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 φ and two satisfying truth assignments σ_{s} and σ_{t} for φ, we are asked to determine whether there is a sequence of satisfying truth assignments for φ starting from σ_{s} and ending with σ_{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 φ during transformation from σ_{s} to σ_{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 φ and two satisfying truth assignments σ_{s} and σ_{t} for φ, we seek a reconfiguration sequence of satisfying truth assignments from σ_s to σ_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 φ satisfied by any truth assignment during transformation from σ_s to σ_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 $\mathbf{P} \neq \mathbf{PSPACE}$, but further disproves the existence of a witness (especially a reconfiguration sequence) of *polynomial length* under $\mathbf{NP} \neq \mathbf{PSPACE}$. 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 ψ_s and ψ_t for G, it is* **PSPACE***-hard to distinguish between* YES *instances, in which ψ_s can be transformed into ψ_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 ε-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

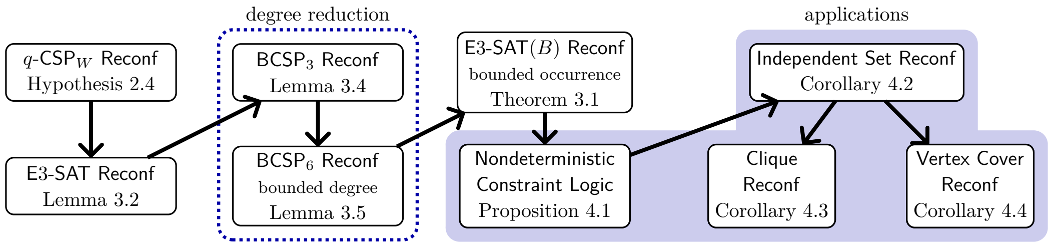

Figure 1 A series of gap-preserving reductions starting from Reconfiguration Inapproximability Hypothesis used in this paper. Here, q-CSP$_W$ Reconf and BCSP$_W$ Reconf denote q-CSP Reconfiguration and Binary CSP Reconfiguration whose alphabet size is restricted to W, respectively; E3-SAT(B) Reconf denotes 3-SAT Reconfiguration in which every clause has exactly 3 literals and each variable occurs in at most B clauses. Note that all reductions preserve the perfect completeness.

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 E3-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 $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, the new one is $\Sigma' = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{ab}, \mathtt{bc}, \mathtt{ca}\}$. Having a vertex to be assigned $\mathtt{ab}$ represents that it has value $\mathtt{a}$ *and* $\mathtt{b}$. 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 ε-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 λ is $\mathcal{O}(\sqrt{d})$; the degree d is determined based on the value of ε.
- 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 E3-SAT Reconfiguration, wherein, but this time, each variable appears in a constant number of clauses, completing the proof of the main result.

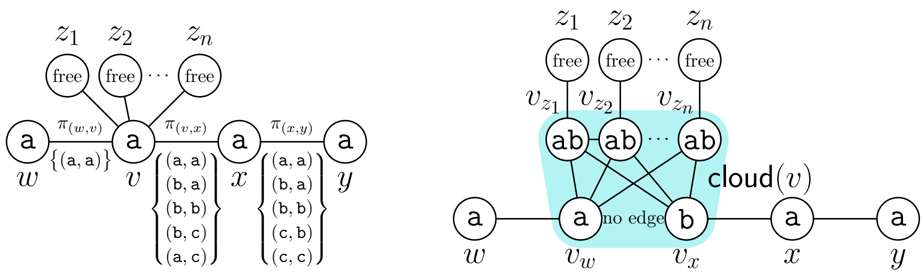

Figure 2 A drawing of Example 3.9. The left side shows an instance G of BCSP Reconfiguration, where we cannot transform from $\psi_{\mathsf{s}}(w,v,x,y) = (\mathsf{a},\mathsf{a},\mathsf{a},\mathsf{a})$ to $\psi_{\mathsf{t}}(w,v,x,y) = (\mathsf{a},\mathsf{a},\mathsf{c},\mathsf{c})$. The right side shows the resulting instance by applying the degree reduction step on v of G. We can now assign conflicting values to v_w and v_x because edge (v_w, v_x) does not exist; in particular, we can transform from $\psi'_{\mathsf{s}}(w,v_w,v_x,x,y) = (\mathsf{a},\mathsf{a},\mathsf{a},\mathsf{a},\mathsf{a})$ into $\psi'_{\mathsf{t}}(w,v_w,v_x,x,y) = (\mathsf{a},\mathsf{a},\mathsf{a},\mathsf{c},\mathsf{c})$.

Once we have established gap-preserving reducibility from RIH to Maxmin E3-SAT Reconfiguration of bounded occurrence, we can apply it to devise conditional **PSPACE**-hardness of approximation for an optimization variant of Nondeterministic Constraint Logic (Proposition 4.1). Nondeterministic Constraint Logic is a **PSPACE**-complete problem proposed by Hearn and Demaine [24, 25] that has been used to prove **PSPACE**-hardness of many games, puzzles, and other reconfiguration problems [5, 9, 11, 30]. We show that under RIH, it is **PSPACE**-hard to distinguish whether an input is a YES instance, or has a property that every transformation must violate more than ε-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 **PSPACE**-hard to approximate, including Independent Set Reconfiguration, Clique Reconfiguration, and 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 $*$ are omitted and can be found in the full version of this paper [38].

Additional Related Work. Other reconfiguration problems whose approximability was analyzed include Set Cover Reconfiguration [28], which is 2-factor approximable, Subset Sum Reconfiguration [27], which admits a PTAS, Shortest Path Reconfiguration [22], and Submodular Reconfiguration [39]. We note that approximability of reconfiguration problems frequently refers to that of *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 *reconfiguration index* [32] or *reconfiguration threshold* [18]. A different type of optimization variant, called *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.

2 Preliminaries

Notations. For two integers $m, n \in \mathbb{N}$ with $m \leqslant n$, let $[n] \triangleq \{1, 2, \ldots, n\}$ and $[m \mathrel{..} n] \triangleq \{m, m+1, \ldots, n-1, n\}$. A *sequence* $\mathcal{E}$ of a finite number of elements $e^{(0)}, e^{(1)}, \ldots, e^{(\ell)}$ is denoted by $\langle e^{(0)}, e^{(1)}, \ldots, e^{(\ell)} \rangle$, 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 "adjacency relation" over a pair of feasible solutions.[1] For two feasible solutions e_{s} and e_{t}, a *reconfiguration sequence from* e_{s} *to* e_{t} is a sequence of feasible solutions, $\mathcal{E} = \langle e^{(0)}, \ldots, e^{(\ell)} \rangle$, starting from e_{s} (i.e., $e^{(0)} = e_{\mathsf{s}}$) and ending with e_{t} (i.e., $e^{(\ell)} = e_{\mathsf{t}}$) such that all consecutive solutions $e^{(i-1)}$ and $e^{(i)}$ are adjacent. In a reconfiguration problem, we wish to decide if there exists a reconfiguration sequence between a pair of feasible solutions.

Boolean Satisfiability. We use the standard terminology and notation of Boolean satisfiability. Truth values are denoted by $\mathtt{T}$ or $\mathtt{F}$. A *Boolean formula* φ consists of variables $x_1, \ldots, x_n$ and the logical operators, AND ($\wedge$), OR ($\vee$), and NOT ($\neg$). A *truth assignment* $\sigma \colon \{x_1, \ldots, x_n\} \to \{\mathtt{T}, \mathtt{F}\}$ for φ is a mapping that assigns a truth value to each variable. A Boolean formula φ is said to be *satisfiable* if there exists some assignment σ such that φ evaluates to $\mathtt{T}$ 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 **PSPACE**-complete k-SAT Reconfiguration problem [23], for a k-CNF formula φ and two satisfying truth assignments σ_{s} and σ_{t} for φ, we are asked to decide if there exists a reconfiguration sequence of satisfying truth assignments for φ from σ_{s} to σ_{t}. 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 φ consisting of m clauses $C_1, \ldots, C_m$ and a truth assignment σ for φ, let $\mathsf{val}_\varphi(\sigma)$ denote the fraction of clauses in φ satisfied by σ; namely,

$$\mathsf{val}_\varphi(\sigma) \triangleq \frac{|\{j \in [m] : \sigma \text{ satisfies } C_j\}|}{m}. \tag{1}$$

For a reconfiguration sequence $\boldsymbol{\sigma} = \langle \sigma^{(0)}, \ldots, \sigma^{(\ell)} \rangle$ of truth assignments, let $\mathsf{val}_\varphi(\boldsymbol{\sigma})$ denote the *minimum* fraction of satisfied clauses over all $\sigma^{(i)}$'s in $\boldsymbol{\sigma}$; i.e.,

$$\mathsf{val}_\varphi(\boldsymbol{\sigma}) \triangleq \min_{\sigma^{(i)} \in \boldsymbol{\sigma}} \mathsf{val}_\varphi(\sigma^{(i)}). \tag{2}$$

Then, Maxmin k-SAT Reconfiguration is defined as a problem of maximizing $\mathsf{val}_\varphi(\boldsymbol{\sigma})$ subject to $\boldsymbol{\sigma} = \langle \sigma_{\mathsf{s}}, \ldots, \sigma_{\mathsf{t}} \rangle$, where σ_{s} and σ_{t} are not necessarily satisfying. For two truth assignments σ_{s} and σ_{t} for φ, let $\mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}})$ denote the maximum value of $\mathsf{val}_\varphi(\boldsymbol{\sigma})$ over all possible reconfiguration sequences $\boldsymbol{\sigma}$ from σ_{s} to σ_{t}; namely,

$$\mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}}) \triangleq \max_{\boldsymbol{\sigma} = \langle \sigma_{\mathsf{s}}, \ldots, \sigma_{\mathsf{t}} \rangle} \mathsf{val}_\varphi(\boldsymbol{\sigma}) = \max_{\boldsymbol{\sigma} = \langle \sigma_{\mathsf{s}}, \ldots, \sigma_{\mathsf{t}} \rangle} \min_{\sigma^{(i)} \in \boldsymbol{\sigma}} \mathsf{val}_\varphi(\sigma^{(i)}). \tag{3}$$

Note that $\mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}}) \leqslant \min\{\mathsf{val}_\varphi(\sigma_{\mathsf{s}}), \mathsf{val}_\varphi(\sigma_{\mathsf{t}})\}$. If $\mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}}) \geqslant \rho$ for some ρ, we can transform σ_{s} into σ_{t} while ensuring that *every* intermediate truth assignment satisfies at least ρ-fraction of the clauses of φ. 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 $0 \leqslant s \leqslant c \leqslant 1$,* $\mathsf{Gap}_{c,s}$ k-SAT $\mathsf{Reconfiguration}$ *requests to determine for a k-CNF formula φ and two truth assignments σ_{s} and σ_{t} for φ, whether* $\mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}}) \geqslant c$ *(the input is a* YES *instance) or* $\mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}}) < s$ *(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 $s \leqslant \mathsf{val}_\varphi(\sigma_{\mathsf{s}} \rightsquigarrow \sigma_{\mathsf{t}}) < c$. The present problem definition does not request an actual reconfiguration sequence. Note that we can assume σ_{s} and σ_{t} to be satisfying ones whenever $c = 1$, and the case of $s = c = 1$ particularly reduces to k-SAT $\mathsf{Reconfiguration}$.

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, Σ is a finite set called the alphabet, and $\Pi = (\pi_e)_{e \in E}$ is a collection of q-ary constraints, where each $\pi_e \subseteq \Sigma^e$ 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 $\psi \colon V \to \Sigma$ that assigns a value of Σ to each vertex of V. We say that ψ *satisfies* hyperedge $e = \{v_1, \ldots, v_q\} \in E$ (or constraint π_e) if $\psi(e) \triangleq (\psi(v_1), \ldots, \psi(v_q)) \in \pi_e$, and ψ *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 $\mathsf{Reconfiguration}$, for a q-ary constraint graph G and two satisfying assignments ψ_{s} and ψ_{t} for G, we are asked to decide if there is a reconfiguration sequence of satisfying assignments for G from ψ_{s} to ψ_{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-$\mathsf{CSP}(\Delta)$ for some $\Delta \in \mathbb{N}$ means that the maximum degree of the constraint graph is bounded by Δ. In an analogous way to the case of Boolean satisfiability, we define $\mathsf{val}_G(\psi) \triangleq \frac{|\{e \in E : \psi \text{ satisfies } e\}|}{|E|}$ for assignment $\psi \colon V \to \Sigma$, $\mathsf{val}_G(\boldsymbol{\psi}) \triangleq \min_{\psi^{(i)} \in \boldsymbol{\psi}} \mathsf{val}_G(\psi^{(i)})$ for reconfiguration sequence $\boldsymbol{\psi} = \langle \psi^{(i)} \rangle_{i \in [0..\ell]}$, and $\mathsf{val}_G(\psi_{\mathsf{s}} \rightsquigarrow \psi_{\mathsf{t}}) \triangleq \max_{\boldsymbol{\psi} = \langle \psi_{\mathsf{s}}, \ldots, \psi_{\mathsf{t}} \rangle} \mathsf{val}_G(\boldsymbol{\psi})$ for $\psi_{\mathsf{s}}, \psi_{\mathsf{t}} \colon V \to \Sigma$. In Maxmin q-CSP $\mathsf{Reconfiguration}$, we wish to maximize $\mathsf{val}_G(\boldsymbol{\psi})$ subject to $\boldsymbol{\psi} = \langle \psi_{\mathsf{s}}, \ldots, \psi_{\mathsf{t}} \rangle$. The corresponding gap version is defined as follows.

▶ **Problem 2.3.** *For every $q \in \mathbb{N}$ and $0 \leqslant s \leqslant c \leqslant 1$,* $\mathsf{Gap}_{c,s}$ q-CSP $\mathsf{Reconfiguration}$ *requests to determine for a q-ary constraint graph G and two (not necessarily satisfying) assignments ψ_{s} and ψ_{t} for G, whether* $\mathsf{val}_G(\psi_{\mathsf{s}} \rightsquigarrow \psi_{\mathsf{t}}) \geqslant c$ *or* $\mathsf{val}_G(\psi_{\mathsf{s}} \rightsquigarrow \psi_{\mathsf{t}}) < s$.

Reconfiguration Inapproximability Hypothesis. We now present a formal description of our working hypothesis, which serves as a starting point for **PSPACE**-hardness of approximation.

▶ **Hypothesis 2.4** (Reconfiguration Inapproximability Hypothesis, RIH). *There exist universal constants $q, W \in \mathbb{N}$, $\varepsilon \in (0, 1)$ such that* $\mathsf{Gap}_{1,1-\varepsilon}$ q-CSP_W $\mathsf{Reconfiguration}$ *is* **PSPACE**-*hard.*

3 Hardness of Approximation for Maxmin E3-SAT(B) Reconfiguration

In this section, we prove the main result of this paper; that is, Maxmin E3-SAT $\mathsf{Reconfiguration}$ of bounded occurrence is **PSPACE**-hard to approximate under RIH.

▶ **Theorem 3.1.** *Under Hypothesis 2.4, there exist universal constants $\varepsilon \in (0,1)$ and $B \in \mathbb{N}$ such that* $\mathsf{Gap}_{1,1-\varepsilon}$ 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 $q, W \geqslant 2$ and $\varepsilon \in (0,1)$, there exists a gap-preserving reduction from* $\mathsf{Gap}_{1,1-\varepsilon}$ q-CSP$_W$ Reconfiguration *to* $\mathsf{Gap}_{1,1-\frac{\varepsilon}{W^q \cdot 2^{qW}(qW-2)}}$ E3-SAT Reconfiguration*. Moreover, if the maximum degree of the constraint graph in the former problem is Δ, then the number of occurrences of each variable in the latter problem is bounded by $W^q \cdot 2^{qW}\Delta$.*

The proof of Lemma 3.2 consists of a reduction from Maxmin q-CSP$_W$ Reconfiguration to Maxmin Ek-SAT Reconfiguration, where the clause size k depends solely on q and W, and that from Maxmin Ek-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 $\mathsf{enc}\colon \{\mathtt{T},\mathtt{F}\}^\Sigma \to \Sigma$ of a binary string $\mathbf{s} \in \{\mathtt{T},\mathtt{F}\}^\Sigma$ to Σ defined as follows:

$$\mathsf{enc}(\mathbf{s}) \triangleq \begin{cases} 1 & \text{if } s_\alpha = \mathtt{F} \text{ for all } \alpha \in \Sigma, \\ \alpha & \text{if } s_\alpha = \mathtt{T} \text{ and } s_\beta = \mathtt{F} \text{ for all } \beta > \alpha. \end{cases} \tag{4}$$

enc exhibits the following property concerning reconfigurability:

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

We use enc to encode each q-tuple of unacceptable values $(\alpha_1, \ldots, \alpha_q) \in \Sigma^e \setminus \pi_e$ 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 Järvisalo and Niemelä [33].

▶ **Lemma 3.4** ($*$)**.** *For every $\varepsilon \in (0,1)$, there exists a gap-preserving reduction from* $\mathsf{Gap}_{1,1-\varepsilon}$ E3-SAT Reconfiguration *to* $\mathsf{Gap}_{1,1-\frac{\varepsilon}{3}}$ BCSP$_3$ Reconfiguration*. 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 E3-SAT Reconfiguration to Maxmin BCSP_3 Reconfiguration. Let $(\varphi, \sigma_{\mathsf{s}}, \sigma_{\mathsf{t}})$ be an instance of Maxmin E3-SAT Reconfiguration, where φ is a 3-CNF formula consisting of m clauses $C_1, \ldots, C_m$ over n variables $x_1, \ldots, x_n$ and σ_{s} and σ_{t} satisfy φ. 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 Σ_v to denote the alphabet assigned to vertex $v \in V$. We then express $\Sigma_{x_i} \triangleq \{\mathtt{T}, \mathtt{F}\}$ for each variable x_i, and $\Sigma_{C_j} \triangleq \{\ell_1, \ell_2, \ell_3\}$ for each clause $C_j = (\ell_1 \vee \ell_2 \vee \ell_3)$. For each edge $(x_i, C_j) \in E$ with $C_j = (\ell_1 \vee \ell_2 \vee \ell_3)$, the constraint $\pi_{(x_i, C_j)} \subseteq \Sigma_{x_i} \times \Sigma_{C_j}$ is defined as follows:

$$\pi_{(x_i,C_j)} \triangleq \begin{cases} (\Sigma_{x_i} \times \Sigma_{C_j}) \setminus \{(\mathtt{F}, x_i)\} & \text{if } x_i \text{ appears in } C_j, \\ (\Sigma_{x_i} \times \Sigma_{C_j}) \setminus \{(\mathtt{T}, \overline{x_i})\} & \text{if } \overline{x_i} \text{ appears in } C_j. \end{cases} \tag{5}$$

For an assignment ψ for G, $\psi(x_i)$ represents the truth value assigned to x_i, and $\psi(C_j)$ specifies which literal should evaluate to $\mathtt{T}$. Note that $|V| = n + m$, $|E| = 3m$, and the maximum degree of (V, E) is $\max\{B, 3\}$. For a satisfying truth assignment σ for φ, 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 ℓ_i appears in C_j and evaluates to $\mathtt{T}$ by σ.[2] Obviously, ψ_σ satisfies G. Constructing ψ_{s} from σ_{s} and ψ_{t} from σ_{t} according to this procedure, we obtain an instance $(G, \psi_{\mathsf{s}}, \psi_{\mathsf{t}})$ 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* $\mathsf{Gap}_{1,1-\varepsilon}$ BCSP_3 Reconfiguration *to* $\mathsf{Gap}_{1,1-\overline{\varepsilon}}$ $\mathsf{BCSP}_6(\Delta)$ Reconfiguration*, where $\overline{\varepsilon} \in (0, 1)$ and $\Delta \in \mathbb{N}$ are some computable functions dependent only on the value of ε. In particular, the constraint graph in the latter problem is of bounded degree for fixed ε.*

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, λ)-expander graph is a d-regular graph G on n vertices such that* $\max\{\lambda_2(G), |\lambda_n(G)|\} \leqslant \lambda < d$*, where $\lambda_i(G)$ is the i-th largest (real-valued) eigenvalue of the adjacency matrix of G.*

An (n, d, λ)-expander graph is called *Ramanujan* if $\lambda \leqslant 2\sqrt{d-1}$. 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 $d \geqslant 3$, $\varepsilon > 0$, and all sufficiently large $n \geqslant n_0(d, \varepsilon)$, where nd is even, there is a deterministic $n^{\mathcal{O}(1)}$-time algorithm that outputs an (n, d, λ)-expander graph with $\lambda \leqslant 2\sqrt{d-1} + \varepsilon$.*

In this paper, we rely only on the special case of $\varepsilon = 2\sqrt{d} - 2\sqrt{d-1}$ so that $\lambda \leqslant 2\sqrt{d}$; 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 $2\sqrt{d} - 2\sqrt{d-1} \geqslant \frac{1}{\sqrt{d}}$. 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 ℓ_i always exists as σ 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, λ)-expander graph. Then, for any two sets S, T of vertices, it holds that*

$$\left| e(S,T) - \frac{d|S| \cdot |T|}{n} \right| \leqslant \lambda \sqrt{|S| \cdot |T|}, \tag{6}$$

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 ε. Redefine $\varepsilon \leftarrow \lceil 1/\varepsilon \rceil^{-1}$ so that $1/\varepsilon$ is a positive integer, which does not increase the value of ε; i.e., $\mathsf{val}_G(\psi_\mathsf{s} \rightsquigarrow \psi_\mathsf{t}) < 1 - \varepsilon$ implies $\mathsf{val}_G(\psi_\mathsf{s} \rightsquigarrow \psi_\mathsf{t}) < 1 - \lceil 1/\varepsilon \rceil^{-1}$. Let $(G = (V, E, \Sigma, \Pi = (\pi_e)_{e \in E}), \psi_\mathsf{s}, \psi_\mathsf{t})$ be an instance of $\mathsf{Gap}_{1,1-\varepsilon}$ BCSP_3 $\mathsf{Reconfiguration}$, where ψ_s and ψ_t satisfy G. For the sake of notation, we denote $\Sigma \triangleq \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. We then create a new instance $(G' = (V', E', \Sigma', \Pi' = (\pi'_{e'})_{e' \in E'}), \psi'_\mathsf{s}, \psi'_\mathsf{t})$ of Maxmin BCSP_6 $\mathsf{Reconfiguration}$, 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

$$\mathsf{cloud}(v) \triangleq \Big\{ (v, e) : e \in E \text{ is incident on } v \Big\}. \tag{7}$$

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

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

$$E' \triangleq \bigcup_{v \in V} E(X_v) \cup \Big\{ ((v,e),(w,e)) : e = (v,w) \in E \Big\}. \tag{8}$$

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

Constraints: The constraint $\pi'_{e'} \subseteq \Sigma'^{e'}$ 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]

$$\pi'_{e'} \triangleq \Big\{ (\alpha, \beta) : \alpha, \beta \in \Sigma', \alpha \subseteq \beta \text{ or } \beta \subseteq \alpha \Big\}. \tag{9}$$

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

$$\pi'_{e'} \triangleq \Big\{ (\alpha, \beta) : \alpha \times \beta \subseteq \pi_e \Big\}. \tag{10}$$

Although the underlying graph (V', E') is the same as that in [19] (except for the use of Theorem 3.7), the definitions of Σ' and Π' 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 $\psi(v') = \mathtt{ab}$ represents that v' has values $\mathtt{a}$ *and* $\mathtt{b}$ simultaneously;

[3] Eq. (9) can be explicitly expanded as $\pi'_{e'} = \{(\mathtt{a},\mathtt{a}), (\mathtt{b},\mathtt{b}), (\mathtt{c},\mathtt{c}), (\mathtt{ab},\mathtt{a}), (\mathtt{ab},\mathtt{b}), (\mathtt{bc},\mathtt{b}), (\mathtt{bc},\mathtt{c}), (\mathtt{ca},\mathtt{c}), (\mathtt{ca},\mathtt{a}), (\mathtt{a},\mathtt{ab}), (\mathtt{b},\mathtt{ab}), (\mathtt{b},\mathtt{bc}), (\mathtt{c},\mathtt{bc}), (\mathtt{c},\mathtt{ca}), (\mathtt{a},\mathtt{ca}), (\mathtt{ab},\mathtt{ab}), (\mathtt{bc},\mathtt{bc}), (\mathtt{ca},\mathtt{ca})\}$.

e.g., if $\psi'(v') = \mathtt{ab}$ and $\psi'(w') = \mathtt{c}$ for some $v' \in \mathsf{cloud}(v)$ and $w' \in \mathsf{cloud}(w)$ with $v \neq w$, then ψ' satisfies $\pi'_{(v',w')}$ if both $(\mathtt{a},\mathtt{b})$ and $(\mathtt{a},\mathtt{c})$ are in $\pi_{(v,w)}$ owing to Eq. (10). Construct two assignments $\psi'_{\mathsf{s}}\colon V' \to \Sigma'$ from ψ_{s} and $\psi'_{\mathsf{t}}\colon V' \to \Sigma'$ from ψ_{t} such that $\psi'_{\mathsf{s}}(v,e) \triangleq \{\psi_{\mathsf{s}}(v)\}$ and $\psi'_{\mathsf{t}}(v,e) \triangleq \{\psi_{\mathsf{t}}(v)\}$ for all $(v,e) \in V'$. Observe that both ψ'_{s} and ψ'_{t} satisfy G', thereby completing the reduction. Note that $|V'| = 2|E|$, $|E'| \leqslant n_0(d_0) \cdot |E|$, $|\Sigma'| = 6$, and the maximum degree of G' is $\Delta \leqslant n_0(d_0)$, which is constant for fixed ε.

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 $G = (V, E, \Sigma, \Pi = (\pi_e)_{e\in E})$ such that $V \triangleq \{w, v, x, y, z_1, \ldots, z_n\}$ for some large integer n, $E \triangleq \{(w,v),(v,x),(x,y),(v,z_1),\ldots,(v,z_n)\}$, $\Sigma \triangleq \{\mathtt{a},\mathtt{b},\mathtt{c}\}$, and each π_e is defined as follows: $\pi_{(w,v)} \triangleq \{(\mathtt{a},\mathtt{a})\}$, $\pi_{(v,x)} \triangleq \{(\mathtt{a},\mathtt{a}),(\mathtt{b},\mathtt{a}),(\mathtt{b},\mathtt{b}),(\mathtt{b},\mathtt{c}),(\mathtt{a},\mathtt{c})\}$, $\pi_{(x,y)} \triangleq \{(\mathtt{a},\mathtt{a}),(\mathtt{b},\mathtt{a}),(\mathtt{b},\mathtt{b}),(\mathtt{c},\mathtt{b}),(\mathtt{c},\mathtt{c})\}$, and $\pi_{(v,z_1)} = \cdots = \pi_{(v,z_n)} \triangleq \Sigma \times \Sigma$. Define $\psi_{\mathsf{s}}, \psi_{\mathsf{t}}\colon V \to \Sigma$ such that $\psi_{\mathsf{s}}(u) \triangleq \mathtt{a}$ for all $u \in V$, $\psi_{\mathsf{t}}(x) = \psi_{\mathsf{t}}(y) \triangleq \mathtt{c}$, and $\psi_{\mathsf{t}}(u) \triangleq \mathtt{a}$ for all other u. Then, it is *impossible* to transform ψ_{s} into ψ_{t} without any constraint violation: As the values of w and v cannot change from $\mathtt{a}$, we can only change the value of x to $\mathtt{c}$, violating (x,y). In particular, $\mathsf{val}_G(\psi_{\mathsf{s}} \rightsquigarrow \psi_{\mathsf{t}}) < 1$.

Consider applying our reduction to v *only.* Create $\mathsf{cloud}(v) \triangleq \{v_w, v_x, v_{z_1}, \ldots, v_{z_n}\}$ with the shorthand notation $v_u \triangleq (v,(v,u))$, and let X_v be an expander graph on $\mathsf{cloud}(v)$. We then construct a new constraint graph $G' = (V', E', \Sigma', \Pi' = (\pi'_e)_{e\in E'})$, where $V' \triangleq \{w, x, y, z_1, \ldots, z_n\} \cup \mathsf{cloud}(v)$, $E' \triangleq E(X_v) \cup \{(w, v_w), (v_x, x), (x, y), (v_{z_1}, z_1), \ldots, (v_{z_n}, z_n)\}$, $\Sigma' \triangleq \{\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{ab},\mathtt{bc},\mathtt{ca}\}$, and each π'_e is defined according to Eqs. (9) and (10). Construct $\psi'_{\mathsf{s}}, \psi'_{\mathsf{t}}\colon V' \to \Sigma'$ from $\psi_{\mathsf{s}}, \psi_{\mathsf{t}}$ using the procedure described above. Suppose now "by chance" $(v_w, v_x) \notin E(X_v)$. The crucial observation is that we can assign $\mathtt{a}$ to v_w, $\mathtt{b}$ to v_x, and $\mathtt{ab}$ to $v_{z_1}, \ldots, v_{z_n}$ to do some "cheating." Consequently, ψ'_{s} can be transformed into ψ'_{t} without sacrificing any constraint: Assign $\mathtt{ab}$ to $v_{z_1}, \ldots, v_{z_n}$ in arbitrary order; assign $\mathtt{b}$ to v_x, x, and y in this order; assign $\mathtt{c}$ to x and y in this order; assign $\mathtt{a}$ to v_x; assign $\mathtt{a}$ to $v_{z_1}, \ldots, v_{z_n}$ in arbitrary order. In particular, $\mathsf{val}_{G'}(\psi'_{\mathsf{s}} \rightsquigarrow \psi'_{\mathsf{t}}) = 1$.

Correctness. The proof of the completeness is immediate from the definition of Σ' and Π'.

▶ **Lemma 3.10** (∗). *If* $\mathsf{val}_G(\psi_{\mathsf{s}} \rightsquigarrow \psi_{\mathsf{t}}) = 1$, *then* $\mathsf{val}_{G'}(\psi'_{\mathsf{s}} \rightsquigarrow \psi'_{\mathsf{t}}) = 1$.

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

▶ **Lemma 3.11.** *If* $\mathsf{val}_G(\psi_{\mathsf{s}} \rightsquigarrow \psi_{\mathsf{t}}) < 1 - \varepsilon$, *then* $\mathsf{val}_{G'}(\psi'_{\mathsf{s}} \rightsquigarrow \psi'_{\mathsf{t}}) < 1 - \overline{\varepsilon}$, *where* $\overline{\varepsilon} = \overline{\varepsilon}(\varepsilon)$ *is some computable function such that* $\overline{\varepsilon} \in (0,1)$ *if* $\varepsilon \in (0,1)$.

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

$$\mathsf{PLR}(\psi')(v) \triangleq \operatorname*{argmax}_{\alpha\in\Sigma} \left|\left\{v' \in \mathsf{cloud}(v) : \alpha \in \psi'(v')\right\}\right|, \tag{11}$$

where ties are arbitrarily broken according to any prefixed ordering over Σ (e.g., $\mathtt{a} \prec \mathtt{b} \prec \mathtt{c}$). Suppose we have a reconfiguration sequence $\boldsymbol{\psi}' = \langle \psi'^{(0)} = \psi'_{\mathsf{s}}, \ldots, \psi'^{(\ell)} = \psi'_{\mathsf{t}}\rangle$ for $(G', \psi'_{\mathsf{s}}, \psi'_{\mathsf{t}})$ with the maximum value. Construct then a sequence of assignments, $\boldsymbol{\psi} \triangleq \langle \psi^{(i)} \rangle_{i\in[0..\ell]}$, such that $\psi^{(i)} \triangleq \mathsf{PLR}(\psi'^{(i)})$ for all $i \in [0 \,..\, \ell]$. Observe that $\boldsymbol{\psi}$ is a valid reconfiguration sequence for $(G, \psi_{\mathsf{s}}, \psi_{\mathsf{t}})$, and we thus must have $\mathsf{val}_G(\boldsymbol{\psi}) < 1 - \varepsilon$; in particular, there exists some $\psi'^{(i)}$ such that $\mathsf{val}_G(\mathsf{PLR}(\psi'^{(i)})) = \mathsf{val}_G(\psi^{(i)}) < 1 - \varepsilon$. We would like to show that $\mathsf{val}_{G'}(\psi'^{(i)}) < 1 - \overline{\varepsilon}$ for some $\overline{\varepsilon} \in (0, 1)$. 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 $\mathsf{cloud}(v)$ whose values *disagree* with the plurality vote $\psi(v)$; namely,

$$D_v \triangleq \left\{ v' \in \mathsf{cloud}(v) : \psi(v) \notin \psi'(v') \right\}. \tag{12}$$

Consider any edge $e = (v, w) \in E$ violated by ψ (i.e., $(\psi(v), \psi(w)) \notin \pi_e$), and let $e' = (v', w') \in E'$ be a unique (inter-cloud) edge such that $v' \in \mathsf{cloud}(v)$ and $w' \in \mathsf{cloud}(w)$. By definition of $\pi'_{e'}$, either of the following must hold: (1) edge e' is violated by ψ' (i.e., $(\psi'(v'), \psi'(w')) \notin \pi'_{e'}$), or (2) $\psi(v) \notin \psi'(v')$ (i.e., $v' \in D_v$) or $\psi(w) \notin \psi'(w')$ (i.e., $w' \in D_w$). Consequently, the number of edges in E violated by ψ is bounded by the sum of the number of edges in E' violated by ψ' and the number of vertices in V' who disagree with the plurality vote; namely,

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

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 ψ' is more than $\frac{\varepsilon}{2}|E|$, then we have done because

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

We now consider the case that $\sum_{v \in V} |D_v| > \frac{\varepsilon}{2}|E|$. Define $x_v \triangleq |D_v|/d_G(v)$ for each $v \in V$, which is the fraction of vertices in $\mathsf{cloud}(v)$ who disagree with $\psi(v)$. We also define $\delta \triangleq \frac{\varepsilon}{8}$. We first show that the total size of $|D_v|$ *conditioned on* $x_v \geqslant \delta$ is $\Theta(\varepsilon|E|)$.

▷ Claim 3.12 (∗). $\displaystyle\sum_{v \in V : x_v \geqslant \delta} |D_v| > \frac{\varepsilon}{4}|E|$, where $\delta = \frac{\varepsilon}{8}$.

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

▷ Claim 3.13 (∗). For each vertex v of V, there exists a pair of disjoint subsets S, T of $\mathsf{cloud}(v)$ such that $|S| \geqslant \frac{|D_v|}{3}$, $|T| \geqslant \frac{|D_v|}{3}$, and ψ' violates all constraints between S and T.

Consider a vertex $v \in V$ such that $x_v \geqslant \delta$; that is, at least δ-fraction of vertices in $\mathsf{cloud}(v)$ disagree with $\psi(v)$. Letting S and T be two disjoint subsets of $\mathsf{cloud}(v)$ 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 $d_0 \triangleq \left(\frac{12}{\delta}\right)^2 = \frac{9216}{\varepsilon^2}$, which is a positive *even* integer (so that Theorem 3.7 is applicable) and depends only on the value of ε. Suppose first $d_G(v) \geqslant n_0(d_0)$; i.e., X_v is an expander graph.

▶ **Lemma 3.14.** *For a vertex v of V such that $x_v \geqslant \delta$ and $d_G(v) \geqslant n_0(d_0)$, let S and T be a pair of disjoint subsets of $\mathsf{cloud}(v)$ obtained by Claim 3.13. Then, $e(S, T) \geqslant \frac{8}{\delta}|D_v|$.*

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

$$e(S, T) \geqslant \frac{d_0|S| \cdot |T|}{d_G(v)} - \lambda\sqrt{|S| \cdot |T|} \geqslant \underbrace{\frac{|S| \cdot |T|}{d_G(v)} \left(\frac{12}{\delta}\right)^2 - \frac{2 \cdot 12}{\delta}\sqrt{|S| \cdot |T|}}_{= \underline{e}(S,T)}. \tag{15}$$

It is easy to see that $\underline{e}(S,T)$ is monotonically increasing in $\sqrt{|S|\cdot|T|}$ when $\sqrt{|S|\cdot|T|} > \frac{\delta}{12}d_G(v)$. Observing that $\sqrt{|S|\cdot|T|} \geqslant \frac{\delta}{3}d_G(v)$ since $|S| \geqslant \frac{x_v}{3}d_G(v)$, $|T| \geqslant \frac{x_v}{3}d_G(v)$, and $x_v \geqslant \delta$ by assumption, we derive

$$\begin{aligned} e(S,T) \geqslant \underline{e}(S,T) &\geqslant \frac{1}{d_G(v)}\left(\frac{x_v\cdot d_G(v)}{3}\right)^2\left(\frac{12}{\delta}\right)^2 - \frac{2\cdot 12}{\delta}\frac{x_v\cdot d_G(v)}{3} \\ &\underbrace{\geqslant}_{\text{use } x_v\geqslant\delta} \frac{1}{d_G(v)}\left(\frac{x_v\cdot d_G(v)}{3}\right)\left(\frac{\delta\cdot d_G(v)}{3}\right)\left(\frac{12}{\delta}\right)^2 - \frac{2\cdot 12}{\delta}\frac{x_v\cdot d_G(v)}{3} = \frac{8}{\delta}|D_v|. \blacktriangleleft \end{aligned}$$

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

$$e(S,T) = |S|\cdot|T| \underbrace{\geqslant}_{\text{Claim 3.13}} \left(\frac{|D_v|}{3}\right)^2 = \frac{x_v\cdot d_G(v)}{9}|D_v| \underbrace{\geqslant}_{\text{use } d_G(v)\geqslant 1} \frac{\delta}{9}|D_v|. \tag{16}$$

By Lemma 3.14 and Eq. (16), for every vertex $v\in V$ such that $x_v \geqslant \delta$, the number of violated intra-cloud edges within X_v is at least $\min\{\frac{8}{\delta},\frac{\delta}{9}\}|D_v| \geqslant \frac{\delta}{9}|D_v|$. Simple calculation using Claim 3.12 bounds the total number of edges violated from below as

$$\sum_{v\in V}(\#\text{ violated edges in } X_v) \geqslant \sum_{v:x_v\geqslant\delta}\frac{\delta}{9}|D_v| \underbrace{>}_{\text{Claim 3.12}} \frac{\varepsilon}{72}\frac{\varepsilon}{4}|E| \geqslant \frac{\varepsilon^2\cdot|E'|}{288\cdot n_0(d_0)}. \tag{17}$$

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

$$\mathsf{val}_{G'}(\boldsymbol{\psi}') \leqslant \mathsf{val}_{G'}(\psi') < \max\left\{1-\frac{\varepsilon}{2\cdot n_0(d_0)}, 1-\frac{\varepsilon^2}{288\cdot n_0(d_0)}\right\} = 1-\frac{\varepsilon^2}{288\cdot n_0\left(\frac{9216}{\varepsilon^2}\right)}.$$

Setting $\overline{\varepsilon} \triangleq \frac{\varepsilon^2}{288\cdot n_0\left(\frac{9216}{\varepsilon^2}\right)}$ accomplishes the proof of Lemma 3.11 and thus Lemma 3.5. ◀

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:[4]

- 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_{s} and O_{t} for G, we are asked if O_{s} 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 $\mathsf{val}_G(\cdot)$ analogously: Let $\mathsf{val}_G(O)$ denote the fraction of nodes satisfied by orientation O, let

$$\mathsf{val}_G(\mathcal{O}) \triangleq \min_{O^{(i)} \in \mathcal{O}} \mathsf{val}_G(O^{(i)}) \tag{18}$$

for reconfiguration sequence $\mathcal{O} = \langle O^{(i)} \rangle_{i \in [0..\ell]}$, and let

$$\mathsf{val}_G(O_{\mathsf{s}} \rightsquigarrow O_{\mathsf{t}}) \triangleq \max_{\mathcal{O} = \langle O_{\mathsf{s}}, \ldots, O_{\mathsf{t}} \rangle} \mathsf{val}_G(\mathcal{O}) \tag{19}$$

for two orientations O_{s} and O_{t} for G. In Maxmin Nondeterministic Constraint Logic, we wish to maximize $\mathsf{val}_G(\mathcal{O})$ subject to $\mathcal{O} = \langle O_{\mathsf{s}}, \ldots, O_{\mathsf{t}} \rangle$. $\mathsf{Gap}_{c,s}$ Nondeterministic Constraint Logic requests to distinguish whether $\mathsf{val}_G(O_{\mathsf{s}} \rightsquigarrow O_{\mathsf{t}}) \geqslant c$ or $\mathsf{val}_G(O_{\mathsf{s}} \rightsquigarrow O_{\mathsf{t}}) < s$. In what follows, RIH implies **PSPACE**-hardness of approximation for Maxmin Nondeterministic Constraint Logic.

▶ **Proposition 4.1.** *For every $B \in \mathbb{N}$ and $\varepsilon \in (0,1)$, there exists a gap-preserving reduction from* $\mathsf{Gap}_{1,1-\varepsilon}$ E3-SAT(B) Reconfiguration *to* $\mathsf{Gap}_{1,1-\Theta(\frac{\varepsilon}{B})}$ 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 $(\varphi, \sigma_{\mathsf{s}}, \sigma_{\mathsf{t}})$ of Maxmin E3-SAT(B) Reconfiguration, where φ is an E3-CNF formula consisting of m clauses $C_1, \ldots, C_m$ over n variables $x_1, \ldots, x_n$, and σ_{s} and σ_{t} satisfy φ, we construct an AND/OR graph G_φ as follows. For each variable x_i of φ, 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.

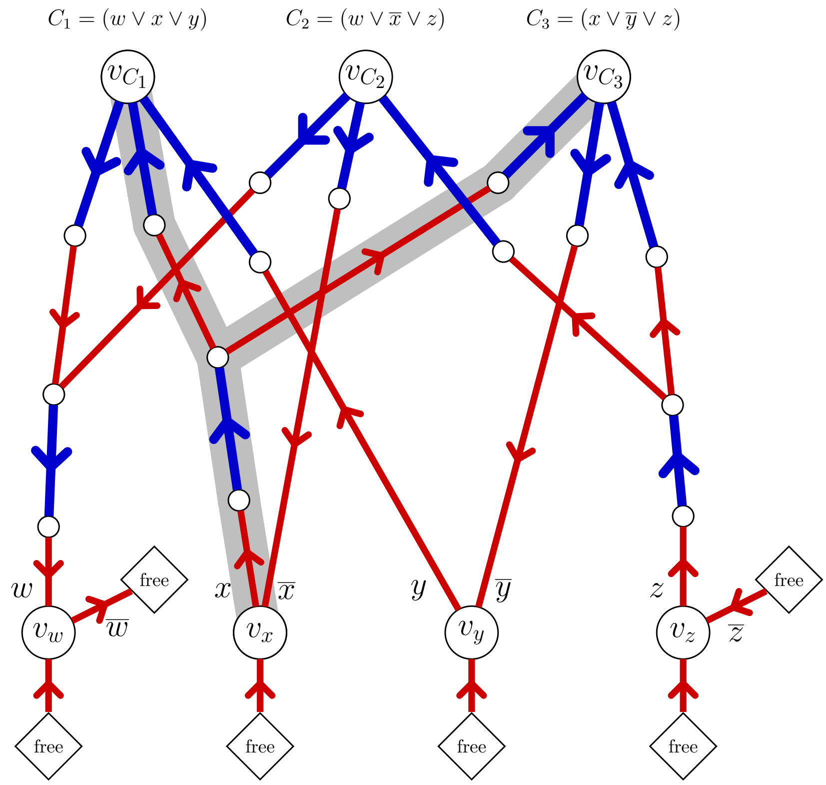

Figure 3 An AND/OR graph G_φ corresponding to an E3-CNF formula $\varphi = (w \vee x \vee y) \wedge (w \vee \overline{x} \vee z) \wedge (x \vee \overline{y} \vee z)$, taken and modified from [25, Figure 5.1]. Here, thicker blue edges have weight 2, thinner red edges have weight 1, and the square node denotes a free edge terminator. The orientation of G_φ shown above is given by O_{ψ_s} such that $\psi_\mathsf{s}(w,x,y,z) = (\mathtt{F},\mathtt{T},\mathtt{T},\mathtt{T})$. If ψ_t is defined as $\psi_\mathsf{t}(w,x,y,z) = (\mathtt{F},\mathtt{F},\mathtt{T},\mathtt{T})$, we can transform O_{ψ_s} into O_{ψ_t}; in particular, edges in the subtree rooted at x, denoted the gray area, can be made directed downward.

to v_{x_i}, one is connected to a free edge terminator, whereas the other two are labeled "x_i" and "$\overline{x_i}$". Thus, either of x_i or $\overline{x_i}$ can be directed outward without breaking v_{x_i}. For each clause C_j of φ, we create an OR node, denoted v_{C_j}, called a *clause node*. The output signals of variable nodes are sent toward the corresponding clause nodes. Specifically, if literal ℓ appears in multiple clauses of φ, we first make a desired number of copied signals of edge ℓ using RED–BLUE and FANOUT nodes; if ℓ does not appear in any clause, we connect the edge ℓ to a free edge terminator. Then, for each clause $C_j = (\ell_1 \vee \ell_2 \vee \ell_3)$ of φ, the clause node v_{C_j} is connected to three edges corresponding to the (copied) signals of ℓ_1, ℓ_2, ℓ_3. This completes the construction of G_φ. See Figure 3 for an example.

Observe that G_φ is satisfiable if and only if φ is satisfiable [24, 25]. In fact, a satisfying orientation O_σ for G_φ can be obtained from any satisfying truth assignment σ for φ. Here, the trick is that if a literal x_i or $\overline{x_i}$ evaluates to $\mathtt{T}$ by σ and appears in clause C_j, we can safely orient *every* edge on the unique path between v_{x_i} and v_{C_j} toward v_{C_j}. Constructing O_s from σ_s and O_t from σ_t according to this procedure, we obtain an instance $(G, O_\mathsf{s}, O_\mathsf{t})$ 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.

Proof sketch of Proposition 4.1. We begin with a few remarks on the construction of G_φ. 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}
$\to$ a RED–BLUE node
$\to$ any number of (a FANOUT node $\to$ a RED–BLUE node)
$\to$ a clause node v_{C_j}.

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

Since the completeness is almost immediate from the above observation, we next prove the soundness; i.e., $\mathsf{val}_\varphi(\sigma_\mathsf{s} \rightsquigarrow \sigma_\mathsf{t}) < 1-\varepsilon$ implies $\mathsf{val}_{G_\varphi}(O_\mathsf{s} \rightsquigarrow O_\mathsf{t}) < 1-\Theta(\frac{\varepsilon}{B})$. Let $\mathcal{O} = \langle O^{(0)} = O_\mathsf{s}, \ldots, O^{(\ell)} = O_\mathsf{t}\rangle$ be any reconfiguration sequence for $(G_\varphi, O_\mathsf{s}, O_\mathsf{t})$. Construct then a sequence of truth assignments, $\boldsymbol{\sigma} = \langle \sigma^{(i)}\rangle_{i\in[0..\ell]}$, such that each $\sigma^{(i)}(x_j)$ for variable x_j is defined to be `T` if edge x_i is directed outward from v_{x_i} and edge $\overline{x_i}$ is directed inward to v_{x_i}, and to be `F` otherwise. Since $\boldsymbol{\sigma}$ is a valid reconfiguration sequence for $(\varphi, \sigma_\mathsf{s}, \sigma_\mathsf{t})$, it holds that $\mathsf{val}_\varphi(\boldsymbol{\sigma}) < 1-\varepsilon$; in particular, there exists some $\sigma^{(i)}$ such that $\mathsf{val}_\varphi(\sigma^{(i)}) < 1-\varepsilon$. 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 εm 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

$$\begin{aligned}
&\varepsilon m - B\cdot(\#\text{ non-clause nodes violated by } O^{(i)}) < (\#\text{ clause nodes violated by } O^{(i)})\\
&\implies (\#\text{ violated nodes by } O^{(i)}) > \frac{\varepsilon}{B}m\\
&\implies \mathsf{val}_{G_\varphi}(\mathcal{O}) \leqslant \mathsf{val}_{G_\varphi}(O^{(i)}) < \frac{|V(G_\varphi)| - \frac{\varepsilon}{B}m}{|V(G_\varphi)|} = 1-\Theta\left(\frac{\varepsilon}{B}\right),
\end{aligned} \tag{20}$$

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,[6] without going through an independent set of size less than $\min\{|I_\mathsf{s}|, |I_\mathsf{t}|\}-1$. For a reconfiguration sequence $\mathcal{I} = \langle I^{(i)}\rangle_{i\in[0..\ell]}$ of independent sets of a graph G, we define $\mathsf{val}_G(\mathcal{I}) \triangleq \min_{I^{(i)}\in\mathcal{I}} \frac{|I^{(i)}|}{\alpha(G)-1}$. Here, division by $\alpha(G)-1$ is derived from the nature that we must remove at least one vertex whenever $|I_\mathsf{s}| = |I_\mathsf{t}| = \alpha(G)$ and $I_\mathsf{s} \neq I_\mathsf{t}$. We then define $\mathsf{val}_G(I_\mathsf{s} \rightsquigarrow I_\mathsf{t}) \triangleq \max_{\mathcal{I}=\langle I_\mathsf{s},\ldots,I_\mathsf{t}\rangle} \mathsf{val}_G(\mathcal{I})$. 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 $\mathsf{val}_G(\mathcal{I})$ subject to $\mathcal{I} = \langle I_{\mathsf{s}}, \ldots, I_{\mathsf{t}} \rangle$, which is **NP**-hard to approximate within any constant factor [28]. $\mathsf{Gap}_{c,s}$ Independent Set Reconfiguration requests to distinguish whether $\mathsf{val}_G(I_{\mathsf{s}} \rightsquigarrow I_{\mathsf{t}}) \geqslant c$ or $\mathsf{val}_G(I_{\mathsf{s}} \rightsquigarrow I_{\mathsf{t}}) < s$. 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* $\mathsf{Gap}_{1,1-\varepsilon}$ Nondeterministic Constraint Logic *to* $\mathsf{Gap}_{1,1-\Theta(\varepsilon)}$ Independent Set Reconfiguration*.*

As an immediate corollary, Maxmin Clique Reconfiguration is **PSPACE**-hard to approximate under Hypothesis 2.4.

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

5 Conclusions

We gave a series of gap-preserving reductions to demonstrate **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 **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

1. Noga Alon. Explicit expanders of every degree and size. *Comb.*, 41(4):447–463, 2021.
2. Noga Alon and Fan R. K. Chung. Explicit construction of linear sized tolerant networks. *Discret. Math.*, 72(1-3):15–19, 1988.
3. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.
4. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998.
5. Rémy Belmonte, Eun Jung Kim, Michael Lampis, Valia Mitsou, Yota Otachi, and Florian Sikora. Token sliding on split graphs. *Theory Comput. Syst.*, 65(4):662–686, 2021.
6. Alexandre Blanché, Haruka Mizuta, Paul Ouvrard, and Akira Suzuki. Decremental optimization of dominating sets under the reconfiguration framework. In *IWOCA*, pages 69–82, 2020.
7. Marthe Bonamy, Marc Heinrich, Takehiro Ito, Yusuke Kobayashi, Haruka Mizuta, Moritz Mühlenthaler, Akira Suzuki, and Kunihiro Wasa. Shortest reconfiguration of colorings under Kempe changes. In *STACS*, pages 35:1–35:14, 2020.

8 Édouard Bonnet, Tillmann Miltzow, and Paweł Rzążewski. Complexity of token swapping and its variants. *Algorithmica*, 80(9):2656–2682, 2018.

9 Paul Bonsma and Luis Cereceda. Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances. *Theor. Comput. Sci.*, 410(50):5215–5226, 2009.

10 Nicolas Bousquet, Tatsuhiko Hatanaka, Takehiro Ito, and Moritz Mühlenthaler. Shortest reconfiguration of matchings. In *WG*, pages 162–174, 2019.

11 Nicolas Bousquet, Takehiro Ito, Yusuke Kobayashi, Haruka Mizuta, Paul Ouvrard, Akira Suzuki, and Kunihiro Wasa. Reconfiguration of spanning trees with degree constraint or diameter constraint. In *STACS*, pages 15:1–15:21, 2022.

12 Nicolas Bousquet and Alice Joffard. Approximating shortest connected graph transformation for trees. In *SOFSEM*, pages 76–87, 2020.

13 Nicolas Bousquet and Arnaud Mary. Reconfiguration of graphs with connectivity constraints. In *WAOA*, pages 295–309, 2018.

14 Luis Cereceda, Jan van den Heuvel, and Matthew Johnson. Finding paths between 3-colorings. *J. Graph Theory*, 67(1):69–82, 2011.

15 Anne Condon, Joan Feigenbaum, Carsten Lund, and Peter W. Shor. Probabilistically checkable debate systems and nonapproximability of PSPACE-hard functions. *Chic. J. Theor. Comput. Sci.*, 1995, 1995.

16 Pierluigi Crescenzi. A short guide to approximation preserving reductions. In *CCC*, pages 262–273, 1997.

17 Pierluigi Crescenzi and Luca Trevisan. On approximation scheme preserving reducibility and its applications. *Theory Comput. Syst.*, 33(1):1–16, 2000.

18 Mark de Berg, Bart M. P. Jansen, and Debankur Mukherjee. Independent-set reconfiguration thresholds of hereditary graph classes. *Discret. Appl. Math.*, 250:165–182, 2018.

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

20 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer, 2012.

21 J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

22 Kshitij Gajjar, Agastya Vibhuti Jha, Manish Kumar, and Abhiruk Lahiri. Reconfiguring shortest paths in graphs. In *AAAI*, pages 9758–9766, 2022.

23 Parikshit Gopalan, Phokion G. Kolaitis, Elitza Maneva, and Christos H. Papadimitriou. The connectivity of Boolean satisfiability: Computational and structural dichotomies. *SIAM J. Comput.*, 38(6):2330–2355, 2009.

24 Robert A. Hearn and Erik D. Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theor. Comput. Sci.*, 343(1-2):72–96, 2005.

25 Robert A. Hearn and Erik D. Demaine. *Games, Puzzles, and Computation*. A K Peters, Ltd., 2009.

26 Lenwood S. Heath and John Paul C. Vergara. Sorting by short swaps. *J. Comput. Biol.*, 10(5):775–789, 2003.

27 Takehiro Ito and Erik D. Demaine. Approximability of the subset sum reconfiguration problem. *J. Comb. Optim.*, 28(3):639–654, 2014.

28 Takehiro Ito, Erik D. Demaine, Nicholas J. A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. *Theor. Comput. Sci.*, 412(12-14):1054–1065, 2011.

29 Takehiro Ito, Naonori Kakimura, Naoyuki Kamiyama, Yusuke Kobayashi, and Yoshio Okamoto. Shortest reconfiguration of perfect matchings via alternating cycles. *SIAM J. Discret. Math.*, 36(2):1102–1123, 2022.

30 Takehiro Ito, Marcin Kamiński, and Erik D. Demaine. Reconfiguration of list edge-colorings in a graph. *Discret. Appl. Math.*, 160(15):2199–2207, 2012.

31 Takehiro Ito, Haruka Mizuta, Naomi Nishimura, and Akira Suzuki. Incremental optimization of independent sets under the reconfiguration framework. *J. Comb. Optim.*, 43(5):1264–1279, 2022.

32 Takehiro Ito, Hiroyuki Nooka, and Xiao Zhou. Reconfiguration of vertex covers in a graph. *IEICE Trans. Inf. Syst.*, 99-D(3):598–606, 2016.

33 Matti Järvisalo and Ilkka Niemelä. A compact reformulation of propositional satisfiability as binary constraint satisfaction. In *Third International Workshop on Modelling and Reformulating Constraint Satisfaction Problems*, pages 111–124, 2004.

34 Marcin Kamiński, Paul Medvedev, and Martin Milanič. Complexity of independent set reconfigurability problems. *Theor. Comput. Sci.*, 439:9–15, 2012.

35 Tillmann Miltzow, Lothar Narins, Yoshio Okamoto, Günter Rote, Antonis Thomas, and Takeaki Uno. Approximation and hardness of token swapping. In *ESA*, pages 66:1–66:15, 2016.

36 Sidhanth Mohanty, Ryan O'Donnell, and Pedro Paredes. Explicit near-Ramanujan graphs of every degree. *SIAM J. Comput.*, 51(3):STOC20–1–STOC20–23, 2021.

37 Naomi Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018.

38 Naoto Ohsaka. Gap preserving reductions between reconfiguration problems. *CoRR*, abs/2212.04207, 2022.

39 Naoto Ohsaka and Tatsuya Matsuoka. Reconfiguration problems on submodular functions. In *WSDM*, pages 764–774, 2022.

40 Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.*, 43(3):425–440, 1991.

41 Jan van den Heuvel. The complexity of change. In *Surveys in Combinatorics 2013*, volume 409, pages 127–160. Cambridge University Press, 2013.

42 Katsuhisa Yamanaka, Erik D. Demaine, Takehiro Ito, Jun Kawahara, Masashi Kiyomi, Yoshio Okamoto, Toshiki Saitoh, Akira Suzuki, Kei Uchizawa, and Takeaki Uno. Swapping labeled tokens on graphs. *Theor. Comput. Sci.*, 586:81–94, 2015.

43 Yusuke Yanagisawa, Akira Suzuki, Yuma Tamura, and Xiao Zhou. Decremental optimization of vertex-coloring under the reconfiguration framework. In *COCOON*, pages 355–366, 2021.