding variable node and $v_{C_j}$’s for all clauses $C_j$ including $\ell$ (see also Figure 3).

Since the completeness is almost immediate from the above observation, we next prove the soundness; i.e., $\forall \varphi, \sigma_1 \leadsto \sigma_2 \Rightarrow \text{val}(G_\varphi(\sigma_1 \leadsto \sigma_2)) < 1 - \varepsilon$ implies $\text{val}(G_\varphi(\sigma_2 \leadsto \sigma_2)) < 1 - \Theta\left(\frac{\varepsilon}{2}\right)$. Let $O = \{O^{(0)} = O_k, \ldots, O^{(1)} = O_1\}$ be any reconfiguration sequence for $(G_\varphi, O_k, O_1)$. Construct then a sequence of truth assignments, $\sigma = (\sigma^{(i)})_{i \in [0, 1]}$, such that each $\sigma^{(i)}(x_j)$ for variable $x_j$ is defined to be $T$ if edge $x_j$ is directed outward from $v_{x_j}$, and $\text{edge } G_\varphi$ is directed inward to $v_{x_j}$, and to $F$ otherwise. Since $\sigma$ is a valid reconfiguration sequence for $(\varphi, \sigma_1, \sigma_2)$, it holds that $\text{val}_\varphi(\sigma) < 1 - \varepsilon$; in particular, there exists some $\sigma^{(i)}$ such that $\text{val}_\varphi(\sigma^{(i)}) < 1 - \varepsilon$. However, the number of clause nodes satisfied by $O^{(i)}$ may not be less than $m(1 - \varepsilon)$ because other nodes may be violated in lieu of them (e.g., both of $x_i$ and $\overline{x_i}$ may be directed outward). Thus, we compare $O^{(i)}$ with an orientation $O_{\sigma^{(i)}}$ constructed from $\sigma^{(i)}$ by the procedure described in the reduction paragraph. Note that $O_{\sigma^{(i)}}$ satisfies every non-clause node, while more than $\varepsilon m$ clause nodes are not satisfied. Transforming $O_{\sigma^{(i)}}$ into $O^{(i)}$ by reversing the directions of edges one by one, we can see that each time a non-clause node is violated, we would be able to make at most $B$ clause nodes satisfied. Consequently, we derive

\[
\varepsilon m - B \cdot (\# \text{ non-clause nodes violated by } O^{(i)}) < (\# \text{ clause nodes violated by } O^{(i)})
\]

\[
\Rightarrow (\# \text{ violated nodes by } O^{(i)}) > \frac{\varepsilon m}{B}
\]

\[
\Rightarrow \text{val}_\varphi(O) \leq \text{val}_\varphi(O^{(i)}) < \frac{|V(G_\varphi)| - \varepsilon m}{|V(G_\varphi)|} = 1 - \Theta\left(\frac{\varepsilon}{B}\right),
\]

where we used the fact that $|V(G_\varphi)| = \Theta(m + n) = \Theta(m)$, completing the proof. □

4.2 Reconfiguration Problems on Graphs

Independent Set Reconfiguration. Denote by $\alpha(G)$ the size of a maximum independent set of a graph $G$. For a pair of independent sets $I_s$ and $I_t$ of $G$, Independent Set Reconfiguration asks if there is a sequence of independent sets of $G$ from $I_s$ to $I_t$, each resulting from the previous one by either adding or removing a single vertex of $G$, without going through an independent set of size less than $\min\{|I_s|, |I_t|\} - 1$. For a reconfiguration sequence $I = (I^{(i)})_{i \in [0, 1]}$ of independent sets of a graph $G$, we define $\text{val}_G(I) \triangleq \min_{I^{(i)} \in I} \frac{\alpha(I^{(i)})}{\alpha(I^{(i)}) - 1}$. Here, division by $\alpha(G) - 1$ is derived from the nature that we must remove at least one vertex whenever $|I_s| = |I_t| = \alpha(G)$ and $I_s \neq I_t$. We then define $\text{val}_G(I_s \leadsto I_t) \triangleq \max_{I = (I_s, \ldots, I_t)} \text{val}_G(I)$. In Maxmin Independent Set

---

6 Such a model of reconfiguration is called token addition and removal [28]. We do not consider token jumping [34] or token sliding [24] since they do not change the size of an independent set.
Reconfiguration, we wish to maximize $\text{val}_G(I)$ subject to $I = \langle I_1, \ldots, I_t \rangle$, which is $\text{NP}$-hard to approximate within any constant factor [28]. $\text{Gap}_{c,s}$ Independent Set Reconfiguration requests to distinguish whether $\text{val}_G(I_s \leadsto I_t) \geq c$ or $\text{val}_G(I_s \leadsto I_t) < s$. The proof of the following corollary is based on a Karp reduction due to [24, 25].

**Corollary 4.2 (**) For every $\varepsilon \in (0, 1)$, there exists a gap-preserving reduction from $\text{Gap}_{1,1-\varepsilon}$ Nondeterministic Constraint Logic to $\text{Gap}_{1,1-\text{e}(\varepsilon)}$ Independent Set Reconfiguration.

As an immediate corollary, Maxmin Clique Reconfiguration is $\text{PSPACE}$-hard to approximate under Hypothesis 2.4.

**Corollary 4.3.** For every $\varepsilon \in (0, 1)$, there exists a gap-preserving reduction from $\text{Gap}_{1,1-\varepsilon}$ Nondeterministic Constraint Logic to $\text{Gap}_{1,1-\varepsilon(\varepsilon)}$ Clique Reconfiguration.

**Vertex Cover Reconfiguration.** We conclude this section with a conditional inapproximability result of Minmax Vertex Cover Reconfiguration, which is $2$-factor approximable [28]. Refer to the full version [38] for the formal definition. The proof uses a gap-preserving reduction from Maxmin Independent Set Reconfiguration on restricted graphs due to Bonsma and Cereceda [9].

**Corollary 4.4 (**) For every $\varepsilon \in (0, 1)$, there exists a gap-preserving reduction from $\text{Gap}_{1,1-\varepsilon}$ Nondeterministic Constraint Logic to $\text{Gap}_{1,1-\varepsilon(\varepsilon)}$ Vertex Cover Reconfiguration.

5 Conclusions

We gave a series of gap-preserving reductions to demonstrate $\text{PSPACE}$-hardness of approximation for optimization variants of popular reconfiguration problems assuming Reconfiguration Inapproximability Hypothesis (RIH). An immediate open question is to verify RIH. One approach is to prove it directly, e.g., by using gap amplification of Dinur [19]. Some steps may be more difficult to prove, as we are required to preserve reconfigurability. Another way entails a reduction from some problems already known to be $\text{PSPACE}$-hard to approximate, such as True Quantified Boolean Formula due to Condon, Feigenbaum, Lund, and Shor [15]. We are currently uncertain whether we can “adapt” a Karp reduction from True Quantified Boolean Formula to Nondeterministic Constraint Logic [24, 25].

References


