Electronic Colloquium on Computational Complexity, Report No. 23 (2024)

Probabilistically Checkable Reconfiguration Proofs and Inapproximability of Reconfiguration Problems

Shuichi Hirahara[*] Naoto Ohsaka[†]

January 21, 2024

Abstract

Motivated by the inapproximability of reconfiguration problems, we present a new PCP-type characterization of PSPACE, which we call a *probabilistically checkable reconfiguration proof* (PCRP): Any PSPACE computation can be encoded into an exponentially long sequence of polynomially long proofs such that every adjacent pair of the proofs differs in at most one bit, and every proof can be probabilistically checked by reading a constant number of bits.

Using the new characterization, we prove PSPACE-completeness of approximate versions of many reconfiguration problems, such as the MAXMIN 3-SAT RECONFIGURATION problem. This resolves the open problem posed by Ito, Demaine, Harvey, Papadimitriou, Sideri, Uehara, and Uno (ISAAC 2008; Theor. Comput. Sci. 2011) as well as the Reconfiguration Inapproximability Hypothesis by Ohsaka (STACS 2023) affirmatively. We also present PSPACE-completeness of approximating the MAXMIN CLIQUE RECONFIGURATION problem to within a factor of n^{ε} for some constant $\varepsilon > 0$.

[*]National Institute of Informatics, Japan. s_hirahara@nii.ac.jp

[†]CyberAgent, Inc., Tokyo, Japan. ohsaka_naoto@cyberagent.co.jp; naoto.ohsaka@gmail.com

ISSN 1433-8092

Contents

1 Introduction **3**
1.1 Background 4
1.2 Our Results 5

2 Proof Overview **7**
2.1 Succinct Graph Reachability as Reconfiguration Problems 7
2.2 PCRP System for Succinct Graph Reachability 8

3 Related Work **9**

4 Preliminaries **10**
4.1 Notations 10
4.2 Error-Correcting and Locally Testable Codes 10
4.3 Probabilistically Checkable Proofs of Proximity 10
4.4 Constraint Satisfaction Problems 12

5 Probabilistic Checkable Reconfiguration Proofs **13**
5.1 PSPACE-completeness of Succinct Graph Reachability 13
5.2 PCRP System for Succinct Graph Reachability 14
5.2.1 Verifier Description 15
5.2.2 Completeness and Soundness 16
5.3 Impossibility of Extension to Average Case 23

6 PSPACE-hardness of Approximation for Reconfiguration Problems **23**
6.1 Constant-factor Inapproximability of Maxmin CSP Reconfiguration . . . 23
6.2 Polynomial-factor Inapproximability of Maxmin Clique Reconfiguration 24

1 Introduction

Reconfiguration problems ask to decide whether there exists a sequence of operations that transform one feasible solution to another. A canonical example is the 3-SAT RECONFIGURATION problem, which is known to be PSPACE-complete [GKMP09].

Definition 1.1 (3-SAT RECONFIGURATION [GKMP09])**.** Given a 3-CNF formula φ and its two satisfying assignments σ^{start} and σ^{goal}, we are required to decide if there is a sequence of satisfying assignments to φ, $(\sigma^{(1)},\ldots,\sigma^{(T)})$, such that $\sigma^{(1)} = \sigma^{\mathsf{start}}$, $\sigma^{(T)} = \sigma^{\mathsf{goal}}$, and $\sigma^{(t)}$ and $\sigma^{(t+1)}$ differ in at most one variable for every $t \in \{1,\ldots,T-1\}$.

Example 1.2. Suppose we are given a 3-CNF formula $\varphi := (\overline{x_1} \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3})$ made up of three clauses over three variables x_1, x_2, x_3 and its two satisfying assignments $\sigma_1^{\mathsf{start}} := (1,0,0)$ and $\sigma_1^{\mathsf{goal}} := (0,1,0)$. Then, $(\varphi, \sigma_1^{\mathsf{start}}, \sigma_1^{\mathsf{goal}})$ is a YES instance of 3-SAT RECONFIGURATION: there exists a sequence $((1,0,0),(0,0,0),(0,1,0))$ from $\sigma_1^{\mathsf{start}}$ to σ_1^{goal} that meets the requirement. On the other hand, if we are given a pair of two satisfying assignments $\sigma_2^{\mathsf{start}} := (1,0,0)$ and $\sigma_2^{\mathsf{goal}} := (1,1,1)$, then $(\varphi, \sigma_2^{\mathsf{start}}, \sigma_2^{\mathsf{goal}})$ is a NO instance because any sequence from $\sigma_2^{\mathsf{start}}$ to σ_2^{goal} must run through $(0,1,1)$, $(1,0,1)$, or $(1,1,0)$, neither of which satisfy φ.

It is natural to consider its approximate variant, whose complexity was posed as an open problem in [IDHPSUU11].

Definition 1.3 (MAXMIN 3-SAT RECONFIGURATION [IDHPSUU11])**.** Given a 3-CNF formula φ over m clauses $C_1,\ldots,C_m$ and its two satisfying assignments σ^{start} and σ^{goal}, we are required to find a sequence of assignments, $(\sigma^{(1)},\ldots,\sigma^{(T)})$, such that $\sigma^{(1)} = \sigma^{\mathsf{start}}$, $\sigma^{(T)} = \sigma^{\mathsf{goal}}$, $\sigma^{(t)}$ and $\sigma^{(t+1)}$ differ in at most one variable for every $t \in \{1,\ldots,T-1\}$, and the following objective value is maximized:

$$\min_{1 \leqslant t \leqslant T} \frac{1}{m} \left| \left\{ j \in \{1,\ldots,m\} \;\middle|\; \sigma^{(t)} \text{ satisfies } C_j \right\} \right|. \tag{1.1}$$

Example 1.4. Consider again the same 3-CNF formula φ and its two satisfying assignments $\sigma_2^{\mathsf{start}} = (1,0,0)$ and $\sigma_2^{\mathsf{goal}} = (1,1,1)$. There is a sequence $((1,0,0),(1,1,0),(1,1,1))$ from $\sigma_2^{\mathsf{start}}$ to σ_2^{goal}, which is a feasible solution to MAXMIN 3-SAT RECONFIGURATION and whose objective value is $\frac{2}{3}$.

The main contribution of this paper is to prove PSPACE-completeness of approximating the MAXMIN 3-SAT RECONFIGURATION problem within a constant factor, which answers the open problem of [IDHPSUU11]. In what follows, we present the background of this result and then the details of our results.

1.1 Background

Given a *source problem* that asks the existence of a feasible solution, *reconfiguration problems* are defined as a problem of deciding the existence of a *reconfiguration sequence*, that is, a step-by-step transformation between a pair of feasible solutions while always preserving the feasibility of solutions. For example, 3-SAT RECONFIGURATION [GKMP09] is defined from 3-SAT as a source problem. Many reconfiguration problems can be defined from Boolean satisfiability, constraint satisfaction problems, graph problems, and others. Studying reconfiguration problems may help elucidate the structure of the solution space [GKMP09], which is motivated by, e.g., the application to the behavior analysis of SAT solvers, such as DPLL [ABM04]. From a different point of view, reconfiguration problems may date back to motion planning [HSS84] and classical puzzles, including 15 puzzles [JS79] and Rubik's Cube.

Typically, a reconfiguration problem becomes PSPACE-complete if its source problem is intractable (say, NP-complete); e.g., 3-SAT [GKMP09], INDEPENDENT SET [HD05, HD09], SET COVER [IDHPSUU11], and 4-COLORING [BC09]. On the other hand, a source problem in P frequently leads to a reconfiguration problem in P, e.g., MATCHING [IDHPSUU11] and 2-SAT [GKMP09]. Some exceptions are known: whereas 3-COLORING is NP-complete, its reconfiguration problem is solvable in polynomial time [CvJ11]; SHORTEST PATH on a graph is tractable, but its reconfiguration problem is PSPACE-complete [Bon13]. We refer the readers to the surveys by Nishimura [Nis18] and van den Heuvel [van13] for algorithmic and hardness results and the Combinatorial Reconfiguration wiki [Hoa23] for an exhaustive list of related articles.

A common way to cope with intractable problems is to consider approximation problems. Relaxing the feasibility of intermediate solutions, we can formalize *approximate variants* for reconfiguration problems, which are also motivated by the situation wherein there does not exist a reconfiguration sequence for the original decision problem. For example, in MAXMIN 3-SAT RECONFIGURATION [IDHPSUU11], we are allowed to include any *non-satisfying* assignment in a reconfiguration sequence, but required to maximize the *minimum fraction of satisfied clauses*. Solving this problem may result in a reasonable reconfiguration sequence consisting of *almost-satisfying* assignments, e.g., each violating at most 1% of clauses. Intriguingly, a different trend regarding the approximability has been observed between a source problem and its reconfiguration analogue; e.g., SET COVER is NP-hard to approximate within a factor better than $\ln n$ [DS14, Fei98, LY94], whereas MINMAX SET COVER RECONFIGURATION admits a 2-factor approximation algorithm [IDHPSUU11]. Other reconfiguration problems whose approximability was investigated include: SUBSET SUM RECONFIGURATION has a PTAS [ID14]; SUBMODULAR RECONFIGURATION [OM22] and POWER SUPPLY RECONFIGURATION [IDHPSUU11] are constant-factor approximable.

Little is known about the hardness of approximation for reconfiguration problems. Using the fact that source problems (e.g., MAX 3-SAT) are NP-hard to approximate [ALMSS98, AS98], Ito, Demaine, Harvey, Papadimitriou, Sideri, Uehara, and Uno [IDHPSUU11] proved that several reconfiguration problems (e.g., MAXMIN 3-SAT RECONFIGURATION) are NP-hard to approximate; however, most reconfiguration problems are in PSPACE, and thus their NP-hardness results are not optimal. It was left open to improve the NP-hardness results to PSPACE-hardness. We here stress the significance of showing PSPACE-hardness compared to NP-hardness:

1. PSPACE-hardness is *tight* because most reconfiguration problems belong to PSPACE [Nis18];
2. it disproves the existence of a polynomial-length witness (in particular, a polynomial-length reconfiguration sequence) assuming NP $\neq$ PSPACE;
3. it rules out any polynomial-time algorithm under the weak assumption that P $\neq$ PSPACE.

In order to improve the NP-hardness of approximation to PSPACE-hardness of approximation, it is crucial to develop a reconfiguration analogue of the PCP theorem [ALMSS98, AS98]. As indicated by the gap in the approximation factors of Set Cover and its reconfiguration counterpart, the required theory must be different and tailored to PSPACE. Ohsaka [Ohs23b] recently postulated a reconfiguration analogue of the PCP theorem, called the *Reconfiguration Inapproximability Hypothesis* (RIH), under which a bunch of popular reconfiguration problems are shown to be PSPACE-hard to approximate. The major open question is whether RIH holds.

1.2 Our Results

Our contribution is to present a new PCP-type characterization of PSPACE, which we call a *probabilistically checkable reconfiguration proof* (PCPR), and thereby affirmatively resolve the open problem posed by Ito, Demaine, Harvey, Papadimitriou, Sideri, Uehara, and Uno [IDHPSUU11] and confirm RIH of Ohsaka [Ohs23b].

Our characterization of PSPACE encodes any PSPACE computation into an exponentially long reconfiguration sequence of polynomial-length proofs, each of which can be probabilistically checked by reading a constant number of bits. A *reconfiguration sequence from* π^{start} *to* π^{goal} *over* $\{0,1\}^n$ is a sequence $(\pi^{(1)},\cdots,\pi^{(T)}) \in (\{0,1\}^n)^*$ such that $\pi^{\mathsf{start}} = \pi^{(1)}$, $\pi^{\mathsf{goal}} = \pi^{(T)}$, and $\pi^{(t)}$ and $\pi^{(t+1)}$ differ in at most one bit for every $t \in \{1,\cdots,T-1\}$.

Theorem 1.5 (Probabilistically Checkable Reconfiguration Proof (PCRP); see also Theorem 5.1)**.** *A language L is in* PSPACE *if and only if there exist a randomized polynomial-time verifier V with randomness complexity $O(\log n)$ and query complexity $O(1)$ on inputs of length n and polynomial-time algorithms π^{start} and π^{goal} with the following properties:*

1. (Completeness) *If $x \in L$, then there exists a reconfiguration sequence $(\pi^{(1)},\cdots,\pi^{(T)})$ from $\pi^{\mathsf{start}}(x)$ to $\pi^{\mathsf{goal}}(x)$ over $\{0,1\}^{\mathsf{poly}(n)}$ such that for every $t \in \{1,\cdots,T\}$,*
$$\mathbb{P}\mathrm{r}\left[V^{\pi^{(t)}}(x) = 1\right] = 1.$$
2. (Soundness) *If $x \notin L$, then for every reconfiguration sequence $(\pi^{(1)},\cdots,\pi^{(T)})$ from $\pi^{\mathsf{start}}(x)$ to $\pi^{\mathsf{goal}}(x)$, for some $t \in \{1,\cdots,T\}$,*
$$\mathbb{P}\mathrm{r}\left[V^{\pi^{(t)}}(x) = 1\right] < \frac{1}{2}.$$

Here, $V^{\pi^{(t)}}(x)$ denotes the output of V on input x given oracle access to $\pi^{(t)}$, and the probabilities are over the $O(\log n)$ random bits of the verifier V.

The verifier V can be regarded as a $\forall\cdot$coRP-type verifier: The verifier co-nondeterministically guesses $t \in \{1, \cdots, T\}$ and probabilistically checks the t-th proof $\pi^{(t)}$. This verifier should be compared with the standard coRP-type PCP verifier V' for PSPACE-complete problems, which can be obtained from the PCP theorem for NEXP $\supseteq$ PSPACE [BFL91]. The number of random bits used by V' is $n^{\Theta(1)}$, whereas the number of random bits of our verifier V is $O(\log n)$. The latter is crucial for the application to inapproximability of reconfiguration problems. The standard verifier V' uses only random bits, whereas our verifier V co-nondeterministically guesses t. Given that V' does not use any nondeterministic choice, it is natural to wonder whether V can be improved to a coRP-type verifier that chooses $t \in \{1, \cdots, T\}$ randomly; however, such an extension is impossible (see Observation 5.9), and thus our characterization is one of the "best" characterizations in this direction.

As a corollary of Theorem 5.1 and [Ohs23a, Ohs23b], we obtain that a host of reconfiguration problems are PSPACE-complete to approximate.

Corollary 1.6 (from Theorem 5.1 and [Ohs23a, Ohs23b])**.** *For some universal constant $\varepsilon_0 \in (0,1)$, the following approximate variants of reconfiguration problems are* PSPACE*-hard to approximate within a factor of* $(1-\varepsilon_0)$*.*

- MAXMIN k-SAT RECONFIGURATION *for all $k \geqslant 2$;*
- MAXMIN q-CSP RECONFIGURATION *for all $q \geqslant 2$;*
- MAXMIN INDEPENDENT SET RECONFIGURATION *on bounded-degree graphs;*
- MINMAX VERTEX COVER RECONFIGURATION *on bounded-degree graphs;*
- MAXMIN CLIQUE RECONFIGURATION*;*
- MINMAX DOMINATING SET RECONFIGURATION*;*
- MINMAX SET COVER RECONFIGURATION*;*
- MAXMIN NONDETERMINISTIC CONSTRAINT LOGIC.

Moreover, we improve an inapproximability factor of MAXMIN CLIQUE RECONFIGURATION to a polynomial; that is, MAXMIN CLIQUE RECONFIGURATION is PSPACE-hard to approximate within a factor of n^{ε} for some constant $\varepsilon > 0$, where n is the number of vertices (Theorem 6.2). This is the first polynomial-factor inapproximability result for approximate variants of reconfiguration problems (to the best of our knowledge).[1]

[1] See Section 3 for discussion of existing hardness-of-approximation results for reconfiguration problems.

2 Proof Overview

Here, we present a proof sketch of Theorem 1.5.

A naïve attempt for the proof of Theorem 1.5 would be to develop reconfiguration counterparts for the simple proof of the PCP theorem by Dinur [Din07]. The proof of the PCP theorem consists of repeated applications of the three steps — a degree reduction (the preprocessing lemma [Din07, Lemma 1.9]), gap amplification and an alphabet reduction. Counterparts of some of the steps have been developed in the recent literature of reconfiguration problems [Ohs23a, Ohs23b]. For example, Ohsaka [Ohs23b] presented a degree reduction for reconfiguration problems, i.e., a reduction that converts a graph that represents a PCRP (probabilistically checkable reconfiguration proof) system with soundness error $1-\varepsilon$ into another graph whose degree Δ is small. However, the parameter achieved in [Ohs23b] is weaker than that of [Din07, PY91]: $\Delta \le \mathrm{poly}(1/\varepsilon)$. If $\varepsilon = o(1)$, the degree Δ can be $\omega(1)$, which is not sufficient for Dinur's proof to go through. It appears to be very difficult to construct PCRPs based on this approach.

Our actual approach is much simpler. We use existing machinery developed in the literature of PCP theorems in a black-box way. The main ingredient for our proof is the *PCP of Proximity* (PCPP) [BGHSV06, DR06]. A PCPP for a language $L \in \mathsf{NP}$ allows us to approximately verify that $x \in L$ by reading a constant number of bits from x and a proof. In particular, by encoding x by an error-correcting code, we can reliably check whether $x \in L$ efficiently.

To construct a PCRP for every problem in PSPACE, it suffices to construct a PCRP for some PSPACE-complete problem. We consider the PSPACE-complete problem called SUCCINCT GRAPH REACHABILITY. In what follows, we first explain how this problem can be regarded as a reconfiguration problem, and then explain how to construct a PCRP system for SUCCINCT GRAPH REACHABILITY.

2.1 SUCCINCT GRAPH REACHABILITY as Reconfiguration Problems

SUCCINCT GRAPH REACHABILITY is the following problem. The input consists of a circuit which succinctly represents an exponentially large graph $G = (V, E)$ and two vertices v^{start} and $v^{\mathsf{goal}} \in V$, and the task is to decide whether there exists a path from v^{start} to v^{goal} in G. Each vertex is represented by an n-bit string; i.e., $V = \{0,1\}^n$. For simplicity of notation, throughout this section, we assume that every vertex in G has a self-loop, i.e., $(x, x) \in E$ for every $x \in V$. For two strings x and y, we denote by $x \circ y$ the concatenation of x and y.

SUCCINCT GRAPH REACHABILITY can be naturally regarded as the following reconfiguration problem. Given a (succinctly described) graph G and two vertices $v^{\mathsf{start}}, v^{\mathsf{goal}} \in V$, the task is to decide whether there exists a sequence $(x_1 \circ y_1, \cdots, x_T \circ y_T) \in (\{0,1\}^{2n})^*$ from $v^{\mathsf{start}} \circ v^{\mathsf{start}}$ to $v^{\mathsf{goal}} \circ v^{\mathsf{goal}}$ such that

1. every configuration $x_t \circ y_t \in \{0,1\}^{2n}$ satisfies the constraint that $(x_t, y_t) \in E$, and
2. each adjacent pair of configurations satisfy $x_t = x_{t+1}$ or $y_t = y_{t+1}$.

In other words, this is the reconfiguration problem which asks to decide whether the token that initially placed at the edge $(v^{\text{start}}, v^{\text{start}})$ can be moved to the edge $(v^{\text{goal}}, v^{\text{goal}})$ by a sequence of operations that move the token from an edge to one of its adjacent edges.

In Theorem 1.5, each adjacent pair of proofs differs in at most one bit. In terms of reconfiguration problems, this means that the operations which we are allowed to perform are to change *one bit* of a configuration instead of *one vertex* of the token placed at an edge. By introducing a special symbol "$\perp$", we can regard SUCCINCT GRAPH REACHABILITY as the following reconfiguration problem in which operations are restricted to changing one bit of configurations: Given a (succinctly described) graph G and two vertices $v^{\text{start}}, v^{\text{goal}} \in V$, the task is to decide whether there exists a sequence $(x_1 \circ y_1, \cdots, x_T \circ y_T) \in (\{0,1,\perp\}^{2n})^*$ from $v^{\text{start}} \circ v^{\text{start}}$ to $v^{\text{goal}} \circ v^{\text{goal}}$ such that

1. $(x_t, y_t) \in E$ or ($x_t \in \{0,1\}^n$ and $y_t \in \{0,1\}^n$), and
2. each adjacent pair $(x_t \circ y_t, x_{t+1} \circ y_{t+1})$ differs in at most one position.

Informally, the existence of the symbol $\perp$ indicates that we are on the way of the transition, and we are allowed to include $\perp$ in at most one of x_t or y_t (that is, we do not allow to change both vertices of the token simultaneously). This reconfiguration problem "simulates" SUCCINCT GRAPH REACHABILITY in the following sense: If a token placed at an edge $(x, y_1) \in E$ is moved to another edge $(x, y_2) \in E$, then in the new reconfiguration problem, we may consider a sequence of operations that first transform $x \circ y_1$ into $x \circ \perp^n$ by replacing each bit of y_1 with $\perp$ one by one, and then transform $x \circ \perp^n$ into $x \circ y_2$ by replacing $\perp$ with a bit of y_2 one by one.

2.2 PCRP System for SUCCINCT GRAPH REACHABILITY

The main idea for constructing a PCRP for SUCCINCT GRAPH REACHABILITY is to probabilistically check item 1, i.e., the condition that $(x_t, y_t) \in E$ or ($x_t \in \{0,1\}^n$ and $y_t \in \{0,1\}^n$), by reading a constant number of bits. To this end, we encode each vertex by a locally testable error-correcting code $\mathsf{Enc}\colon \{0,1\}^n \to \{0,1\}^\ell$ and use the PCPP to check whether the encoded pair of vertices (x_t, y_t) satisfies $(x_t, y_t) \in E$. Specifically, let V_{PCPP} be a PCPP verifier for the language $L_G = \{\mathsf{Enc}(x) \circ \mathsf{Enc}(y) \mid (x,y) \in E\}$. This verifier takes random access to $f \circ g$ and a proof $\pi \in \{0,1\}^p$ and checks whether $f \circ g$ is close to some $\mathsf{Enc}(x) \circ \mathsf{Enc}(y) \in L_G$. Then, given a sequence of proofs $(\sigma^{(1)}, \cdots, \sigma^{(T)}) \in (\{0,1,\perp\}^{2\ell+p})^*$, we probabilistically check each proof $\sigma^{(t)} = f \circ g \circ \pi$ as follows:

1. Using the local tester for Enc, we check that both f and g are close to some codewords $\mathsf{Enc}(x)$ and $\mathsf{Enc}(y)$, respectively. If both are far from codewords, then we reject.
2. By random sampling, we test whether either f or g contains many $\perp$ symbols. If so, we accept.
3. Finally, by running V_{PCPP} for $(f \circ g, \pi)$, we check that $(x,y) \in E$. We accept if and only if V_{PCPP} accepts.

The first item ensures that either f or g is close to some codewords $\mathsf{Enc}(x)$ and $\mathsf{Enc}(y)$. The second item checks whether we are on the way of the transition from one edge to another, in which case we accept. We run the test of the third item only if either f or g is close to some codewords, and both f and g do not contain many $\perp$ symbols. Using the PCPP, we check that f and g encode x and y such that $(x, y) \in E$.

We note that the size of alphabets $\{0, 1, \perp\}$ of the PCRP system is 3. This can be reduced to 2 by using a simple alphabet reduction of Ohsaka [Ohs23b], which transforms any PCRP system with *perfect completeness* over alphabets of constant size into a PCRP system over the binary alphabets $\{0, 1\}$.

It is thus important to make sure that the PCRP system has perfect completeness, i.e., in the YES case, the verifier accepts with probability 1. For this reason, in the actual proof, we need to modify the PCPP verifier V_{PCPP} so that it immediately accepts if a $\perp$ symbol in f or g is queried by V_{PCPP}. Details can be found in Section 5.

3 Related Work

Another characterization of PSPACE is *probabilistically checkable debate systems* due to Condon, Feigenbaum, Lund, and Shor [CFLS95], which can be used to show PSPACE-hardness of approximating QUANTIFIED BOOLEAN FORMULA and the problem of selecting as many finite-state automata as possible that accept a common string. These results are incomparable to our PCRP because the underlying structure of the problems is different from each other.

We summarize approximate variants of reconfiguration problems whose inapproximability was investigated. MAXMIN CLIQUE RECONFIGURATION and MAXMIN SAT RECONFIGURATION are NP-hard to approximate [IDHPSUU11]. SHORTEST PATH RECONFIGURATION is PSPACE-hard to approximate with respect to its objective value [GJKL22]. The obejctive value called the *price* is determined based on the number of vertices in a path changed at a time, which is fundamentally different from those of reconfiguration problems listed in Corollary 1.6. SUBMODULAR RECONFIGURATION is constant-factor inapproximable [OM22], whose proof resorts to inapproximability results of SUBMODULAR FUNCTION MAXIMIZATION [FMV11].

We note that approximability of reconfiguration problems frequently refers to that of *the shortest sequence* [BHIKMMSW20, IKKKO22, KMM11, MNORTU16], which seems orthogonal to the present study.

The *pebble game* [PH70] is a single-player game, which models the trade-off between the memory usage and running time of a computation and is recently used in the context of proof complexity [Nor13]. This game can be thought of as a reconfiguration problem, whose objective function, called the *pebbling price*, is defined as the maximum number of pebbles at any time required to place a pebble to the unique sink. The pebbling price is known to be PSPACE-hard to approximate within an additive $n^{\frac{1}{3}-\varepsilon}$ term for the graph size n [CLNV15, DL17]. We leave open whether our PCRPs can be used to derive PSPACE-hardness of approximating the pebbling price within a multiplicative factor.

4 Preliminaries

4.1 Notations

For a nonnegative integer $n \in \mathbb{N}$, let $[n] := \{1, 2, \ldots, n\}$. A *sequence* $\mathscr{E}$ of a finite number of elements $E^{(1)}, \ldots, E^{(T)}$ is denoted by $(E^{(1)}, \ldots, E^{(T)})$, and we write $E \in \mathscr{E}$ to indicate that E appears in $\mathscr{E}$. The symbol $\circ$ stands for a concatenation of two strings. Let Σ be a finite set called *alphabet*. For a length-n string $f \in \Sigma^n$ and index set $I \subseteq [n]$, we use $f|_I$ to denote the restriction of f to I. We write 0^n and 1^n for $\underbrace{0 \cdots 0}_{n \text{ times}}$ and $\underbrace{1 \cdots 1}_{n \text{ times}}$, respectively. The *relative distance* between two strings $f, g \in \Sigma^n$, denoted $\Delta(f, g)$, is defined as the fraction of positions on which f and g differ; namely, $\Delta(f, g) = \mathbb{P}\mathrm{r}_{i \sim [n]}[f_i \neq g_i] = \frac{|\{i \in [n] | f_i \neq g_i\}|}{n}$. We say that f is *ε-close* to g if $\Delta(f, g) \leqslant \varepsilon$ and *ε-far* from g if $\Delta(f, g) > \varepsilon$. For a set of strings $S \subseteq \Sigma^n$, analogous notions are defined; e.g., $\Delta(f, S) := \min_{g \in S} \Delta(f, g)$ and f is ε-close to S if $\Delta(f, S) \leqslant \varepsilon$.

4.2 Error-Correcting and Locally Testable Codes

Here, we introduce error-correcting and locally testable codes.

Definition 4.1 (Error-correcting codes)**.** For any $\rho \in [0, 1]$, a function $\mathsf{Enc}\colon \{0, 1\}^n \to \{0, 1\}^\ell$ is an *error-correcting code with relative distance* ρ if $\Delta(\mathsf{Enc}(\alpha), \mathsf{Enc}(\beta)) > \rho$ for every $\alpha \neq \beta \in \{0, 1\}^n$. We call $\mathsf{Enc}(\alpha)$ for each $\alpha \in \{0, 1\}^n$ a *codeword* of Enc. Denote by $\mathsf{Enc}(\cdot)$ the set of all codewords of Enc.

Definition 4.2 (Locally testable codes; e.g., Goldreich and Sudan [GS06])**.** For any $q \in \mathbb{N}$ and $\kappa > 0$, an error-correcting code $\mathsf{Enc}\colon \{0, 1\}^n \to \{0, 1\}^\ell$ is said to be (q, κ)-*locally testable* if there exists a probabilistic polynomial-time algorithm M that, given oracle access to a string $f \in \{0, 1\}^\ell$, makes at most q nonadaptive queries of f and satisfies the following conditions:

- (Completeness) If $f \in \mathsf{Enc}(\cdot)$, then M always accepts; namely, $\mathbb{P}\mathrm{r}[M^f \text{ accepts}] = 1$.
- (Soundness) If $f \notin \mathsf{Enc}(\cdot)$, then M rejects with probability at least $\kappa \cdot \Delta(f, \mathsf{Enc}(\cdot))$; namely, $\mathbb{P}\mathrm{r}[M^f \text{ rejects}] \geqslant \kappa \cdot \Delta(f, \mathsf{Enc}(\cdot))$.

Such an algorithm M is called a (q, κ)-*local tester* for Enc.

Theorem 4.3 ([BGHSV06, BSVW03])**.** *There exist $\rho, \kappa > 0$ and $q \in \mathbb{N}$ such that for infinitely many n's, there exists a polynomial-time construction of a (q, κ)-locally testable error-correcting code $\mathsf{Enc}_n\colon \{0, 1\}^n \to \{0, 1\}^{\ell(n)}$ with relative distance ρ and $\ell(n) = n^{1+o(1)}$. Moreover, if the code $\mathsf{Enc}_n\colon \{0, 1\}^n \to \{0, 1\}^{\ell(n)}$ of the desired property exists for integer $n \in \mathbb{N}$, then the next integer $n' \in \mathbb{N}$ for which $\mathsf{Enc}_{n'}\colon \{0, 1\}^{n'} \to \{0, 1\}^{\ell(n')}$ exists is at most $n^{1+o(1)}$.*

4.3 Probabilistically Checkable Proofs of Proximity

We formally define the notion of *verifier*.

Definition 4.4 (Verifier)**.** A *verifier* with *randomness complexity* $r: \mathbb{N} \to \mathbb{N}$ and *query complexity* $q: \mathbb{N} \to \mathbb{N}$ is a probabilistic polynomial-time algorithm V that given an input $x \in \{0,1\}^*$, tosses $r = r(|x|)$ random bits R and use R to generate a sequence of $q = q(|x|)$ queries $I = (i_1, \ldots, i_q)$ and a circuit $D: \{0,1\}^q \to \{0,1\}$. We write $(I,D) \sim V(x)$ to denote the random variable for a pair of the query sequence and circuit generated by V on input $x \in \{0,1\}^*$. Denote by $V^\pi(x) := D(\pi|_I)$ the output of V on input x given oracle access to a proof $\pi \in \{0,1\}^*$. We say that $V(x)$ *accepts* a proof π if $V^\pi(x) = 1$; i.e., $D(\pi|_I) = 1$ for $(I,D) \sim V(x)$.

We proceed to the definition of *PCPs of proximity* [BGHSV06] (a.k.a. *assignment testers* [DR06]). For any pair language $L \subseteq \{0,1\}^* \times \{0,1\}^*$, we define $L(x) := \{y \in \{0,1\}^* \mid (x,y) \in L\}$ for an input $x \in \{0,1\}^*$.

Definition 4.5 (PCP of proximity [BGHSV06, DR06])**.** A *PCP of proximity* (PCPP) verifier for a pair language $L \subseteq \{0,1\}^* \times \{0,1\}^*$ with *proximity parameter* $\delta \in (0,1)$ and *soundness error* $s \in (0,1)$ is a verifier V such that for every pair of an explicit input $x \in \{0,1\}^*$ and an implicit input oracle $y \in \{0,1\}^*$, the following conditions hold:

- (Completeness) If $(x,y) \in L$, there exists a proof $\pi \in \{0,1\}^*$ such that $V(x)$ accepts $y \circ \pi$ with probability 1; namely,

$$\exists \pi \in \{0,1\}^*, \quad \Pr_{(I,D)\sim V(x)}\Big[D((y \circ \pi)|_I) = 1\Big] = 1. \tag{4.1}$$

- (Soundness) If y is δ-far from $L(x)$, for every alleged proof $\pi \in \{0,1\}^*$, $V(x)$ accepts $y \circ \pi$ with probability less than s; namely,

$$\forall \pi \in \{0,1\}^*, \quad \Pr_{(I,D)\sim V(x)}\Big[D((y \circ \pi)|_I) = 1\Big] < s. \tag{4.2}$$

Subsequently, we introduce smooth PCPP verifiers. We say that a verifier is *smooth* if each position in its proof is equally likely to be queried.

Definition 4.6 (Smoothness)**.** A PCPP verifier V is *smooth* if V queries each position in implicit input y and proof π with equal probability; namely, there exists $p \in (0,1)$ such that

$$p = \Pr_{(I,D)\sim V(x)}\Big[i \in I\Big] \tag{4.3}$$

for every position i of $y \circ \pi$.

Theorem 4.7 (Smooth PCPP [BGHSV06, Par21])**.** *For every pair language L in* NP *and every number $\delta \in (0,1)$, there exists a smooth PCPP verifier V for L with randomness complexity $r(n) = O(\log n)$, query complexity $q(n) = O(1)$, proximity parameter δ, and soundness error $s = 1 - \Omega(\delta)$. Moreover, for every pair $(x,y) \in L$, a proof $\pi \in \{0,1\}^{\mathsf{poly}(n)}$ such that $V(x)$ always accepts $y \circ \pi$ can be constructed in polynomial time.*

4.4 Constraint Satisfaction Problems

Here, we review constraint satisfaction problems.

Definition 4.8. A *q-ary constraint system* over variable set N and alphabet Σ is defined as a collection of q-ary *constraints*, $\Psi = (\psi_j)_{j\in[m]}$, where each constraint $\psi_j\colon \Sigma^N \to \{0,1\}$ depends on q variables of N; namely, there exist $i_1,\dots,i_q \in N$ and $f\colon \Sigma^q \to \{0,1\}$ such that $\psi_j(A) = f(A(i_1),\dots,A(i_q))$ for every $A\colon N \to \Sigma$.

For an assignment $A\colon N \to \Sigma$, we say that A *satisfies* constraint ψ_j if $\psi_j(A) = 1$, and A *satisfies* Ψ if it satisfies all constraints of Ψ. Moreover, we say that Ψ is *satisfiable* if Ψ is satisfied by some assignment. For an assignment $A\colon N \to \Sigma$, its *value* is defined as the fraction of constraints of Ψ satisfied by A; namely,

$$\mathsf{val}_\Psi(A) := \frac{1}{|\Psi|}\cdot\left|\left\{\psi_j \in \Psi \;\middle|\; A \text{ satisfies } \psi_j\right\}\right|. \tag{4.4}$$

We refer to the equivalence between a PCP system and GAP q-CSP (see, e.g., [AB09, Section 11.3]), whose proof is included for the sake of completeness.

Proposition 4.9. *Let V be a verifier with randomness complexity $O(\log n)$, query complexity $O(1)$, and alphabet Σ, and let $x \in \{0,1\}^*$ be an input. Then, one can construct in polynomial time a constraint system $\Psi = (\psi_j)_{j\in[m]}$ over $\mathsf{poly}(|x|)$ variables and alphabet Σ such that $\mathsf{val}_\Psi(\pi) = \mathbb{P}\mathrm{r}[V^\pi(x) = 1]$ for every proof $\pi \in \Sigma^{\mathsf{poly}(|x|)}$.*

On the other hand, for a q-ary constraint system Ψ over variable set N and alphabet Σ, one can construct in polynomial time a verifier V with randomness complexity $O(\log n)$, query complexity $O(1)$, and alphabet Σ such that $\mathbb{P}\mathrm{r}[V^A = 1] = \mathsf{val}_\Psi(A)$ for every assignment $A\colon N \to \Sigma$.

Proof. Let V be a verifier with randomness complexity $r(n) = O(n)$, query complexity $q(n) = q \in \mathbb{N}$, and alphabet Σ. Given an input $x \in \{0,1\}^*$, we can assume the proof length for V to be $\mathsf{poly}(|x|)$. We construct a q-ary constraint system Ψ over variable set $N := [\mathsf{poly}(|x|)]$ and alphabet Σ as follows:

- for every possible sequence $R \in \{0,1\}^{r(|x|)}$ of $r(|x|)$ random bits, we run $V(x)$ to generate a query sequence $I_R = (i_1,\dots,i_q)$ and a circuit $D_R\colon \Sigma^q \to \{0,1\}$ in polynomial time.
- create a new constraint ψ_j such that

  $$\psi_j(A) := D_R(A(i_1),\dots,A(i_q)) \tag{4.5}$$

  for every assignment $A\colon N \to \Sigma$.

Note that the construction of Ψ completes in polynomial time; in particular, the size of Ψ is polynomial in $|x|$. Observe that for any proof $\pi \in \Sigma^N$, which can be thought of as an assignment to Ψ,

$$\mathsf{val}_\Psi(\pi) = \mathop{\mathbb{P}\mathrm{r}}_{\psi_j\sim\Psi}\Big[\psi_j(\pi) = 1\Big] = \mathop{\mathbb{P}\mathrm{r}}_{(I,D)\sim V(S)}\Big[D(\pi|_I) = 1\Big] = \mathbb{P}\mathrm{r}\Big[V^\pi(S) = 1\Big], \tag{4.6}$$

completing the proof of the first statement. The second statement is omitted as can be shown similarly. $\square$

5 Probabilistic Checkable Reconfiguration Proofs

In this section, we prove the main result of this paper, i.e., a PCRP verifier for a PSPACE-complete reconfiguration problem. For any pair of proofs $\pi^{\mathsf{start}}, \pi^{\mathsf{goal}} \in \Sigma^n$, a *reconfiguration sequence from* π^{start} *to* π^{goal} *over* Σ^n is a sequence $(\pi^{(1)}, \ldots, \pi^{(T)}) \in (\Sigma^n)^*$ such that $\pi^{(1)} = \pi^{\mathsf{start}}$, $\pi^{(T)} = \pi^{\mathsf{goal}}$, and $\pi^{(t)}$ and $\pi^{(t+1)}$ differ in at most one symbol for every $t \in [T-1]$.

Theorem 5.1 (Probabilistic Checkable Reconfiguration Proof (PCRP))**.** *A language L is in* PSPACE *if and only if there exists a verifier V with randomness complexity $r(n) = O(\log n)$ and query complexity $q(n) = O(1)$, coupled with polynomial-time computable proofs $\pi^{\mathsf{start}}, \pi^{\mathsf{goal}}$: $\{0,1\}^* \to \{0,1\}^*$ such that the following hold for every input $x \in \{0,1\}^*$:*

- (Completeness) *If $x \in L$, there exists a reconfiguration sequence $\pi = (\pi^{(1)}, \ldots \pi^{(T)})$ from $\pi^{\mathsf{start}}(x)$ to $\pi^{\mathsf{goal}}(x)$ over $\{0,1\}^{\mathsf{poly}(n)}$ such that $V(x)$ accepts every proof with probability* 1*; namely,*

$$\forall t \in [T],\ \mathbb{P}\mathrm{r}\left[V^{\pi^{(t)}}(x) = 1\right] = 1. \tag{5.1}$$

- (Soundness) *If $x \notin L$, every reconfiguration sequence $\pi = (\pi^{(1)}, \ldots \pi^{(T)})$ from $\pi^{\mathsf{start}}(x)$ to $\pi^{\mathsf{goal}}(x)$ over $\{0,1\}^{\mathsf{poly}(n)}$ includes a proof that is rejected by $V(x)$ with probability more than $\frac{1}{2}$; namely,*

$$\exists t \in [T],\ \mathbb{P}\mathrm{r}\left[V^{\pi^{(t)}}(x) = 1\right] < \frac{1}{2}. \tag{5.2}$$

5.1 PSPACE-completeness of Succinct Graph Reachability

We first introduce a canonical PSPACE-complete problem called Succinct Graph Reachability, for which we design a PCRP system.

Problem 5.2. For a polynomial-size circuit $S\colon \{0,1\}^n \to \{0,1\}^n$ promised that $S(1^n) = 1^n$, Succinct Graph Reachability requests to decide if there is a sequence of a finite number of assignments, $(\alpha^{(1)}, \ldots, \alpha^{(T)})$, from 0^n to 1^n such that $\alpha^{(t)} = \alpha^{(t+1)}$, $S(\alpha^{(t)}) = \alpha^{(t+1)}$, or $S(\alpha^{(t+1)}) = \alpha^{(t)}$ for all $t \in [T-1]$.

Similar variants were formulated previously, e.g., [GW83, PY86]. PSPACE-completeness of Succinct Graph Reachability is shown below for the sake of completeness.

Proposition 5.3. Succinct Graph Reachability *is* PSPACE*-complete.*

Proof. For the sake of convenience, we first show that a circuit $S\colon \{0,1\}^n \to \{0,1\}^n$ is a YES instance of Succinct Graph Reachability if and only if

$$\exists m \in \mathbb{N} \text{ such that } \underbrace{S \circ \cdots \circ S}_{m \text{ times}}(0^n) = 1^n. \tag{5.3}$$

The "if" direction is obvious because whenever $\underbrace{S \circ \cdots \circ S}_{m \text{ times}}(0^n) = 1^n$, the sequence $(\alpha^{(1)}, \ldots, \alpha^{(m+1)})$ such that $\alpha^{(t)} = \underbrace{S \circ \cdots \circ S}_{t-1 \text{ times}}(0^n)$ for all $t \in [m+1]$ satisfies the desired property. Suppose we have a sequence $(\alpha^{(1)}, \ldots, \alpha^{(T)})$ from 0^n to 1^n such that $\alpha^{(t)} = \alpha^{(t+1)}$, $S(\alpha^{(t)}) = \alpha^{(t+1)}$, or $S(\alpha^{(t+1)}) = \alpha^{(t)}$. It is not hard to see that for each t, $\underbrace{S \circ \cdots \circ S}_{m_t \text{ times}}(\alpha^{(t)}) = 1^n$ for some $m_t \in \mathbb{N}$; in particular, this is the case for $\alpha^{(1)} = 0^n$, completing the "only if" direction.

We now show the PSPACE-completeness of SUCCINCT GRAPH REACHABILITY. Membership in PSPACE follows from the fact that SUCCINCT GRAPH REACHABILITY $\in$ NPSPACE and Savitch's theorem [Sav70]. Consider the following PSPACE-complete problem: Given a deterministic Turing machine M, input $x \in \{0,1\}^*$, and 1^n, does M accept x in space n? Note that all possible configurations of M having space n on input x are specified by $\{0,1\}^{an}$ for some constant a depending on M. Let $c_{\text{init}} \in \{0,1\}^{an}$ denote the initial configuration of M on input x, and $M(x,c) \in \{0,1\}^{an}$ denote the next configuration of M following $c \in \{0,1\}^{an}$. Define now a circuit $S\colon \{0,1\}^{2+an} \to \{0,1\}^{2+an}$ of polynomial size (in $|M|$, $|x|$, and n) as follows:

$$
\begin{aligned}
S(00 \circ 0^{an}) &:= 01 \circ c_{\text{init}}, \\
S(01 \circ c) &:= \begin{cases} 11 \circ 1^{an} & \text{if } c \text{ is any accepting configuration,} \\ 00 \circ 0^{an} & \text{if } c \text{ is any rejecting configuration,} \\ 01 \circ M(x,c) & \text{otherwise,} \end{cases} \\
S(11 \circ 1^{an}) &:= 11 \circ 1^{an}.
\end{aligned}
\tag{5.4}
$$

Observe easily that $\underbrace{S \circ \cdots \circ S}_{m \text{ times}}(0^{2+an}) = 1^{2+an}$ for some $m \in \mathbb{N}$ if and only if M accepts x in space n, completing the proof. □

5.2 PCRP System for SUCCINCT GRAPH REACHABILITY

Here, we will construct a PCRP system for SUCCINCT GRAPH REACHABILITY. We first encode the assignment to a circuit by an error-correcting code. For a polynomial-size circuit $S\colon \{0,1\}^n \to \{0,1\}^n$, let $\mathsf{Enc}\colon \{0,1\}^n \to \{0,1\}^{\ell(n)}$ be an $(\mathcal{O}(1), \kappa)$-locally testable error-correcting code with relative distance ρ such that $\rho \in (0,1)$, $\kappa \in \mathbb{N}$, and $\ell(n) = n^{1+o(1)}$ by Theorem 4.3.[2] Let M be an $(\mathcal{O}(1), \kappa)$-local tester for Enc. Consider the following pair language $L_{\text{ckt}} \subseteq \{0,1\}^* \times \{0,1\}^*$: Given a polynomial-size circuit $S\colon \{0,1\}^n \to \{0,1\}^n$ and two strings $f, g \in \{0,1\}^{\ell(n)}$, we define $(S, f \circ g) \in L_{\text{ckt}}$ if and only if $f = \mathsf{Enc}(\alpha)$ and $g = \mathsf{Enc}(\beta)$ for a pair $\alpha, \beta \in \{0,1\}^n$ such that $\alpha = \beta$, $S(\alpha) = \beta$, or $S(\beta) = \alpha$. Intuitively, L_{ckt} determines the adjacency relation of codewords of Enc with respect to S. Observe easily that L_{ckt} is in NP, and by Proposition 5.3, S is a YES instance of SUCCINCT GRAPH REACHABILITY if and only if there exists a sequence of

[2]Without loss of generality, we can assume Theorem 4.3 holds for given n because we can find $n' = n^{1+o(1)}$ for which Theorem 4.3 holds and construct a slightly larger circuit $S'\colon \{0,1\}^{n'} \to \{0,1\}^{n'}$ such that S' is a YES instance if and only if so is S.

strings, $(f^{(1)},\ldots,f^{(T)})$, from $\mathsf{Enc}(0^n)$ to $\mathsf{Enc}(1^n)$ such that $(S, f^{(t)} \circ f^{(t+1)}) \in L_{\mathrm{ckt}}$ for all $t \in [T-1]$. We will use $L_{\mathrm{ckt}}(S)$ to denote the set of all strings $f \circ g \in \{0,1\}^{2\ell(n)}$ such that $(S, f \circ g) \in L_{\mathrm{ckt}}$; namely,

$$
\begin{aligned}
L_{\mathrm{ckt}}(S) &:= \left\{ f \circ g \in \{0,1\}^{2\ell(n)} \;\middle|\; (S, f \circ g) \in L_{\mathrm{ckt}} \right\} \\
&= \left\{ \mathsf{Enc}(\alpha) \circ \mathsf{Enc}(\beta) \;\middle|\; \alpha = \beta \vee S(\alpha) = \beta \vee S(\beta) = \alpha \right\}.
\end{aligned}
\tag{5.5}
$$

Let V_{ckt} denote a smooth PCPP verifier for L_{ckt}, having randomness complexity $r(n) = O(\log n)$, query complexity $q(n) = q \in \mathbb{N}$, proximity parameter $\delta_{\mathrm{ckt}} := \frac{\rho}{4} \in (0,1)$, and soundness error $s_{\mathrm{ckt}} := 1 - \Omega(\delta_{\mathrm{ckt}}) \in (0,1)$ obtained by Theorem 4.7. Note that the proof length can be bounded by some polynomial $p(n)$ in the input length n. Hereafter, we use a new symbol "$\perp$" that is neither 0 nor 1. We will write $f \circ g \circ \pi$ for a string provided to $V_{\mathrm{ckt}}(S)$, where $f \circ g \in \{0,1,\perp\}^{\ell(n)} \times \{0,1,\perp\}^{\ell(n)}$ and $\pi \in \{0,1,\perp\}^{p(n)}$ is an alleged proof.

5.2.1 Verifier Description

Our verifier V is given a polynomial-size circuit $S\colon \{0,1\}^n \to \{0,1\}^n$ and oracle access to $f \circ g \circ \pi \in \{0,1,\perp\}^{2\ell(n)+p(n)}$, and is designed as follows:

1. V ensures that f *or* g must be a codeword of Enc by running the local tester M on f and g separately. Note that M rejects whenever it reads $\perp$ at least once, which still ensures that $\Pr[M^f \text{ rejects}] \geq \kappa \cdot \Delta(f, \mathsf{Enc}(\cdot))$.

2. V allows $f \circ g$ to contain $\perp$, enabling f or g to transform between different codewords of Enc. Specifically, V accepts with probability equal to the fraction of $\perp$ in f or g, which can be done by testing whether $f_i = \perp$ or $g_j = \perp$ for independently and uniformly chosen $i, j \in [\ell(n)]$. During $f = \perp^n$ or $g = \perp^n$, the contents of π can be modified arbitrarily without being rejected, which is essential in the perfect completeness (Lemma 5.4).

3. On the other hand, if neither f nor g contains "many" $\perp$'s, V expects $f \circ g$ to be close to $L_{\mathrm{ckt}}(S)$; thus, it wants to execute the smooth PCPP verifier $V_{\mathrm{ckt}}(S)$, whose behavior is, however, undefined if $f \circ g \circ \pi$ contains $\perp$. Instead, we run a *modified verifier* $V'_{\mathrm{ckt}}(S)$, which accepts if and only if $(f \circ g)|_I$ contains $\perp$ or ($\pi|_I$ does not contain $\perp$ and $D((f \circ g \circ \pi)|_I) = 1$) for $(I, D) \sim V_{\mathrm{ckt}}(S)$.[3] This test is crucial for proving the soundness (Lemma 5.5).

The precise pseudocode of $V(S)$ is presented below.

[3] In the latter case, we can safely assume that $(f \circ g)|_I$ does not contain $\perp$.

Verifier $V^{f \circ g \circ \pi}(S)$ using local tester M for Enc and smooth PCPP verifier V_{ckt} for L_{ckt}.

Input: a polynomial-size circuit $S\colon \{0,1\}^n \to \{0,1\}^n$.
Oracle access: strings $f, g \in \{0,1,\perp\}^{\ell(n)}$ representing an implicit input and a proof $\pi \in \{0,1,\perp\}^{p(n)}$.

run local tester M on f and g. ▷ *M declares reject if it reads $\perp$.*
if both runs of M declare reject **then**
  reject.
pick $i \sim [\ell(n)]$ and $j \sim [\ell(n)]$ independently and uniformly.
if $f_i \neq \perp$ and $g_j \neq \perp$ **then**
  ▷ *run a modified PCPP verifier $V'_{\text{ckt}}(S)$.* ◁
  execute PCPP verifier $V_{\text{ckt}}(S)$ to generate a query sequence $I = (i_1, \ldots, i_q)$ and a circuit $D\colon \{0,1\}^q \to \{0,1\}$.
  if $(f \circ g)|_I$ contains $\perp$ **then**
    accept.
  else if $\pi|_I$ does not contain $\perp$ **and** $D((f \circ g \circ \pi)|_I) = 1$ **then**
    accept.
  else
    reject.
else
  accept.

For any two strings $\alpha, \beta \in \{0,1\}^n$ such that $\mathsf{Enc}(\alpha) \circ \mathsf{Enc}(\beta) \in L_{\text{ckt}}(S)$, let $\Pi(\alpha,\beta) \in \{0,1\}^{p(n)}$ denote a polynomial-time computable proof that makes $V_{\text{ckt}}(S)$ to accept $\mathsf{Enc}(\alpha) \circ \mathsf{Enc}(\beta) \circ \Pi(\alpha,\beta)$ with probability 1. Note that $\mathsf{Enc}(0^n) \circ \mathsf{Enc}(0^n) \circ \Pi(0^n, 0^n)$ and $\mathsf{Enc}(1^n) \circ \mathsf{Enc}(1^n) \circ \Pi(1^n, 1^n)$ are accepted by $V(S)$ with probability 1.

5.2.2 Completeness and Soundness

We now prove the completeness and soundness. Define $\sigma^{\mathsf{start}} := \mathsf{Enc}(0^n) \circ \mathsf{Enc}(0^n) \circ \Pi(0^n, 0^n) \in \{0,1\}^{2\ell(n)+p(n)}$ and $\sigma^{\mathsf{goal}} := \mathsf{Enc}(1^n) \circ \mathsf{Enc}(1^n) \circ \Pi(1^n, 1^n) \in \{0,1\}^{2\ell(n)+p(n)}$. Let $\mathsf{val}_V(S)$ denote the maximum possible value of

$$\min_{\sigma^{(t)} \in \sigma} \mathbb{P}\mathrm{r}\Big[V(S) \text{ accepts } \sigma^{(t)}\Big] \tag{5.6}$$

over all possible reconfiguration sequences $\sigma = (\sigma^{(1)}, \ldots, \sigma^{(T)})$ from σ^{start} to σ^{goal}. The perfect completeness ensures that $\mathsf{val}_V(S) = 1$ if S is a YES instance, while the soundness guarantees that $\mathsf{val}_V(S) < 1 - \delta$ for some $\delta \in (0,1)$ if S is a NO instance.

We first show the completeness.

Lemma 5.4. *Suppose a circuit $S\colon \{0,1\}^n \to \{0,1\}^n$ is a YES instance of* Succinct Graph Reachability. *Then, there exists a reconfiguration sequence σ from σ^{start} to σ^{goal} over $\{0,1,\perp\}^{2\ell(n)+p(n)}$ such that $V(S)$ accepts any proof in σ with probability* 1.

Proof. It suffices to show that for any $\alpha \neq \beta \in \{0,1\}^n$ such that $\alpha = S(\beta)$ or $\beta = S(\alpha)$ (i.e., $\mathsf{Enc}(\alpha) \circ \mathsf{Enc}(\beta) \in L_{\text{ckt}}(S)$), there is a reconfiguration sequence σ from $\mathsf{Enc}(\alpha) \circ \mathsf{Enc}(\alpha) \circ \Pi(\alpha, \alpha)$

to $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\beta)\circ\Pi(\alpha,\beta)$ such that $V(S)$ accepts any proof in σ with probability 1. Such a reconfiguration sequence is obtained by the following procedure:

Reconfiguration σ from $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\alpha)\circ\Pi(\alpha,\alpha)$ to $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\beta)\circ\Pi(\alpha,\beta)$.

1: ▷ *start from* $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\alpha)\circ\Pi(\alpha,\alpha)$. ◁
2: change the second string from $\mathsf{Enc}(\alpha)$ to $\perp^{\ell(n)}$ one by one.
3: ▷ *obtain* $\mathsf{Enc}(\alpha)\circ\perp^{\ell(n)}\circ\Pi(\alpha,\alpha)$. ◁
4: change the proof from $\Pi(\alpha,\alpha)$ to $\Pi(\alpha,\beta)$ one by one.
5: ▷ *obtain* $\mathsf{Enc}(\alpha)\circ\perp^{\ell(n)}\circ\Pi(\alpha,\beta)$. ◁
6: change the second string from $\perp^{\ell(n)}$ to $\mathsf{Enc}(\beta)$.
7: ▷ *end at* $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\beta)\circ\pi^{(\alpha,\beta)}$. ◁

By the following case analysis, $V(S)$ turns out to accept every intermediate proof $f\circ g\circ\pi$ with probability 1, as desired.

- (Line 2) $f\circ g\circ\pi$ is obtained from $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\alpha)\circ\Pi(\alpha,\alpha)$ by replacing some symbols of the second $\mathsf{Enc}(\alpha)$ by $\perp$. Observe that the local tester M always accepts $f=\mathsf{Enc}(\alpha)$. We show that $V'_{\mathrm{ckt}}(S)$ always accepts $f\circ g\circ\pi$. Let $(I,D)\sim V_{\mathrm{ckt}}(S)$. If $(f\circ g)|_I$ contains $\perp$, $V'_{\mathrm{ckt}}(S)$ accepts. Otherwise, since $\pi=\Pi(\alpha,\alpha)$ does not contain $\perp$, it holds that $(f\circ g\circ\pi)|_I=(\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\alpha)\circ\Pi(\alpha,\alpha))|_I$, implying $D((f\circ g\circ\pi)|_I)=1$.
- (Line 4) $f\circ g\circ\pi$ has a form of $\mathsf{Enc}(\alpha)\circ\perp^{\ell(n)}\circ\pi$ for some $\pi\in\{0,1,\perp\}^{p(n)}$. The local tester M always accepts $\mathsf{Enc}(\alpha)$, and $V(S)$ would not have run the modified verifier $V'_{\mathrm{ckt}}(S)$; i.e., $V(S)$ always accepts $f\circ g\circ\pi$.
- (Line 6) $f\circ g\circ\pi$ is obtained from $\mathsf{Enc}(\alpha)\circ\mathsf{Enc}(\beta)\circ\Pi(\alpha,\beta)$ by replacing some symbols of $\mathsf{Enc}(\beta)$ by $\perp$. Similarly to the first case, we can show that $V(S)$ always accepts $f\circ g\circ\pi$.

□

We then show the soundness.

Lemma 5.5. *Suppose a circuit $S\colon\{0,1\}^n\to\{0,1\}^n$ is a NO instance of* Succinct Graph Reachability. *Then, for any reconfiguration sequence σ from σ^{start} to σ^{goal} over $\{0,1,\perp\}^{2\ell(n)+p(n)}$, σ includes a proof that is rejected by $V(S)$ with probability at least*

$$\min\left\{(\kappa\varepsilon)^2,(1-\varepsilon)^2\cdot\frac{1-s_{\mathrm{ckt}}}{2}\right\},\ \textit{where}\ \varepsilon:=\min\left\{\frac{1-s_{\mathrm{ckt}}}{2q},\frac{\rho}{3}\right\}. \tag{5.7}$$

By Lemmas 5.4 and 5.5, we can complete the proof of Theorem 5.1.

Proof of Theorem 5.1. We first prove the "only if" direction. Since Succinct Graph Reachability is PSPACE-complete, it is sufficient to create its verifier V and polynomial-time computable proofs π^{start} and π^{goal}. The verifier V is described in Section 5.2.1. For a polynomial-size circuit $S\colon\{0,1\}^n\to\{0,1\}$, the number of queries that V makes is bounded by $2\cdot(\#\text{ queries of }M)+2+(\#\text{ queries of }V_{\mathrm{ckt}})=\mathcal{O}(1)$, and the number of random bits that V

uses is bounded by $2 \cdot (\text{random bits of } M) + 2 \cdot (\log \ell(n)) + (\text{random bits of } V_{\text{ckt}}) = O(\log n)$. We define $\pi^{\text{start}} := \mathsf{Enc}(0^n) \circ \mathsf{Enc}(0^n) \circ \Pi(0^n, 0^n)$ and $\pi^{\text{goal}} := \mathsf{Enc}(1^n) \circ \mathsf{Enc}(1^n) \circ \Pi(1^n, 1^n)$, which are polynomial-time computable.

We reduce the alphabet size of V from three (i.e., $\{0,1,\perp\}$) to two. Using Proposition 4.9, we first convert $V(S)$ into a constraint system $\Psi = (\psi_j)_{j \in [m]}$ over alphabet $\{0,1,\perp\}$ such that $\mathbb{Pr}[V(S) \text{ accepts } \pi]$ is equal to $\mathsf{val}_\Psi(\pi)$ for any proof $\pi \in \{0,1,\perp\}^{\mathsf{poly}(n)}$. By [Ohs23b], we obtain a constraint system $\Psi' = (\psi'_j)_{j \in [m']}$ over alphabet $\{0,1\}$ and its two satisfying assignments A^{start} and A^{goal} such that $\mathsf{val}_\Psi(\pi^{\text{start}} \leftrightsquigarrow \pi^{\text{goal}}) = 1$ implies $\mathsf{val}_{\Psi'}(A^{\text{start}} \leftrightsquigarrow A^{\text{goal}}) = 1$, and $\mathsf{val}_\Psi(\pi^{\text{start}} \leftrightsquigarrow \pi^{\text{goal}}) < 1 - \varepsilon$ implies $\mathsf{val}_{\Psi'}(A^{\text{start}} \leftrightsquigarrow A^{\text{goal}}) < 1 - \Omega(\varepsilon)$. Using Proposition 4.9 again, we convert Ψ' into a verifier V' with randomness complexity $O(\log n)$, query complexity $O(1)$, and alphabet $\{0,1\}$ such that $\mathbb{Pr}[V' \text{ accepts } \pi']$ is equal to $\mathsf{val}_{\Psi'}(\pi')$ for any proof $\pi' \in \{0,1\}^{\mathsf{poly}(n)}$. Consequently, if S is a YES instance, by Lemma 5.4, there exists a reconfiguration sequence $\mathscr{A}$ from A^{start} to A^{goal} over $\{0,1\}^{\mathsf{poly}(n)}$ such that V' accepts any proof in $\mathscr{A}$ with probability 1, whereas if S is a NO instance, by Lemma 5.5, for any reconfiguration sequence $\mathscr{A}$ from A^{start} to A^{goal} over $\{0,1\}^{\mathsf{poly}(n)}$, $\mathscr{A}$ includes a proof that is rejected by V' with probability $\Omega(1)$, which can be amplified to $\frac{1}{2}$ by a constant number of repetition, as desired.

We then prove the "if" direction. Suppose a language L admits a verifier V with randomness complexity $r(n) = O(\log n)$ and query complexity $q(n) = O(1)$, associated with polynomial-time computable proofs π^{start} and π^{goal}. Consider then the following nondeterministic algorithm for finding a reconfiguration sequence from π^{start} to π^{goal}.

Nondeterministic polynomial-space algorithm for finding a reconfiguration sequence.

Input: $x \in \{0,1\}^*$.

compute proofs $\pi^{\text{start}}(x)$ and $\pi^{\text{goal}}(x)$ that are accepted by V with probability 1.
let $\pi^{(0)} := \pi^{\text{start}}(x)$ **and** $t \leftarrow 0$.
repeat
    if $\pi^{(t)} = \pi^{\text{goal}}(x)$ **then**
        accept.
    nondeterministically guess the next proof $\pi^{(t+1)} \in \{0,1\}^{\mathsf{poly}(|x|)}$.
    check if $\pi^{(t)}$ and $\pi^{(t+1)}$ differ in at most one bit, and V accepts $\pi^{(t+1)}$ with probability 1 by enumerating all possible $r(|x|)$ random bits.
    if the above test passes **then**
        forget $\pi^{(t)}$ **and** let $t \leftarrow t+1$.
    else
        reject.
until $t > 2^{\mathsf{poly}(|x|)}$
reject.

The above algorithm accepts x if and only if $x \in L$. Moreover, it requires polynomial space and terminates within a finite steps; namely, $L \in \mathsf{NPSPACE}$. By Savitch's theorem [Sav70], $L \in \mathsf{PSPACE}$. □

t	1	$\cdots$	t_1	t_1+1	$\cdots$	t_2-1	t_2	t_2+1	$\cdots$	t_3-1	t_3	t_3+1	$\cdots$	t_4-1	t_4	t_4+1	$\cdots$	$T-1$	T
$\alpha^{(t)}$	0^n	$\cdots$	0^n	$*$	$\cdots$	$*$	$\alpha^{(t_2)}$	$\alpha^{(t_2)}$	$\cdots$	$\alpha^{(t_2)}$	$\alpha^{(t_2)}$	$*$	$\cdots$	$*$	1^n	1^n	$\cdots$	1^n	1^n
$\beta^{(t)}$	0^n	$\cdots$	0^n	0^n	$\cdots$	0^n	0^n	$*$	$\cdots$	$*$	$\beta^{(t_3)}$	$\beta^{(t_3)}$	$\cdots$	$\beta^{(t_3)}$	$\beta^{(t_3)}$	$*$	$\cdots$	$*$	1^n

Table 1: Illustration of Claim 5.7, which finds a sequence γ from 0^n to 1^n over $\{0,1\}^n$ using $((\alpha^{(1)},\beta^{(1)}),\ldots,(\alpha^{(T)},\beta^{(T)}))$. Colored strings are included in γ, resulting in $\gamma=(0^n,\ldots,0^n,\alpha^{(t_2)},\ldots,\alpha^{(t_2)},\beta^{(t_3)},\ldots,\beta^{(t_3)},1^n,\ldots,1^n)$. If an input circuit S is a NO instance, at least one of $\mathsf{Enc}(\alpha^{(t_2)})\circ\mathsf{Enc}(0^n)$, $\mathsf{Enc}(\alpha^{(t_2)})\circ\mathsf{Enc}(\beta^{(t_3)})$, or $\mathsf{Enc}(\beta^{(t_3)})\circ\mathsf{Enc}(1^n)$ is not in $L_{\mathrm{ckt}}(S)$.

The remainder of this section is devoted to the proof of Lemma 5.5.

Proof of Lemma 5.5. Suppose we are given a reconfiguration sequence $\sigma=(\sigma^{(1)},\ldots,\sigma^{(T)})=(f^{(1)}\circ g^{(1)}\circ\pi^{(1)},\ldots,f^{(T)}\circ g^{(T)}\circ\pi^{(T)})$ from σ^{start} to σ^{goal} such that $\mathrm{val}_V(S)=\min_{t\in[T]}\mathbb{P}\mathrm{r}[V(S)\text{ accepts }\sigma^{(t)}]$. Define

$$\varepsilon:=\min\left\{\frac{1-s_{\mathrm{ckt}}}{2q},\frac{\rho}{3}\right\},\tag{5.8}$$

where q is the query complexity of V_{ckt}, s_{ckt} is the soundness error of V_{ckt}, and ρ is the relative distance of Enc. Observe that if *both* $f^{(t)}$ and $g^{(t)}$ for some $t\in[T]$ are ε-far from $\mathsf{Enc}(\cdot)$, then M rejects each $f^{(t)}$ and $g^{(t)}$ with probability more than $\kappa\varepsilon$; namely,

$$\mathbb{P}\mathrm{r}\Big[V(S)\text{ rejects }\sigma^{(t)}\Big]\geqslant\mathbb{P}\mathrm{r}\Big[M\text{ rejects }f\Big]\cdot\mathbb{P}\mathrm{r}\Big[M\text{ rejects }g\Big]>(\kappa\varepsilon)^2.\tag{5.9}$$

Hereafter, we assume that $f^{(t)}$ or $g^{(t)}$ is ε-close to $\mathsf{Enc}(\cdot)$ for every $t\in[T]$.

We then define $\mathsf{Dec}\colon\{0,1\}^{\ell(n)}\to\{0,1\}^n\cup\{*\}$ as

$$\mathsf{Dec}(f):=\begin{cases}\operatorname*{argmin}_{\alpha\in\{0,1\}^n}\Delta(f,\mathsf{Enc}(\alpha)) & \text{if } f \text{ is } \varepsilon\text{-close to } \mathsf{Enc}(\cdot),\\ * & \text{otherwise,}\end{cases}\tag{5.10}$$

where $*$ means "undefined". Using Dec, we obtain a sequence from $(0^n,0^n)$ to $(1^n,1^n)$, denoted $((\alpha^{(1)},\beta^{(1)}),\ldots,(\alpha^{(T)},\beta^{(T)}))$, where $\alpha^{(t)}:=\mathsf{Dec}(f^{(t)})$ and $\beta^{(t)}:=\mathsf{Dec}(g^{(t)})$ for all $t\in[T]$. By assumption, $\alpha^{(t)}$ or $\beta^{(t)}$ must not be $*$ for all $t\in[T]$. We claim the following (see also Table 1):

Claim 5.6. *The following hold:*

(P1) $\alpha^{(t)}=\alpha^{(t+1)}$ *or* $\beta^{(t)}=\beta^{(t+1)}$ *for each* t.

(P2) *If* $\alpha^{(t)}\neq*$ *and* $\alpha^{(t+1)}\neq*$, *then* $\alpha^{(t)}=\alpha^{(t+1)}$.

(P3) *If* $\beta^{(t)}\neq*$ *and* $\beta^{(t+1)}\neq*$, *then* $\beta^{(t)}=\beta^{(t+1)}$.

Proof. Suppose first $\alpha^{(t)}\neq\alpha^{(t+1)}$ and $\beta^{(t)}\neq\beta^{(t+1)}$ for some t. Then, $f^{(t)}\circ g^{(t)}$ and $f^{(t+1)}\circ g^{(t+1)}$ differ in at least two symbols, contradicting that σ is a reconfiguration sequence; thus, (P1) must hold.

Suppose then $\alpha^{(t)} \neq *$, $\alpha^{(t+1)} \neq *$, and $\alpha^{(t)} \neq \alpha^{(t+1)}$. Since $f^{(t)}$ and $f^{(t+1)}$ are assumed to be ε-close to $\mathsf{Enc}(\cdot)$, by triangle inequality, we have

$$\begin{aligned} \Delta(f^{(t)}, f^{(t+1)}) &\geqslant \underbrace{\Delta(\mathsf{Enc}(\alpha^{(t)}), \mathsf{Enc}(\alpha^{(t+1)}))}_{\geqslant \rho} - \underbrace{\Delta(f^{(t)}, \mathsf{Enc}(\alpha^{(t)}))}_{\leqslant \varepsilon} - \underbrace{\Delta(f^{(t+1)}, \mathsf{Enc}(\alpha^{(t+1)}))}_{\leqslant \varepsilon} \\ &\geqslant \rho - 2\varepsilon, \end{aligned} \tag{5.11}$$

implying that $f^{(t)}$ and $f^{(t+1)}$ differ in $(\rho - 2\varepsilon) \cdot \ell(n) \geqslant 2$ bits (for sufficiently large n), contradicting that σ is a reconfiguration sequence; thus, (P2) holds. Similarly, (P3) can be shown. □

Now, we can find a *valid* sequence from 0^n to 1^n over $\{0,1\}^n$, denoted $\gamma = (\gamma^{(1)}, \ldots, \gamma^{(T')})$, along a "path" over a grid $\{\alpha, \beta\} \times [T]$ by the following procedure (see also Table 1):

Sequence γ from 0^n to 1^n.

let $\alpha^{(T+1)} := 1^n$ **and** $\beta^{(T+1)} := 1^n$ for convenience.
let $t' \leftarrow 0$ **and** place a token at $(\alpha, 1)$.
repeat
  let $t' \leftarrow t' + 1$.
  if token is at (α, t) **then**
    let $\gamma^{(t')} := \alpha^{(t)}$.
    if $\alpha^{(t+1)} \neq *$ **then** token goes to $(\alpha, t+1)$.
    else token goes to (β, t).
  if token is at (β, t) **then**
    let $\gamma^{(t')} := \beta^{(t)}$.
    if $\beta^{(t+1)} \neq *$ **then** token goes to $(\beta, t+1)$.
    else token goes to (α, t).
until token is at $(\alpha, T+1)$ or $(\beta, T+1)$
return $\gamma := (\gamma^{(1)}, \ldots, \gamma^{(t')})$.

The correctness of the above procedure is shown in the following claim.

Claim 5.7. *The above procedure terminates and returns a sequence $\gamma = (\gamma^{(1)}, \ldots, \gamma^{(T')})$ from 0^n to 1^n consisting only of strings of $\{0,1\}^n$. Moreover, each pair $(\gamma^{(t')}, \gamma^{(t'+1)})$ is equal to either $(\alpha^{(t)}, \alpha^{(t+1)})$, $(\beta^{(t)}, \beta^{(t+1)})$, $(\alpha^{(t)}, \beta^{(t)})$, or $(\beta^{(t)}, \alpha^{(t)})$ for some t.*

Proof. (P1) of Claim 5.6 ensures that

- if $\alpha^{(t)} \neq *$, then $\alpha^{(t+1)} \neq *$ or $\beta^{(t)} \neq *$;[4]
- if $\beta^{(t)} \neq *$, then $\beta^{(t+1)} \neq *$ or $\alpha^{(t)} \neq *$.

Thus, by construction, γ does not include any $*$.

[4] Because otherwise, we have $\alpha^{(t)} \neq *$ and $\alpha^{(t+1)} = \beta^{(t)} = *$, which implies $\beta^{(t+1)} = *$ by (P1), a contradiction that $\alpha^{(t+1)}$ or $\beta^{(t+1)}$ is not $*$.

Suppose the token is currently placed at (α, t) and just before at (β, t). Then, the token must be placed at $(\alpha, t+1)$ in the next step. Similarly, if the token is placed at (β, t) and just before at (α, t); then, it must be at $(\beta, t+1)$ in the next step, which ensures that we eventually reach $(\alpha, T+1)$ or $(\beta, T+1)$ to terminate. The latter statement is obvious from the construction. □

Since we have been given a NO instance, γ must include $(\gamma^{(t')}, \gamma^{(t'+1)})$ for some $t' \in [T']$ such that $\gamma^{(t')} \neq \gamma^{(t'+1)}$, $S(\gamma^{(t')}) \neq \gamma^{(t'+1)}$, and $S(\gamma^{(t'+1)}) \neq \gamma^{(t')}$. By (P2) and (P3) and Claim 5.7, either of $(\alpha^{(t)}, \beta^{(t)}) = (\gamma^{(t')}, \gamma^{(t'+1)})$ or $(\beta^{(t)}, \alpha^{(t)}) = (\gamma^{(t')}, \gamma^{(t'+1)})$ must hold for some $t \in [T]$, implying that $\alpha^{(t)} \neq \beta^{(t)}$, $S(\alpha^{(t)}) \neq \beta^{(t)}$, and $S(\beta^{(t)}) \neq \alpha^{(t)}$; namely, $\mathsf{Enc}(\alpha^{(t)}) \circ \mathsf{Enc}(\beta^{(t)}) \notin L_{\mathrm{ckt}}(S)$.

We now estimate the probability that $V(S)$ rejects $\sigma^{(t)}$. Recall that $f^{(t)}$ and $g^{(t)}$ are ε-close to $\mathsf{Enc}(\cdot)$, implying that

$$\Pr_{i,j \sim [n]}\left[f_i^{(t)} \neq \perp \text{ and } g_j^{(t)} \neq \perp\right] \geqslant (1-\varepsilon)^2. \tag{5.12}$$

So, we would run the modified PCPP verifier V'_{ckt} with probability $\geqslant (1-\varepsilon)^2$. We use the following claim to bound the rejection probability of $V'_{\mathrm{ckt}}(S)$:

Claim 5.8. *Suppose $\alpha = \mathsf{Dec}(f) \in \{0,1\}^n$, $\beta = \mathsf{Dec}(g) \in \{0,1\}^n$, and $\mathsf{Enc}(\alpha) \circ \mathsf{Enc}(\beta) \notin L_{\mathrm{ckt}}(S)$ for $f \circ g \in \{0,1,\perp\}^{2\ell(n)}$. Then, for every proof $\pi \in \{0,1,\perp\}^{p(n)}$, the modified PCPP verifier $V'_{\mathrm{ckt}}(S)$ rejects $f \circ g \circ \pi$ with probability more than $1 - s_{\mathrm{ckt}} - \varepsilon q$.*

Proof. We first show that $(f \circ g)|_I$ contains $\perp$ for $(I, D) \sim V_{\mathrm{ckt}}(S)$ with probability at most εq. Denote by I_{ckt} the indices of $f \circ g \circ \pi$, where $|I_{\mathrm{ckt}}| = 2\ell(n) + p(n)$. By smoothness of V_{ckt}, we have $p_{\mathrm{ckt}} := \Pr_{(I,D)}[i \in I]$ for all $i \in I_{\mathrm{ckt}}$. Since $|I| = q$ for any $(I, D) \sim V_{\mathrm{ckt}}(S)$, we obtain

$$|I_{\mathrm{ckt}}| \cdot p_{\mathrm{ckt}} = \sum_{i \in I_{\mathrm{ckt}}} \Pr_{(I,D)}\left[i \in I\right] \leqslant q \implies p_{\mathrm{ckt}} \leqslant \frac{q}{2\ell(n) + p(n)}. \tag{5.13}$$

Using a union bound and the assumption that each f and g contains $\perp$ in at most $\varepsilon \cdot \ell(n)$ positions, we derive

$$\begin{aligned}\Pr_{(I,D)}\left[(f \circ g)_I \text{ contains } \perp\right] &= \Pr_{(I,D)}\left[\exists i \in I \text{ s.t. } (f \circ g)_i = \perp\right] \\ &\leqslant \sum_{i : (f \circ g)_i = \perp} \Pr_{(I,D)}\left[i \in I\right] \\ &\leqslant \varepsilon \cdot 2\ell(n) \cdot p_{\mathrm{ckt}} \leqslant \varepsilon q.\end{aligned} \tag{5.14}$$

Subsequently, we show that $f \circ g$ is δ_{ckt}-far from $L_{\mathrm{ckt}}(S)$, where $\delta_{\mathrm{ckt}} = \frac{\rho}{4}$. Letting $(\alpha^\star, \beta^\star) \in \{0,1\}^n \times \{0,1\}^n$ such that $\mathsf{Enc}(\alpha^\star) \circ \mathsf{Enc}(\beta^\star) \in L_{\mathrm{ckt}}(S)$, we have $\alpha \neq \alpha^\star$ or $\beta \neq \beta^\star$. Suppose $\alpha \neq \alpha^\star$; then, we have

$$\Delta(f, \mathsf{Enc}(\alpha^\star)) \geqslant \Delta(\mathsf{Enc}(\alpha^\star), \mathsf{Enc}(\alpha)) - \Delta(\mathsf{Enc}(\alpha), f) \geqslant \rho - \varepsilon. \tag{5.15}$$

Similarly, $\Delta(g, \mathsf{Enc}(\beta^\star)) \geqslant \rho - \varepsilon$ if $\beta \neq \beta^\star$. Consequently, we obtain

$$\Delta(f \circ g, \mathsf{Enc}(\alpha^\star) \circ \mathsf{Enc}(\beta^\star)) \geqslant \frac{\rho - \varepsilon}{2} > \frac{\rho}{4} = \delta_{\mathrm{ckt}}, \tag{5.16}$$

where we used the fact that $\varepsilon \leqslant \frac{\rho}{3}$.

Taking a union bound, we derive

$$\begin{aligned}
&\mathbb{P}\mathrm{r}\Big[\text{modified verifier } V'_{\mathrm{ckt}}(S) \text{ accepts } f \circ g \circ \pi\Big] \\
&= \mathop{\mathbb{P}\mathrm{r}}_{(I,D)}\Big[(f \circ g)|_I \text{ contains } \perp \text{ or } \Big(\pi|_I \text{ doesn't contain } \perp \text{ and } D((f \circ g \circ \pi)|_I) = 1\Big)\Big] \\
&= \mathop{\mathbb{P}\mathrm{r}}_{(I,D)}\Big[(f \circ g)|_I \text{ contains } \perp \text{ or } \Big((f \circ g \circ \pi)|_I \text{ doesn't contain } \perp \text{ and } D((f \circ g \circ \pi)|_I) = 1\Big)\Big] \\
&\leqslant \underbrace{\mathop{\mathbb{P}\mathrm{r}}_{(I,D)}\Big[(f \circ g)|_I \text{ contains } \perp\Big]}_{\leqslant \varepsilon q} + \mathop{\mathbb{P}\mathrm{r}}_{(I,D)}\Big[(f \circ g \circ \pi)|_I \text{ doesn't contain } \perp \text{ and } D((f \circ g \circ \pi)|_I) = 1\Big].
\end{aligned} \tag{5.17}$$

Let $\widetilde{\pi}$ be a proof obtained from π by replacing every occurrence of $\perp$ by 0. If $(f \circ g \circ \pi)|_I$ does not contain $\perp$ and $D((f \circ g \circ \pi)|_I) = 1$, then $D((f \circ g \circ \widetilde{\pi})|_I) = 1$. Since $f \circ g$ is δ_{ckt}-far from $L_{\mathrm{ckt}}(S)$, we have

$$\begin{aligned}
&\mathop{\mathbb{P}\mathrm{r}}_{(I,D)}\Big[(f \circ g \circ \pi)|_I \text{ doesn't contain } \perp \text{ and } D((f \circ g \circ \pi)|_I) = 1\Big] \\
&\leqslant \mathop{\mathbb{P}\mathrm{r}}_{(I,D)}\Big[D((f \circ g \circ \widetilde{\pi})|_I) = 1\Big] \\
&= \mathbb{P}\mathrm{r}\Big[V_{\mathrm{ckt}}(S) \text{ accepts } f \circ g \circ \widetilde{\pi}\Big] \\
&< s_{\mathrm{ckt}}.
\end{aligned} \tag{5.18}$$

Accordingly, we get

$$\mathbb{P}\mathrm{r}\Big[V'_{\mathrm{ckt}}(S) \text{ rejects } f \circ g \circ \pi\Big] = 1 - \mathbb{P}\mathrm{r}\Big[V'_{\mathrm{ckt}}(S) \text{ accepts } f \circ g \circ \pi\Big] > 1 - s_{\mathrm{ckt}} - \varepsilon q, \tag{5.19}$$

completing the proof. □

Using Claim 5.8 and the definition of ε in Eq. (5.8), we derive

$$\begin{aligned}
\mathbb{P}\mathrm{r}\Big[V(S) \text{ rejects } \sigma^{(t)}\Big] &\geqslant \mathop{\mathbb{P}\mathrm{r}}_{i,j \sim [\ell(n)]}\Big[f_i^{(t)} \neq \perp \text{ and } g_j^{(t)} \neq \perp\Big] \cdot \mathbb{P}\mathrm{r}\Big[V'_{\mathrm{ckt}}(S) \text{ rejects } \sigma^{(t)}\Big] \\
&> (1-\varepsilon)^2 \cdot (1 - s_{\mathrm{ckt}} - \varepsilon q) \\
&\underbrace{\geqslant}_{\varepsilon \leqslant \frac{1-s_{\mathrm{ckt}}}{2q}} (1-\varepsilon)^2 \cdot \frac{1 - s_{\mathrm{ckt}}}{2},
\end{aligned} \tag{5.20}$$

Consequently, we get

$$\max_{t \in [T]} \mathbb{P}\mathrm{r}\Big[V(S) \text{ rejects } \sigma^{(t)}\Big] > \min\left\{(\kappa\varepsilon)^2, (1-\varepsilon)^2 \cdot \frac{1 - s_{\mathrm{ckt}}}{2}\right\}, \tag{5.21}$$

accomplishing the proof of Lemma 5.5. □

5.3 Impossibility of Extension to Average Case

Since the verifier of Theorem 5.1 co-nondeterministically guesses $t \in [T]$ and probabilistically checks $\pi^{(t)}$, one might think of extending it so as to choose $t \in [T]$ randomly. The soundness case then requires that the verifier accepts "most" of but rejects a constant fraction of the proofs in any reconfiguration sequence. The resulting reconfiguration proof $(\pi^{(1)}, \ldots, \pi^{(T)})$ can be thought of as a (kind of) *rectangular PCPs* [BHPT20], whose column is of exponential length and row is of polynomial length, and we pick $t \in [T]$ and run a verifier on $\pi^{(t)}$ to decide whether to accept. However, such relaxation is impossible, as formally stated below.

Observation 5.9. *Let V be a verifier with randomness complexity $r(n) = O(\log n)$ and query complexity $q(n) = O(1)$, $x \in \{0,1\}^n$ be an input of length n, and $\pi^{\mathsf{start}}(x), \pi^{\mathsf{goal}}(x) \in \{0,1\}^{\mathsf{poly}(n)}$ be a pair of proofs that are accepted by V with probability* 1. *Then, there always exists a reconfiguration sequence $\pi = (\pi^{(1)}, \ldots, \pi^{(T)})$ from $\pi^{\mathsf{start}}(x)$ to $\pi^{\mathsf{goal}}(x)$ over $\{0,1\}^{\mathsf{poly}(n)}$ such that*

$$\Pr_{t \sim [T]}\left[V \text{ accepts } \pi^{(t)}\right] = \Pr_{\substack{t \sim [T] \\ (I,D) \sim V(x)}}\left[D(\pi^{(t)}|_I) = 1\right] > 1 - \frac{1}{2^{\Omega(n)}}, \tag{5.22}$$

where the probability is over the $r(n)$ random bits of V and $t \sim [T]$.

Proof. Consider any reconfiguration sequence $(\pi^{(1)}, \ldots \pi^{(\mathsf{poly}(n)+1)})$ from $\pi^{\mathsf{start}}(x)$ to $\pi^{\mathsf{goal}}(x)$. By appending $(\underbrace{\pi^{\mathsf{goal}}(x), \ldots, \pi^{\mathsf{goal}}(x)}_{2^n - (\mathsf{poly}(n)+1) \text{ times}})$ to it, we obtain an exponential-length reconfiguration sequence $\pi = (\pi^{(1)}, \ldots, \pi^{(2^n)})$. Since π contains (at least) $2^n - (\mathsf{poly}(n)+1)$ number of $\pi^{\mathsf{goal}}(x)$, we have

$$\Pr_{t \sim [T]}\left[V \text{ accepts } \pi^{(t)}\right] \geqslant \frac{2^n - (\mathsf{poly}(n)+1)}{2^n} = 1 - \frac{1}{2^{\Omega(n)}}, \tag{5.23}$$

as desired. $\square$

6 PSPACE-hardness of Approximation for Reconfiguration Problems

In this section, we show that many popular reconfiguration problems are PSPACE-hard to approximate, answering an open problem of [IDHPSUU11]. Since Ohsaka [Ohs23b] has already shown gap-preserving reductions starting from the Reconfiguration Inapproximability Hypothesis (RIH), which asserts that a gap version of MAXMIN CSP RECONFIGURATION is PSPACE-hard, we prove that RIH is true.

6.1 Constant-factor Inapproximability of MAXMIN CSP RECONFIGURATION

We first define reconfiguration problems on constraint satisfaction. For a q-ary constraint system $\Psi = (\psi_j)_{j \in [m]}$ over variable set N and alphabet Σ and its two satisfying assignments A^{start} and A^{goal} for Ψ, a *reconfiguration sequence from A^{start} to A^{goal} over Σ^N* is a sequence $(A^{(1)},\ldots,A^{(T)}) \in (\Sigma^N)^*$ such that $A^{(1)} = A^{\mathsf{start}}$, $A^{(T)} = A^{\mathsf{goal}}$, and $A^{(t)}$ and $A^{(t+1)}$ differ in at most one vertex for every $t \in [T-1]$. In the q-CSP RECONFIGURATION problem, we are asked to decide if there is a reconfiguration sequence of satisfying assignments to Ψ from A^{start} to A^{goal}. Subsequently, we formulate an approximate variant of q-CSP RECONFIGURATION [IDHPSUU11, Ohs23b]. For a reconfiguration sequence $\mathscr{A} = (A^{(1)},\ldots,A^{(T)})$ of assignments, let $\mathsf{val}_\Psi(\mathscr{A})$ denote the *minimum fraction* of satisfied constraints over all $A^{(i)}$'s in $\mathscr{A}$; namely,

$$\mathsf{val}_\Psi(\mathscr{A}) := \min_{A^{(i)}\in\mathscr{A}} \mathsf{val}_\Psi(A^{(i)}). \tag{6.1}$$

In MAXMIN q-CSP RECONFIGURATION, we wish to maximize $\mathsf{val}_\Psi(\mathscr{A})$ subject to $\mathscr{A} = (A^{\mathsf{start}},\ldots,A^{\mathsf{goal}})$. For two assignments $A^{\mathsf{start}}, A^{\mathsf{goal}} : N \to \Sigma$, let $\mathsf{val}_\Psi(A^{\mathsf{start}} \leftrightsquigarrow A^{\mathsf{goal}})$ denote the maximum value of $\mathsf{val}_\Psi(\mathscr{A})$ over all possible reconfiguration sequences $\mathscr{A}$ from A^{start} to A^{goal}; namely,

$$\mathsf{val}_\Psi(A^{\mathsf{start}} \leftrightsquigarrow A^{\mathsf{goal}}) := \max_{\mathscr{A}=(A^{\mathsf{start}},\ldots,A^{\mathsf{goal}})} \mathsf{val}_\Psi(\mathscr{A}) = \max_{\mathscr{A}=(A^{\mathsf{start}},\ldots,A^{\mathsf{goal}})} \min_{A^{(i)}\in\mathscr{A}} \mathsf{val}_\Psi(A^{(i)}). \tag{6.2}$$

For every numbers $0 \leqslant s \leqslant c \leqslant 1$, GAP$_{c,s}$ q-CSP RECONFIGURATION requests to determine for a q-ary constraint system Ψ and its two assignments A^{start} and A^{goal}, whether $\mathsf{val}_\Psi(A^{\mathsf{start}} \leftrightsquigarrow A^{\mathsf{goal}}) \geqslant c$ or $\mathsf{val}_\Psi(A^{\mathsf{start}} \leftrightsquigarrow A^{\mathsf{goal}}) < s$. As a corollary of Theorem 5.1 and Proposition 4.9, we immediately obtain a proof of RIH, as formally stated below.

Theorem 6.1. *The Reconfiguration Inapproximability Hypothesis holds; that is, there exist a universal constant $q \in \mathbb{N}$ such that* GAP$_{1,\frac{1}{2}}$ q-CSP RECONFIGURATION *with alphabet size* 2 *is* PSPACE*-complete.*

By Theorem 6.1 and [Ohs23a, Ohs23b], a host of reconfiguration problems turn out to be PSPACE-hard to approximate within a constant factor, as listed in Corollary 1.6.

6.2 Polynomial-factor Inapproximability of MAXMIN CLIQUE RECONFIGURATION

We amplify inapproximability of MAXMIN CLIQUE RECONFIGURATION from a constant factor to a polynomial factor. The proof of the following result uses the *derandomized graph product* due to Alon, Feige, Wigderson, and Zuckerman [AFWZ95]; see also [HLW06, Section 3.3.2] and [AB09, Example 22.7].

Theorem 6.2. *There exists a constant $\varepsilon \in (0,1)$ such that* MAXMIN CLIQUE RECONFIGURATION *is* PSPACE*-hard to approximate within a factor of n^ε, where n is the number of vertices.*

We here formulate CLIQUE RECONFIGURATION and its approximate variant. Denote by $\omega(G)$ the clique number of a graph G. For a pair of cliques C^{start} and C^{goal} of a graph G, a *reconfiguration sequence from C^{start} to C^{goal}* is a sequence $(C^{(1)},\ldots,C^{(T)})$ of cliques of G such that $C^{(1)} = C^{\mathsf{start}}$, $C^{(T)} = C^{\mathsf{goal}}$, and $C^{(t)}$ and $C^{(t+1)}$ differ in at most one vertex; i.e.,

$|C^{(t)} \Delta C^{(t+1)}| = 1$.[5] Clique Reconfiguration asks if there is a reconfiguration sequence from C^{start} to C^{goal} made up of cliques only of size at least $\min\{|C^{\mathsf{start}}|, |C^{\mathsf{goal}}|\} - 1$. For a reconfiguration sequence of cliques of G, denoted $\mathscr{C} = (C^{(1)}, \ldots, C^{(T)})$, let

$$\mathsf{val}_G(\mathscr{C}) := \min_{C^{(i)} \in \mathscr{C}} |C^{(i)}|. \tag{6.3}$$

Then, for a pair of cliques C^{start} and C^{goal} of G, Maxmin Clique Reconfiguration requires to maximize $\mathsf{val}_G(\mathscr{C})$ subject to $\mathscr{C} = (C^{\mathsf{start}}, \ldots, C^{\mathsf{goal}})$. Subsequently, let $\mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}})$ denote the maximum value of $\mathsf{val}_G(\mathscr{C})$ over all possible reconfiguration sequences $\mathscr{C}$ from C^{start} to C^{goal}; namely,

$$\mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}}) := \max_{\mathscr{C} = (C^{\mathsf{start}}, \ldots, C^{\mathsf{goal}})} \mathsf{val}_G(\mathscr{C}). \tag{6.4}$$

Reduction. We first describe a gap-amplification reduction from Maxmin Clique Reconfiguration to itself using the derandomized graph product [AFWZ95]. Let $(G, C^{\mathsf{start}}, C^{\mathsf{goal}})$ be an instance of Maxmin Clique Reconfiguration, where $G = (V, E)$ is a graph on n vertices. By Theorem 6.1 and [Ohs23b], it is PSPACE-hard to distinguish whether $\mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}}) \geqslant \omega(G) - 1$ or $\mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}}) \geqslant (1-\varepsilon)(\omega(G) - 1)$ for some constant $\varepsilon \in (0,1)$ even when $|C^{\mathsf{start}}| = |C^{\mathsf{goal}}| = \omega(G)$ and $\frac{\omega(G)}{n} \in \left[\frac{1}{3}, \frac{1}{2}\right]$.

Construct then a new instance $(H, D^{\mathsf{start}}, D^{\mathsf{goal}})$ of Maxmin Clique Reconfiguration as follows. Let $\ell = \lceil \log n \rceil$, and X be a (d, λ)-expander graph over the same vertex as G. The precise value of d and λ will be determined later. Graph $H = (W, F)$ is defined as follows:

- **Vertex set:** W is the set consisting of all length-$(\ell - 1)$ walks $\mathbf{w} = (w_1, \ldots, w_\ell)$ over X. Note that the number of vertices is equal to $N := |W| = nd^{\ell-1}$, which is polynomial in n.
- **Edge set:** H contains an edge between $\mathbf{w}_1 \neq \mathbf{w}_2 \in W$ if and only if a subgraph of G induced by $\mathbf{w}_1 \cup \mathbf{w}_2$ forms a clique.

For any clique $C \subseteq V$ of G, define $D_C \subseteq W$ as

$$D_C := \left\{ \mathbf{w} \in W \;\middle|\; \mathbf{w} \subseteq C \right\}, \tag{6.5}$$

which is a clique of H as well. Constructing $D^{\mathsf{start}} := D_{C^{\mathsf{start}}}$ and $D^{\mathsf{goal}} := D_{C^{\mathsf{goal}}}$ completes the reduction. We refer to the following property about random walks over expander graphs.

Lemma 6.3 ([AFWZ95]). *Let S be any vertex set of X, and $\mathbf{Z} := (Z_1, \ldots, Z_\ell)$ a ℓ-tuple of random variables denoting the vertices of a uniformly chosen $(\ell - 1)$-length random walk over X. Then, it holds that*

$$\left(\frac{|S|}{|V|} - 2\frac{\lambda}{d} \right)^\ell \leqslant \mathop{\mathbb{P}\mathrm{r}}_{\mathbf{Z}}\left[\forall i \in [\ell],\, Z_i \in S \right] \leqslant \left(\frac{|S|}{|V|} + 2\frac{\lambda}{d} \right)^\ell. \tag{6.6}$$

[5] Such a model of reconfiguration is called *token addition and removal* [IDHPSUU11].

The completeness and soundness are shown below.

Lemma 6.4. *If* $\mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}}) \geqslant \omega(G)-1$*, then*

$$\mathsf{val}_H(D^{\mathsf{start}} \leftrightsquigarrow D^{\mathsf{goal}}) \geqslant |W| \cdot \left(\frac{\omega(G)-1}{|V|} - 2\frac{\lambda}{d}\right)^{\ell}. \tag{6.7}$$

Proof. It suffices to consider the case that C^{start} and C^{goal} differ in exactly two vertices (i.e., C^{goal} is obtained from C^{start} by removing and adding a single vertex). There is a reconfiguration sequence $(C^{\mathsf{start}}, C^{\circ}, C^{\mathsf{goal}})$ from C^{start} to C^{goal}, where $C^{\circ} := C^{\mathsf{start}} \cap C^{\mathsf{goal}}$ is a clique of size $\omega(G)-1$. Since $D_{C^{\mathsf{start}}} \supset D_{C^{\circ}}$ and $D_{C^{\mathsf{goal}}} \supset D_{C^{\circ}}$ by definition, we can reconfigure from $D^{\mathsf{start}} = D_{C^{\mathsf{start}}}$ to $D^{\mathsf{goal}} = D_{C^{\mathsf{goal}}}$ by first removing the vertices of $D_{C^{\mathsf{start}}} \setminus D_{C^{\circ}}$ one by one and then adding the vertices of $D_{C^{\mathsf{goal}}} \setminus D_{C^{\circ}}$ one by one. Thus, we have $\mathsf{val}_H(D^{\mathsf{start}} \leftrightsquigarrow D^{\mathsf{goal}}) \geqslant |D_{C^{\circ}}|$. Lemma 6.3 derives that

$$\frac{|D_{C^{\circ}}|}{|W|} = \Pr_{\mathbf{Z}}\left[\forall i \in [\ell],\, Z_i \in C^{\circ}\right] \geqslant \left(\frac{|C^{\circ}|}{|V|} - 2\frac{\lambda}{d}\right)^{\ell} = \left(\frac{\omega(G)-1}{|V|} - 2\frac{\lambda}{d}\right)^{\ell}, \tag{6.8}$$

completing the proof. □

Lemma 6.5. *If* $\mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}}) < (1-\varepsilon)(\omega(G)-1)$*, then*

$$\mathsf{val}_H(D^{\mathsf{start}} \leftrightsquigarrow D^{\mathsf{goal}}) < |W| \cdot \left((1-\varepsilon)\frac{\omega(G)-1}{|V|} + 2\frac{\lambda}{d}\right)^{\ell}. \tag{6.9}$$

Proof. We show the contrapositive. Suppose we are given a reconfiguration sequence $\mathscr{D} = (D^{(1)}, \ldots, D^{(T)})$ from D^{start} to D^{goal} such that $\mathsf{val}_H(\mathscr{D}) \geqslant |W| \cdot \left((1-\varepsilon)\frac{\omega(G)-1}{|V|} + 2\frac{\lambda}{d}\right)^{\ell}$. For any clique D of H, define C_D as

$$C_D := \bigcup_{\mathbf{w} \in D} \mathbf{w}, \tag{6.10}$$

which is a clique of G as well. Observe that $\mathsf{val}_G(C_{D^{(t)}} \leftrightsquigarrow C_{D^{(t+1)}}) \geqslant \min\{|C_{D^{(t)}}|, |C_{D^{(t+1)}}|\}$ for any $t \in [T-1]$ since $C_{D^{(t)}} \subset C_{D^{(t+1)}}$ or $C_{D^{(t)}} \supset C_{D^{(t+1)}}$, implying further that

$$\begin{aligned} \mathsf{val}_G(C^{\mathsf{start}} \leftrightsquigarrow C^{\mathsf{goal}}) &\geqslant \min_{t \in [T-1]} \mathsf{val}_G(C_{D^{(t)}} \leftrightsquigarrow C_{D^{(t+1)}}) \\ &\geqslant \min_{t \in [T-1]} \min\left\{|C_{D^{(t)}}|, |C_{D^{(t+1)}}|\right\} \\ &\geqslant \min_{D^{(t)} \in \mathscr{D}} |C_{D^{(t)}}|. \end{aligned} \tag{6.11}$$

Lemma 6.3 derives that

$$\frac{|D^{(t)}|}{|W|} = \Pr_{\mathbf{Z}}\left[\mathbf{Z} \in D^{(t)}\right] \leqslant \Pr_{\mathbf{Z}}\left[\forall i \in [\ell],\, Z_i \in C_{D^{(t)}}\right] \leqslant \left(\frac{|C_{D^{(t)}}|}{|V|} + 2\frac{\lambda}{d}\right)^{\ell}. \tag{6.12}$$

On the other hand, by assumption,

$$\frac{|D^{(t)}|}{|W|} \geqslant \left((1-\varepsilon)\frac{\omega(G)-1}{|V|} + 2\frac{\lambda}{d}\right)^{\ell}. \tag{6.13}$$

Consequently, we have $|C_{D^{(t)}}| \geqslant (1-\varepsilon)(\omega(G)-1)$ for all $t \in [T]$; thus, $\mathrm{val}_G(C^{\mathrm{start}} \leftrightsquigarrow C^{\mathrm{goal}}) \geqslant (1-\varepsilon)(\omega(G)-1)$, as desired. □

We are now ready to accomplish the proof of Theorem 6.2.

Proof of Theorem 6.2. Letting X satisfy $\frac{\lambda}{d} < \frac{\varepsilon}{32}$ so that $\frac{\lambda}{d} < \frac{1}{8}\frac{\omega(G)-1}{n}\varepsilon$ for sufficiently large n, we have

$$\frac{\omega(G)-1}{|V|} - 2\frac{\lambda}{d} \geqslant \frac{\omega(G)-1}{|V|}\left(1-\frac{\varepsilon}{4}\right) \tag{6.14}$$

$$(1-\varepsilon)\frac{\omega(G)-1}{|V|} + 2\frac{\lambda}{d} < \frac{\omega(G)-1}{|V|}\left(1-\frac{3}{4}\varepsilon\right). \tag{6.15}$$

Such an expander graph X can be constructed in polynomial time in n, e.g., by using an explicit construction of near-Ramanujan graphs [Alo21, MOP21]. By Lemmas 6.4 and 6.5 and Eqs. (6.14) and (6.15), it is PSPACE-hard to approximate MAXMIN CLIQUE RECONFIGURATION within a factor of

$$\frac{|W|\cdot\left(\frac{\omega(G)-1}{n} - 2\frac{\lambda}{d}\right)^{\ell}}{|W|\cdot\left((1-\varepsilon)\frac{\omega(G)-1}{n} + 2\frac{\lambda}{d}\right)^{\ell}} > \nu^{\ell}, \text{ where } \nu := \frac{1-\frac{\varepsilon}{4}}{1-\frac{3}{4}\varepsilon}. \tag{6.16}$$

Suppose $\nu^{\ell} = N^{\delta}$ for some δ; then, δ should be

$$\begin{aligned}
&\nu^{\lceil \log n \rceil} = (n \cdot d^{\lceil \log n \rceil - 1})^{\delta} \\
&\implies \lceil \log n \rceil \cdot \log \nu = \delta \cdot (\log n + (\lceil \log n \rceil - 1)\cdot \log d) \\
&\implies \delta = \frac{\lceil \log n \rceil \cdot \log \nu}{\log n + (\lceil \log n \rceil - 1)\cdot \log d} = \Theta\left(\frac{\log \nu}{1 + \log d}\right).
\end{aligned} \tag{6.17}$$

Consequently, MAXMIN CLIQUE RECONFIGURATION is PSPACE-hard to approximate within a factor of N^{δ} for some $\delta \in (0,1)$. □

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009 (↰ pp. 12, 24).

[ABM04] Dimitris Achlioptas, Paul Beame, and Michael Molloy. "Exponential Bounds for DPLL Below the Satisfiability Threshold". In: *SODA*. 2004, pp. 139–140 (↰ p. 4).

[AFWZ95] Noga Alon, Uriel Feige, Avi Wigderson, and David Zuckerman. "Derandomized graph products". In: *Comput. Complex.* 5 (1995), pp. 60–75 (↰ pp. 24, 25).

[ALMSS98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. "Proof Verification and the Hardness of Approximation Problems". In: *J. ACM* 45.3 (1998), pp. 501–555 (↰ pp. 4, 5).

[Alo21] Noga Alon. "Explicit Expanders of Every Degree and Size". In: *Comb.* 41.4 (2021), pp. 447–463 (↰ p. 27).

[AS98] Sanjeev Arora and Shmuel Safra. "Probabilistic Checking of Proofs: A New Characterization of NP". In: *J. ACM* 45.1 (1998), pp. 70–122 (↰ pp. 4, 5).

[BC09] Paul Bonsma and Luis Cereceda. "Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances". In: *Theor. Comput. Sci.* 410.50 (2009), pp. 5215–5226 (↰ p. 4).

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. "Non-Deterministic Exponential Time has Two-Prover Interactive Protocols". In: *Computational Complexity* 1 (1991), pp. 3–40. DOI: 10.1007/BF01200056 (↰ p. 6).

[BGHSV06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. "Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding". In: *SIAM J. Comput.* 36.4 (2006), pp. 889–974 (↰ pp. 7, 10, 11).

[BHIKMMSW20] 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*. 2020, 35:1–35:14 (↰ p. 9).

[BHPT20] Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. "Rigid Matrices From Rectangular PCPs or: Hard Claims Have Complex Proofs". In: *FOCS*. 2020, pp. 858–869 (↰ p. 23).

[Bon13] Paul Bonsma. "The Complexity of Rerouting Shortest Paths". In: *Theor. Comput. Sci.* 510 (2013), pp. 1–12 (↰ p. 4).

[BSVW03] Eli Ben-Sasson, Madhu Sudan, Salil Vadhan, and Avi Wigderson. "Randomness-efficient low degree tests and short PCPs via epsilon-biased sets". In: *STOC*. 2003, pp. 612–621 (↰ p. 10).

[CFLS95] Anne Condon, Joan Feigenbaum, Carsten Lund, and Peter W. Shor. "Probabilistically Checkable Debate Systems and Nonapproximability of PSPACE-Hard Functions". In: *Chic. J. Theor. Comput. Sci.* 1995 (1995) (↰ p. 9).

[CLNV15] Siu Man Chan, Massimo Lauria, Jakob Nordströmm, and Marc Vinyals. "Hardness of Approximation in PSPACE and Separation Results for Pebble Games". In: *FOCS*. 2015, pp. 466–485 (↰ p. 9).

[CvJ11] Luis Cereceda, Jan van den Heuvel, and Matthew Johnson. "Finding paths between 3-colorings". In: *J. Graph Theory* 67.1 (2011), pp. 69–82 (↰ p. 4).

[Din07] Irit Dinur. "The PCP Theorem by Gap Amplification". In: *J. ACM* 54.3 (2007), p. 12 (↰ p. 7).

[DL17] Erik D. Demaine and Quanquan C. Liu. "Inapproximability of the Standard Pebble Game and Hard to Pebble Graphs". In: *WADS*. 2017, pp. 313–324 (↰ p. 9).

[DR06] Irit Dinur and Omer Reingold. "Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem". In: *SIAM J. Comput.* 36.4 (2006), pp. 975–1024 (↰ pp. 7, 11).

[DS14] Irit Dinur and David Steurer. "Analytical approach to parallel repetition". In: *STOC*. 2014, pp. 624–633 (↰ p. 4).

[Fei98] Uriel Feige. "A Threshold of $\ln n$ for Approximating Set Cover". In: *J. ACM* 45.4 (1998), pp. 634–652 (↰ p. 4).

[FMV11] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. "Maximizing non-monotone submodular functions". In: *SIAM J. Comput.* 40.4 (2011), pp. 1133–1153 (↰ p. 9).

[GJKL22] Kshitij Gajjar, Agastya Vibhuti Jha, Manish Kumar, and Abhiruk Lahiri. "Reconfiguring Shortest Paths in Graphs". In: *AAAI*. 2022, pp. 9758–9766 (↰ p. 9).

[GKMP09] Parikshit Gopalan, Phokion G. Kolaitis, Elitza Maneva, and Christos H. Papadimitriou. "The Connectivity of Boolean Satisfiability: Computational and Structural Dichotomies". In: *SIAM J. Comput.* 38.6 (2009), pp. 2330–2355 (↰ pp. 3, 4).

[GS06] Oded Goldreich and Madhu Sudan. "Locally testable codes and PCPs of almost-linear length". In: *J. ACM* 53.4 (2006), pp. 558–655 (↰ p. 10).

[GW83] Hana Galperin and Avi Wigderson. "Succinct Representations of Graphs". In: *Inf. Control.* 56.3 (1983), pp. 183–198 (↰ p. 13).

[HD05] Robert A. Hearn and Erik D. Demaine. "PSPACE-Completeness of Sliding-Block Puzzles and Other Problems through the Nondeterministic Constraint Logic Model of Computation". In: *Theor. Comput. Sci.* 343.1-2 (2005), pp. 72–96 (↰ p. 4).

[HD09] Robert A. Hearn and Erik D. Demaine. *Games, Puzzles, and Computation*. A K Peters, Ltd., 2009 (↰ p. 4).

[HLW06] Shlomo Hoory, Nathan Linial, and Avi Wigderson. "Expander graphs and their applications". In: *Bull. Am. Math. Soc.* 43.4 (2006), pp. 439–561 (↰ p. 24).

[Hoa23] Duc A. Hoang. *Combinatorial Reconfiguration*. `https://reconf.wikidot.com/`. 2023 (↰ p. 4).

[HSS84] John E. Hopcroft, Jacob Theodore Schwartz, and Micha Sharir. "On the Complexity of Motion Planning for Multiple Independent Objects; *PSPACE*-Hardness of the "Warehouseman's Problem"". In: *Int. J. Robot. Res.* 3.4 (1984), pp. 76–88 (↰ p. 4).

[ID14] Takehiro Ito and Erik D. Demaine. "Approximability of the subset sum reconfiguration problem". In: *J. Comb. Optim.* 28.3 (2014), pp. 639–654 (↰ p. 4).

[IDHPSUU11] 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". In: *Theor. Comput. Sci.* 412.12-14 (2011), pp. 1054–1065 (↰ pp. 3–5, 9, 23–25).

[IKKKO22] Takehiro Ito, Naonori Kakimura, Naoyuki Kamiyama, Yusuke Kobayashi, and Yoshio Okamoto. "Shortest Reconfiguration of Perfect Matchings via Alternating Cycles". In: *SIAM J. Discret. Math.* 36.2 (2022), pp. 1102–1123 (↰ p. 9).

[JS79] Wm Woolsey Johnson and William Edward Story. "Notes on the "15" puzzle". In: *Am. J. Math.* 2.4 (1879), pp. 397–404 (↰ p. 4).

[KMM11] Marcin Kamiński, Paul Medvedev, and Martin Milanič. "Shortest Paths Between Shortest Paths". In: *Theor. Comput. Sci.* 412.39 (2011), pp. 5205–5210 (↰ p. 9).

[LY94] Carsten Lund and Mihalis Yannakakis. "On the Hardness of Approximating Minimization Problems". In: *J. ACM* 41.5 (1994), pp. 960–981 (↰ p. 4).

[MNORTU16] Tillmann Miltzow, Lothar Narins, Yoshio Okamoto, Günter Rote, Antonis Thomas, and Takeaki Uno. "Approximation and Hardness of Token Swapping". In: *ESA*. 2016, 66:1–66:15 (↰ p. 9).

[MOP21] Sidhanth Mohanty, Ryan O'Donnell, and Pedro Paredes. "Explicit Near-Ramanujan Graphs of Every Degree". In: *SIAM J. Comput.* 51.3 (2021), STOC20-1-STOC20–23 (↰ p. 27).

[Nis18] Naomi Nishimura. "Introduction to Reconfiguration". In: *Algorithms* 11.4 (2018), p. 52 (↰ pp. 4, 5).

[Nor13] Jakob Nordström. "Pebble Games, Proof Complexity, and Time-Space Trade-offs". In: *Log. Methods Comput. Sci.* 9 (2013) (↰ p. 9).

[Ohs23a] Naoto Ohsaka. "Gap Amplification for Reconfiguration Problems". In: *CoRR* abs/2310.14160 (2023) (↰ pp. 6, 7, 24).

[Ohs23b] Naoto Ohsaka. "Gap Preserving Reductions Between Reconfiguration Problems". In: *STACS*. 2023, 49:1–49:18 (↰ pp. 5–7, 9, 18, 23–25).

[OM22] Naoto Ohsaka and Tatsuya Matsuoka. “Reconfiguration Problems on Submodular Functions”. In: *WSDM*. 2022, pp. 764–774 (↰ pp. 4, 9).

[Par21] Orr Paradise. “Smooth and Strong PCPs”. In: *Comput. Complex.* 30.1 (2021), p. 1 (↰ p. 11).

[PH70] Michael S. Paterson and Carl E. Hewitt. “Comparative Schematology”. In: *Record of the Project MAC Conference on Concurrent Systems and Parallel Computation*. 1970, pp. 119–127 (↰ p. 9).

[PY86] Christos H. Papadimitriou and Mihalis Yannakakis. “A Note on Succinct Representations of Graphs”. In: *Inf. Control.* 71.3 (1986), pp. 181–185 (↰ p. 13).

[PY91] Christos H. Papadimitriou and Mihalis Yannakakis. “Optimization, Approximation, and Complexity Classes”. In: *J. Comput. Syst. Sci.* 43.3 (1991), pp. 425–440 (↰ p. 7).

[Sav70] Walter J. Savitch. “Relationships between nondeterministic and deterministic tape complexities”. In: *J. Comput. Syst. Sci.* 4.2 (1970), pp. 177–192 (↰ pp. 14, 18).

[van13] Jan van den Heuvel. “The Complexity of Change”. In: *Surveys in Combinatorics 2013*. Vol. 409. Cambridge University Press, 2013, pp. 127–160 (↰ p. 4).

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il