Vol.2018-AL-169 No.8
2018/9/3

Reconfiguration of Satisfying Assignments for CSP

Tatsuhiko Hatanaka[1,a)] Takehiro Ito[1,b)] Xiao Zhou[1,c)]

Abstract: Constraint satisfaction problem (CSP) is a well-studied combinatorial problem, in which we are asked to find an assignment of values to given variables so as to satisfy all of given constraints. We study a reconfiguration variant of CSP, in which we are given an instance of CSP and two satisfying assignments, and asked to determine whether one assignment can be transformed into the other by changing a single variable assignment at a time, while always remaining satisfying assignment. This problem generalizes several well-studied reconfiguration problems such as Boolean satisfiability reconfiguration, vertex coloring reconfiguration, homomorphism reconfiguration. In this report, we study the problem from the viewpoints of polynomial-time solvability and parameterized complexity.

Keywords: Combinatorial reconfiguration, constraint satisfaction problem, graph algorithm

1. Introduction

Recently, the framework of *reconfiguration* [23] has been extensively studied in the field of theoretical computer science. This framework models several "dynamic" situations where we wish to find a step-by-step transformation between two feasible solutions of a combinatorial (search) problem such that all intermediate solutions are also feasible and each step respects a fixed reconfiguration rule. This reconfiguration framework has been applied to several well-studied combinatorial problems. (See surveys [25], [30], [34].)

1.1 Our problem

We study a reconfiguration variant of the well-known constraint satisfaction problem (CSP, for short). CSP can be defined as a problem on hypergraphs.

A *hypergraph* G is a pair (V, E), where V is a set of vertices and E is a family of non-empty vertex subsets, called *hyperedges*. A hypergraph G is *r-uniform* if every hyperedge consists of exactly r (≥ 1) vertices. Sometimes, a 2-uniform hypergraph G is simply called a *graph* and each hyperedge of G is called an *edge*. An edge $\{v, w\}$ is sometimes denoted as vw or wv for notational convenience.

Let $G = (V, E)$ be a hypergraph. Let D be a set, called a *domain*; each element of D is called a *value* and we always denote by k the size of a domain. In CSP, each hyperedge $X \in E$ has a set $C(X)$ of mappings from X to D, called a *constraint* (of X). An *arity* of a constraint $C(X)$ of X is exactly $|X|$, and we call $C(X)$ an *r-ary* constraint, where $r = |X|$. Let $f\colon V \to D$ be a mapping. For a hyperedge $X \in E$, we say that f *satisfies* a constraint of X if $f|_X \in C(X)$ holds, where $f|_X$ is the restriction of f on X. f is a *solution* if it satisfies all constraints. An instance of CONSTRAINT SATISFIABILITY is a triple (G, D, C) consisting of a hypergraph G, a domain D, and a constraint assignment to hyperedges over D. Then, the problem asks whether there exists a solution or not. CONSTRAINT SATISFIABILITY includes many combinatorial problems as its special cases, such as BOOLEAN CONSTRAINT SATISFIABILITY, r-ARY CONSTRAINT SATISFIABILITY, (LIST) HOMOMORPHISM, and (LIST) COLORING.

We then define a reconfiguration variant of CONSTRAINT SATISFIABILITY, that is, CONSTRAINT SATISFIABILITY RECONFIGURATION.

Let f and f' be two solutions for $\mathcal{I} = (G, D, C)$. We define the *difference* $\mathsf{dif}(f, f')$ between f and f' as the set $\{v \in V(G)\colon f(v) \neq f'(v)\}$. We now define the *solution graph* $\mathscr{S}(\mathcal{I})$ for $\mathcal{I}$ as follows. $V(\mathscr{S}(\mathcal{I}))$ is the set of all solutions for $\mathcal{I}$, and two solutions f and f' are connected by an edge if and only if $|\mathsf{dif}(f, f')| = 1$. A walk in $\mathscr{S}(\mathcal{I})$ is called a *reconfiguration sequence*. Two solutions f and f' are *reconfigurable* if and only if there exists a reconfiguration sequence between them.

An instance of CONSTRAINT SATISFIABILITY RECONFIGURATION (CSR for short) is a 5-tuple (G, D, C, f_s, f_t), where (G, D, C) is an instance of CONSTRAINT SATISFIABILITY, and f_s and f_t are two solutions to (G, D, C), called *initial* and *target* solutions. Then, the problem asks whether f_s and f_t are reconfigurable or not. Similarly, we define, for each special case X of CONSTRAINT SATISFIABILITY, X RECONFIGURATION as a special case of CSR where (G, D, C) is an instance of X. We use the following abbreviations:

- BCSR for BOOLEAN CONSTRAINT SATISFIABILITY RECONFIGURATION [18];
- r-CSR for r-ARY CONSTRAINT SATISFIABILITY RECONFIGURATION for each integer r [18];
- (L)HR for (LIST) HOMOMORPHISM RECONFIGURATION [35]; and
- (L)CR for (LIST) COLORING RECONFIGURATION [4].

Relationships between problems are illustrated in Fig. 1(a).

[1] Graduate School of Information Sciences, Tohoku University, Aobayama 6-6-05, Sendai 980-8579, Japan.
[a)] hatanaka@ecei.tohoku.ac.jp
[b)] takehiro@ecei.tohoku.ac.jp
[c)] zhou@ecei.tohoku.ac.jp

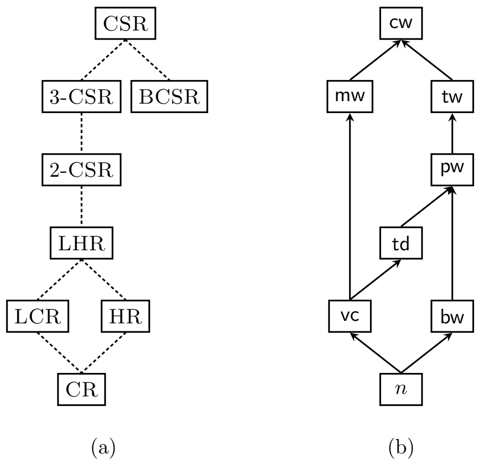

Fig. 1 (a) Relationships between problems. Each dotted line between X (lower) and Y (upper) means that X is a special case of Y. (b) Relationships between graph parameters. cw, mw, tw, pw, td, vc, bw and n are the cliquewidth, the modular-width, the treewidth, the pathwidth, the tree-depth, the size of a minimum vertex cover, the bandwidth and the number of vertices of a graph, respectively. Each arrow $\alpha \rightarrow \beta$ means that α is stronger than β, that is, if α is bounded by a constant then β is also bounded by some constant.

1.2 Known and related results

There are many literatures which study special cases of CSR and their *shortest variants*. In the shortest variant, we are given an instance with an integer ℓ, and asked whether there exists a reconfiguration sequence of length at most $\ell \geq 0$. We here state only the results from the viewpoint of the computational complexity.

One of the most well-studied special cases of CSR is BCSR [5], [12], [18], [27], [28], [29], [33]. Gopalan et al. [18] gave a computational dichotomy for BCSR with respect to a set $\mathcal{S}$ of logical relations which can be used to define each constraint; the problem is PSPACE-complete or in P for each fixed $\mathcal{S}$. It is also known that the problem is PSPACE-complete even if $\mathcal{S}$ is equivalent to monotone Not-All-Equal 3-SAT (i.e., each constraint is a set of surjections) and a "variable-clause incidence graph" is planar [12]. For the shortest variant, a computational trichotomy is known; the problem is PSPACE-complete, NP-complete or in P for each fixed $\mathcal{S}$ [29]. Bonsma et al. [5] proved that the shortest variant is $W[1]$-hard when parameterized by ℓ even if $\mathcal{S}$ is equivalent to Horn SAT.

Another well-studied spacial case is CR [1], [2], [4], [5], [6], [7], [13], [14], [15], [16], [17], [19], [20], [21], [26], [31], [32], [36]. A dichotomy with respect to k is known for CR; it is PSPACE-complete for $k \geq 4$ and bipartite planar graphs [4] but in P for $k \leq 3$ [15]. We note that the second tractability result can be extended for LCR. Moreover, it is known that the problem remains PSPACE-complete even if k is a fixed constant for several graph classes such as line graphs (for any fixed $k \geq 5$) [32], bounded bandwidth graphs [36], and chordal graphs [21]. On the other hand, several polynomial-time algorithms are known for subclasses of chordal graphs such as k'-trees, trivially perfect graphs, split graphs [21], and $(k-2)$-connected chordal graphs [6]. For the shortest variant parameterized by ℓ, some intractability results are known; it is $W[1]$-hard [5] and does not admit a polynomial kernelization when k is fixed unless the polynomial hierarchy collapses [26].

Table 1 Computational complexities with respect to the size k of a domain.

	$k \geq 4$	$k = 3$	$k = 2$
CSR	PSPACE-c.	PSPACE-c.	PSPACE-c.
3-CSR	PSPACE-c.	PSPACE-c.	PSPACE-c. [18]
2-CSR	PSPACE-c.	PSPACE-c. [Thm. 1]	P [Thm. 3]
LHR	PSPACE-c.	P [Thm. 2]	P
LCR	PSPACE-c.	P [15]	P
HR	PSPACE-c.	P [35]	P
CR	PSPACE-c. [4]	P	P

Table 2 Computational complexities for graphs with pathwidth at most two. The result marked with * is ours but omitted from this report.

	pw = 2	pw = 1
CSR	PSPACE-c.	PSPACE-c.
3-CSR	PSPACE-c.	PSPACE-c.
2-CSR	PSPACE-c.	PSPACE-c.
LHR	PSPACE-c.	PSPACE-c. [*]
LCR	PSPACE-c. [20], [36]	P [20]
HR	PSPACE-c. [36]	P [36]
CR	P [21]	P

As a generalization of CR, LCR is studied well [20], [22], [24], [32], [36]. The problem is PSPACE-complete even if k is a constant for graphs with pathwidth two [36], while it is polynomial-time solvable for graphs with pathwidth one [20]. Osawa et al. [32] showed the PSPACE-completeness for line graphs and any fixed $k \geq 4$. Hatanaka et al. [22] gave fixed-parameter algorithms for LCR parameterized by k + mw and for the shortest variant parameterized by k+vc. In contrast, they also showed that the problem is $W[1]$-hard when parameterized only by vc.

HR is also well-studied as a generalization of CR. Several literatures investigated HR from the viewpoint of graph classes [8], [9], [10], [11], [35], [36].

Finally, we refer to the shortest variant of general CSR. Bonsma et al. [5] gave a fixed-parameter algorithm for the shortest variant parameterized by $k + r + \ell$, where r is the maximum arity of a constraint. This implies that shortest variants of BCSR and 2-CSR are fixed-parameter tractable when parameterized by $r + \ell$ and $k + \ell$, respectively. They also showed that the problem is intractable if at least one of $\{k, r, \ell\}$ is excluded from the parameter.

1.3 Our contribution

In this report, we investigate the complexity of CSR and its spacial cases, especially 3-CSR, 2-CSR, (L)HR and (L)CR, from several viewpoints.

1.3.1 The size of a domain

We first classify the complexity of the problems for each fixed size k of a domain in Section 3. The known and our results are summarized in Table 1.

1.3.2 Graph parameters

Since an instance of 2-CSR includes a graph (2-uniform hypergraph), several graph parameters are naturally defined for such an instance. We extend the notion of graph parameters to CSR by taking a "primal graph." The *primal graph* $\mathcal{P}(G)$ of a hypergraph G is a graph such that $V(\mathcal{P}(G)) = V(G)$ and two distinct vertices are connected by an edge if they are contained in the same hyperedge of G. Then, we define any graph parameter of a hypergraph

Table 3 Parameterized complexities with respect to k plus graph parameters. The result marked with * is ours but omitted from this report.

Parameter	$k + \mathsf{mw}$	$k + \mathsf{td}$	$k + \mathsf{vc}$	$k + \mathsf{bw}$
CSR	PSPACE-c.	FPT [Thm. 5]	FPT [Thms. 6, 7]	PSPACE-c.
3-CSR	PSPACE-c.	FPT	FPT	PSPACE-c.
2-CSR	PSPACE-c. [*]	FPT	FPT	PSPACE-c.
LHR	FPT [Thm. 4]	FPT	FPT	PSPACE-c.
LCR	FPT [22]	FPT	FPT	PSPACE-c.
HR	FPT	FPT [36]	FPT	PSPACE-c.
CR	FPT	FPT	FPT	PSPACE-c. [36]

G as the parameter of its primal graph $\mathcal{P}(G)$.[*1] Then we can draw Tables 2 and 3 from this viewpoint. The relationships between graph parameters are summarized in Fig. 1(b); tractability (resp., intractability) result propagates downward (resp., upward).

We omit several proofs and theorems from this report.

2. Preliminally

2.1 Hypergraphs and mappings

Let G be a hypergraph, and let $v \in V(G)$ be a vertex. We denote by $N(G, v)$ the set $\{w \in X \setminus \{v\} : v \in X \in E(G)\}$ of vertices which are *adjacent* to v. For a vertex subset $V' \subseteq V(G)$, we denote $N(G, V') := \bigcup_{v \in V'} N(G, v) \setminus V'$.

Two hypergraphs G and G' are *isomorphic* if there exist two bijections $\phi\colon V(G) \to V(G')$ and $\pi\colon E(G) \to E(G')$ such that $\pi(X) = \{\phi(v_1), \phi(v_2), \ldots, \phi(v_r)\} \in E(G')$ holds for each hyperedge $X = \{v_1, v_2, \ldots, v_r\} \in E(G)$. (See Fig. 2.) For a hypergraph G and a vertex subset $V' \subseteq V(G)$, we define the *subhypergraph* of *G induced by V'* as the hypergraph G' such that $V(G') = V'$ and $E(G') = \{X \cap V' : X \in E(G), X \cap V' \neq \emptyset\}$. We denote by $G[V']$ the subhypergraph of G induced by V' for any vertex subset V'. (See Fig. 3.) We use the notation $G \setminus V'$ to denote $G[V(G) \setminus V']$.

We denote by B^A the set of all mappings from A to B for any sets A and B. Let $\phi \in B^A$ and $\phi' \in B^A$ be two mappings. For any subset A'' of A, we denote by $\phi|_{A''}$ the restriction of ϕ on A''; that is, $\phi|_{A''}$ is a mapping from A'' to B such that $\phi|_{A''}(a) = \phi(a)$ for any $a \in A''$. We say that ϕ and ϕ' are *compatible* if $\phi|_{A \cap A''} = \phi'|_{A \cap A''}$ holds.

2.2 Constraint satisfiability

We first formally define several special cases of CONSTRAINT SATISFIABILITY. BOOLEAN CONSTRAINT SATISFIABILITY is a special case of CONSTRAINT SATISFIABILITY where a domain has size two. For an integer r, r-ARY CONSTRAINT SATISFIABILITY is a special case of CONSTRAINT SATISFIABILITY where all constraints are of arity at most r, that is, all hyperedges have size at most r.

Let (G, D, C) be an instance of CONSTRAINT SATISFIABILITY. We define the *constraint* $C(G)$ of G as the union of all constraints, that is, $C(G) = \bigcup_{X \in E(G)} C(X)$. For a mapping $g \in C(G)$, the *range* $\mathsf{Ran}(g)$ of g is $X \in E(G)$, where $g \in C(X) \subseteq D^X$. For a vertex $v \in V(G)$, a *list* $\mathcal{L}(v)$ of v is the set $\{i \in D : \exists g \in C(G), g(v) = i\}$; notice this is consistent with the notion of lists introduced in the definition of LIST HOMOMORPISM. A *Boolean vertex* is a vertex $v \in V(G)$ with $|\mathcal{L}(v)| \leq 2$, and a *non-Boolean vertex* is a vertex $v \in V(G)$ with $|\mathcal{L}(v)| > 2$. Let X and X' be hyperedges in $E(G)$ such that $|X| = |X'|$. We say that $C(X)$ is *trivial* if $C(X) = D^X$. For a bijection $\phi\colon X \to X'$, we denote by $C[\phi](X)$ the set $\{g \circ \phi^{-1} : g \in C(X)\} \subseteq D^{X'}$ of mappings from X' to D, where $\circ$ means the composition of mappings. Intuitively, $C[\phi](X)$ is a "translation" of $C(X)$ into a constraint of X' via a bijection ϕ. For example, assume that $C(\{v_1, v_2, v_3\})$ contains a mapping g such that $(g(v_1), g(v_2), g(v_3)) = (1, 3, 4)$. If a bijection $\phi\colon \{v_1, v_2, v_3\} \to \{u_1, u_2, u_3\}$ maps v_1, v_2, v_3 to u_2, u_1, u_3, respectively, then $C[\phi](\{v_1, v_2, v_3\})$ contains a mapping g' such that $(g'(v_1), g'(v_2), g'(v_3)) = (g \circ \phi^{-1}(u_1), g \circ \phi^{-1}(u_2), g \circ \phi^{-1}(u_3)) = (g(v_2), g(v_1), g(v_3)) = (3, 1, 4)$.

Let (G, D, C) be an instance of 2-CONSTRAINT SATISFIABILITY. Without loss of generality, we assume that G is connected, $|V(G)| \geq 2$, and $D = \{0, 1\}$. Moreover, we can assume that G is 2-uniform as follows. If G contains a size-one hyperedge $\{v\}$, there must exist a size-two hyperedge (i.e., an edge) $vw \in E(G)$ from the assumption. Then, we remove $\{v\}$ from $E(G)$ and replace $C(vw)$ with the set of all solutions satisfying $C(\{v\})$ and $C(vw)$; this modification does not change the set of solutions. For any $vw \in E(G)$, we sometimes identify a mapping $g\colon \{v, w\} \to D$ with a vector $(g(v), g(w)) \in D^2$. Therefore, a constraint $C(vw)$ can be considered as a subset of D^2.

3. Computational complexity with respect to k

In this section, we classify the complexity of the problems for each fixed size k of a domain.

Theorem 1 2-CSR is PSPACE-complete for bipartite planar graphs even if $k = 3$.

In contrast to Theorem 1, there exist polynomial-time algorithms for more restricted cases. We first show that the problem becomes tractable when restricted to LHR and $k = 3$.

Theorem 2 LHR can be solved in polynomial time if $k = 3$.

We next show that 2-CSR becomes tractable if k is reduced from three to two.

Theorem 3 2-CSR can be solved in polynomial time if $k = 2$.

Proof. We reduce the problem to BIJUNCTIVE BCSR, which is solvable in polynomial time [18]. BIJUNCTIVE BCSR is a special case of BCSR where $D = \{0, 1\}$ and there exists a 2-CNF formula $\phi(v_1, \ldots, v_r)$ such that $C(\{v_1 \ldots, v_r\})$ is exactly the set of all satisfying assignments of ϕ for every hyperedge $\{v_1 \ldots, v_r\} \in E(G)$. Let $\mathcal{I} = (G, D, C, f_s, f_t)$ be a given instance of 2-CSR where G is a graph and $D = \{0, 1\}$. We now show that for every edge $vw \in E(G)$ there exists a 2-CNF formula $\phi(vw)$ such that $C(wv)$ is exactly the set of all satisfying assignments of ϕ. For each $i \in D$

*1 For example, when we refer to the treewidth of a hypergraph G, it means the treewidth of its primal graph $\mathcal{P}(G)$. Note that $\mathcal{P}(G) = G$ if G is 2-uniform.

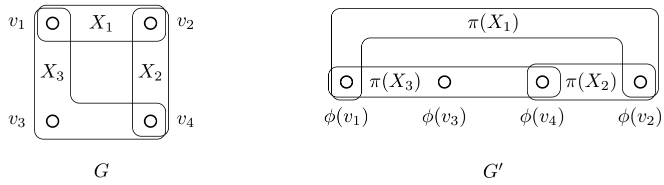

Fig. 2 Two isomorphic hypergraphs G and G' under the bijections ϕ and π.

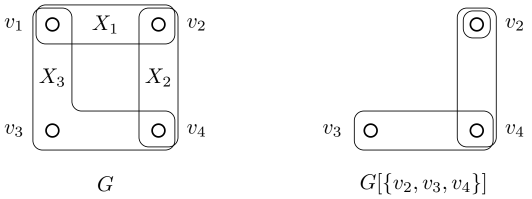

Fig. 3 A graph G and the subhypergraph $G[\{v_2, v_3, v_4\}]$ induced by $\{v_2, v_3, v_4\}$.

and $u \in \{v, w\}$, we denote by u^i a literal u if $i = 0$ or $\bar{u}$ if $i = 1$. Then we define a 2-CNF formula $\phi(v, w)$ as follows:

$$\phi(v, w) = \bigwedge_{(a,b) \in D^2 \setminus C(vw)} (v^a \vee w^b).$$

Notice that a clause $(v^a \vee w^b)$ corresponds to a set $D^2 \setminus \{(a, b)\}$. Therefore, $\phi(v, w)$ corresponds to the set

$$\bigcap_{(a,b) \in D^2 \setminus C(vw)} D^2 \setminus \{(a, b)\} = D^2 \setminus (D^2 \setminus C(vw)) = C(vw)$$

as required. □

4. Fixed-parameter algorithm with respect to graph parameters

We give the following theorems in this section.

Theorem 4 LHR is fixed-parameter tractable when parameterized by $k + \mathsf{mw}$.

Theorem 5 CSR is fixed-parameter tractable when parameterized by $k + \mathsf{td}$.

In this report, we only give an idea of our kernelization algorithm, which compresses an input hypergraph into a smaller hypergraph with keeping the reconfigurability. We note that this is the extension of the lemma given in [22] to obtain a fixed-parameter algorithm for LCR parameterized by $k+\mathsf{mw}$. The main idea is to "identify" two subgraphs which behave in the same way with respect to the reconfigurability.

We now formally characterize such subhypergraphs and explain how to identify them. Let $\mathcal{I} = (G, D, C, f_s, f_t)$ be an instance of CSR. For each vertex $v \in V(G)$, we define $\mathcal{A}(v)$ as a pair $(f_s(v), f_t(v))$ consisting of the initial and the target value assignments of v. Let V_1 and V_2 be two non-empty vertex subsets of G such that $|V_1| = |V_2|$, and $V_1 \cap V_2 = \emptyset$. Assume that $N(G, V_1) = N(G, V_2) = W$. Let $H_1 = G[V_1]$, $H_2 = G[V_2]$, $H'_1 = G[V_1 \cup W]$ and $H'_2 = G[V_2 \cup W]$.

Definition 1 Two induced subhypergraphs H_1 and H_2 are *identical* if there exist two bijections $\phi\colon V(H'_1) \to V(H'_2)$ and $\pi\colon E(H'_1) \to E(H'_2)$ which satisfy the following four conditions:

(1) H'_1 and H'_2 are isomorphic under ϕ and π.

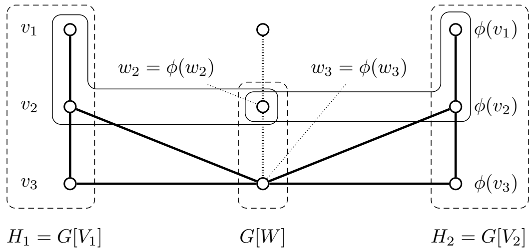

Fig. 4 An example of two subhypergraphs H_1 and H_2 of G which satisfies the conditions (1) and (2). We draw each hyperedge of size two as a solid line, and omit the bijection $\pi\colon E(H'_1) \to E(H'_2)$ since it is uniquely defined from $\phi\colon V(H'_1) \to V(H'_2)$. If $\mathcal{A}$ and C satisfy the conditions (3) and (4), H_1 and H_2 are identical.

(2) for every vertex $v \in W$, $\phi(v) = v$;

(3) for every vertex $v \in V_1$, $\mathcal{A}(v) = \mathcal{A}(\phi(v))$, that is, $f_s(v) = f_s(\phi(v))$ and $f_t(v) = f_t(\phi(v))$; and

(4) for every hyperedge $X \in E(H_1)$, $C(\pi(X)) = C[\widehat{\phi}](X)$, where $\widehat{\phi} = \phi|_X$.

See Fig. 4 for an example.

We next define another instance $\mathcal{I}' = (G', D, C', f'_s, f'_t)$ as follows:

- $G' = G \setminus V_2$;
- $f'_s = f_s|_{V(G')}$ and $f'_s = f_s|_{V(G')}$; and
- for each $X \in E(G')$, $C'(X) = \{g|_X : g \in C(G), \mathsf{Ran}(g) \setminus V_2 = X\}$.

Intuitively, $\mathcal{I}'$ is obtained by restricting all components (hypergraphs, mappings in constraints, and two solutions) of $\mathcal{I}$ on $V(G) \setminus V_2$. We say that $\mathcal{I}'$ is obtained from $\mathcal{I}$ by *identifying* H_1 with H_2.

Then, we have the following lemma, which says that $\mathcal{I}$ and $\mathcal{I}'$ are equivalent with respect to the feasibility.

Lemma 1 Let $f'\colon V(G') \to D$ be a mapping from $V(G')$ to D. Then, f' is a solution for (G', D, C') if and only if there exists a solution f for (G, D, C) such that $f' = f|_{V(G')}$.

We now give the following key lemma, which says that $\mathcal{I}$ and $\mathcal{I}'$ are equivalent with respect to even the reconfigurability.

Lemma 2 (Reduction rule) Let $\mathcal{I}$ and $\mathcal{I}'$ be instances of CSR defined as above. Then, $\mathcal{I}'$ is a yes-instance if and only if $\mathcal{I}$ is.

5. Vertex cover

In this section, we consider the size vc of a minimum vertex cover. Note that Theorem 5 implies CSR is fixed-parameter tractable when parameterized by $k + \mathsf{vc}$. We strengthen it as follows.

Theorem 6 The shortest variant of CSR is fixed-parameter tractable when parameterized by $k + \mathsf{vc}$.

Theorem 7 There exists a fixed-parameter algorithm for CSR parameterized by $k + \mathsf{vc}$ which runs in time $O^*(k^{\mathsf{vc}})$.

5.1 Proof of Theorem 7

In order to prove the theorem, we first introduce the notion of a "contracted solution graph", which was first introduced in [3] and used in several literatures such as [6], [20].

Let $\mathcal{I} = (\mathcal{J}, f_s, f_t)$ be an instance of CSR, where $\mathcal{J} = (G, D, C)$, and let $\mathscr{P}$ be a partition of the vertex set of the solution graph $\mathscr{S}(\mathcal{J})$. The *contracted solution graph* (or *CSG* for short) $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$ is defined as follows. The vertex set $V(\mathsf{CSG}(\mathcal{J}, \mathscr{P}))$ is exactly $\mathscr{P}$; we call each vertex of the CSG a *node*. Each pair of distinct nodes (i.e., sets of solutions) $P, P' \in \mathscr{P}$ are adjacent in the CSG if and only if there exist two solutions $f \in P$ and $f' \in P'$ such that $ff' \in E(\mathscr{S}(\mathcal{J}))$. In other words, $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$ is obtained by contracting a (possibly disconnected) subgraph of $\mathscr{S}(\mathcal{J})$ induced by each set $P \in \mathscr{P}$ into one node. A partition $\mathscr{P}$ is *proper* if every set $P \in \mathscr{P}$ induces a connected subgraph of $\mathscr{S}(\mathcal{J})$. Since the contraction of a connected subgraph maintains the connectivity of a graph, we have the following proposition.

Proposition 1 Let $\mathcal{I} = (\mathcal{J}, f_s, f_t)$ be an instance of CSR, where $\mathcal{J} = (G, D, C)$, and let $\mathscr{P}$ be a proper partition of $V(\mathscr{S}(\mathcal{J}))$. Then, $\mathcal{I}$ is a yes-instance if and only if there exists a walk between P_s and P_t in $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$, where $f_s \in P_s$ and $f_t \in P_t$. Moreover, the above condition can be checked in time polynomial in $|\mathscr{P}|$.

Therefore, we first define a proper partition $\mathscr{P}$ such that $|\mathscr{P}|$ depends only on $k + \mathsf{vc}$, and then give an algorithm constructing the CSG and specifying the nodes corresponding to f_s and f_t.

5.1.1 Defining a proper partition

Let $\mathcal{I} = (\mathcal{J}, f_s, f_t)$ be an instance of CSR, where $\mathcal{J} = (G, D, C)$. Assume that $\mathcal{P}(G)$ has a vertex cover C of size at most vc. For each solution $f \in V(\mathscr{S}(\mathcal{J}))$, we define $[f] = \{f' : f|_C = f'|_C\}$. Then, we define $\mathscr{P} = \{[f] : f \in V(\mathscr{S}(\mathcal{J}))\}$; that is, $\mathscr{P}$ is the set of the equivalence classes under the equivalence relation "their restrictions on C are the same". Clearly, $\mathscr{P}$ is a partition of $V(\mathscr{S}(\mathcal{J}))$ and $|\mathscr{P}|$ is bounded by the number of mappings from C to D, that is, $|\mathscr{P}| \leq k^{\mathsf{vc}}$.

In order to prove that $\mathscr{P}$ is proper, we introduce some notation. Let $S \subseteq V(G)$ be a vertex subset, and let $h : S \to D$ be a mapping from S to D. We define the *substitution* $\mathsf{SUB}(\mathcal{J}; h)$ as an instance (G', D, C') of CONSTRAINT SATISFIABILITY such that:

- $G' = G \setminus S$; and
- for each $X' \in E(G')$, $C'(X') = \bigcap_{X \in E'} \mathcal{G}(X)$, where $E' = \{X \in E(G) : X \setminus S = X'\}$ and $\mathcal{G}(X) = \{g|_{X'} : g \in C(X),$ h and g are compatible$\}$

We have the following lemma.

Lemma 3 Let $f' : V(G) \setminus S \to D$ and $f : V(G) \to D$ be two mappings such that $f|_{V(G)\setminus S} = f'$. Then, f' is a solution for $\mathsf{SUB}(\mathcal{J}; f|_S) = (G', D, C')$ if and only if f is a solution for (G, D, C).

The following lemma implies that $\mathscr{P}$ is proper.

Lemma 4 Let P be a solution set in $\mathscr{P}$ such that $f|_C = h$ holds for every $f \in P$. Then, $\mathscr{S}(\mathcal{J})[P]$ is connected.

5.1.2 Algorihm computing CSG

In order to give an algorithm computing $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$ correctly, we first show two claims.

Claim 1 Let h be a mapping from C to D. Then, $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$ has a node corresponding to h if and only if $\mathsf{SUB}(\mathcal{J}; h) = (G', D, C')$ has a solution.

Claim 2 Let P_1 and P_2 be two nodes of $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$, and let $h_1 : C \to D$ and $h_2 : C \to D$ be mappings corresponding to P_1 and P_2, respectively. Then, $P_1P_2 \in E(\mathsf{CSG}(\mathcal{J}, \mathscr{P}))$ if and only if both of the following conditions hold:

- $|\mathsf{dif}(h_1, h_2)| = 1$; and
- $\mathsf{SUB}(\mathcal{J}; h_1)$ and $\mathsf{SUB}(\mathcal{J}; h_2)$ has a common solution f'.

From Claims 1 and 2, we can construct the following algorithm to compute $\mathsf{CSG}(\mathcal{J}, \mathscr{P})$ with nodes corresponding to f_s and f_t.

Phase 1 For each mapping h from C to D, check if $\mathsf{SUB}(\mathcal{J}; h)$ has a solution. If so, create a node corresponding to h. For each $r \in \{s, t\}$, if $h = f_r|_C$, it corresponds to f_r.

Phase 2 For each pair of two nodes P_1 and P_2, check if the two conditions of Claim 2 hold. If so, join them by an edge.

The correctness follows from Claims 1 and 2. The first phase can be done in polynomial time for each mapping, because the constructed instance $\mathsf{SUB}(\mathcal{J}; h)$ of CONSTRAINT SATISFIABILITY contains only 1-ary constraints. Since $|D^C| \leq k^{\mathsf{vc}}$, whole running time of this phase is $O^*(k^{\mathsf{vc}})$. In the second phase, the second condition of Claim 2 can be checked as follows. Let C_1 and C_2 are constraint assignments in the substitutions $\mathsf{SUB}(\mathcal{J}; h_1)$ and $\mathsf{SUB}(\mathcal{J}; h_2)$. We now define for each $X' \in E(G')$ a constraint $C'(X') = C_1(X') \cap C_2(X')$. Then, a solution for (G', D, C') is also a solution for both of $\mathsf{SUB}(\mathcal{J}; h_1)$ and $\mathsf{SUB}(\mathcal{J}; h_2)$. Because (G', D, C') is an instance of CONSTRAINT SATISFIABILITY which contains only 1-ary constraints, we can solve it in polynomial time. Therefore, whole running time of this phase is $O^*(k^{\mathsf{vc}})$.

We thus completed the proof of Theorem 7. □

5.2 Discussions

We conclude this section by discussing hitting sets on hypergraphs, which is a well-known generalization of vertex covers on graphs. Although a hitting set of a 2-uniform hypergraph is equivalent to a vertex cover of the graph, such an equivalence does not hold for general hypergraphs. Thus, it is worth considering the complexity of CSR with respect to the size of a hitting set of a given hypergraph. We have the following theorem, which implies that a fixed-parameter algorithm for CSR is unlikely to exist when parameterized by the size of a hitting set plus k.

Theorem 8 3-CSR is PSPACE-complete even for hypergraphs with a hitting set of size one and $k = O(1)$.

Acknowledgments This work is partially supported by JST CREST Grant Number JPMJCR1402, and by JSPS KAKENHI Grant Numbers JP18H04091, JP16J02175, JP16K00003, and JP16K00004, Japan.

References

[1] Bonamy, M. and Bousquet, N.: Recoloring Graphs via Tree Decompositions, *European Journal of Combinatorics*, Vol. 69, pp. 200–213 (online), DOI: 10.1016/j.ejc.2017.10.010 (2018).

[2] Bonamy, M., Johnson, M., Lignos, I., Patel, V. and Paulusma, D.: Reconfiguration graphs for vertex colourings of chordal and chordal bipartite graphs, *Journal of Combinatorial Optimization*, Vol. 27, No. 1, pp. 132–143 (online), DOI: 10.1007/s10878-012-9490-y (2014).

[3] Bonsma, P.: Rerouting shortest paths in planar graphs, *Discrete Applied Mathematics*, Vol. 231, pp. 95 – 112 (online), DOI: https://doi.org/10.1016/j.dam.2016.05.024 (2017). Algorithmic Graph Theory on the Adriatic Coast.

[4] Bonsma, P. and Cereceda, L.: Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances, *Theoretical Computer Science*, Vol. 410, No. 50, pp. 5215–5226 (online), DOI: 10.1016/j.tcs.2009.08.023 (2009).

[5] Bonsma, P., Mouawad, A. E., Nishimura, N. and Raman, V.: The Complexity of Bounded Length Graph Recoloring and CSP Reconfiguration, *Parameterized and Exact Computation - 9th International Symposium, IPEC 2014, Wroclaw, Poland, September 10-12, 2014. Revised Selected Papers*, pp. 110–121 (online), DOI: 10.1007/978-3-319-13524-3_10 (2014).

[6] Bonsma, P. and Paulusma, D.: Using Contracted Solution Graphs for Solving Reconfiguration Problems, *41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), Vol. 58, pp. 20:1–20:15 (online), DOI: 10.4230/LIPIcs.MFCS.2016.20 (2016).

[7] Bousquet, N. and Perarnau, G.: Fast Recoloring of Sparse Graphs, *European Journal of Combinatorics*, Vol. 52, pp. 1–11 (online), DOI: 10.1016/j.ejc.2015.08.001 (2016).

[8] Brewster, R. C., baek Lee, J. and Siggers, M.: Recolouring reflexive digraphs, *Discrete Mathematics*, Vol. 341, No. 6, pp. 1708 – 1721 (online), DOI: https://doi.org/10.1016/j.disc.2018.03.006 (2018).

[9] Brewster, R. C., Lee, J.-B., Moore, B., Noel, J. A. and Siggers, M.: Graph Homomorphism Reconfiguration and Frozen H-Colourings, *arXiv preprint* (2017).

[10] Brewster, R. C., McGuinness, S., Moore, B. and Noel, J. A.: A dichotomy theorem for circular colouring reconfiguration, *Theoretical Computer Science*, Vol. 639, pp. 1–13 (online), DOI: 10.1016/j.tcs.2016.05.015 (2016).

[11] Brewster, R. C. and Noel, J. A.: Mixing Homomorphisms, Recolorings, and Extending Circular Precolorings, *Journal of Graph Theory*, Vol. 80, No. 3, pp. 173–198 (online), DOI: 10.1002/jgt.21846 (2015).

[12] Cardinal, J., Demaine, E. D., Eppstein, D., Hearn, R. A. and Winslow, A.: Reconfiguration of Satisfying Assignments and Subset Sums: Easy to Find, Hard to Connect, *Computing and Combinatorics* (Wang, L. and Zhu, D., eds.), pp. 365–377 (online), DOI: https://doi.org/10.1007/978-3-319-94776-1_31 (2018).

[13] Celaya, M., Choo, K., MacGillivray, G. and Seyffarth, K.: Reconfiguring k-Colourings of Complete Bipartite Graphs, *Kyungpook Mathematical Journal*, Vol. 56, pp. 647–655 (online), DOI: 10.5666/KMJ.2016.56.3.647 (2016).

[14] Cereceda, L.: Mixing Graph Colourings, PhD Thesis, The London School of Economics and Political Science (2007).

[15] Cereceda, L., van den Heuvel, J. and Johnson, M.: Finding Paths Between 3-colorings, *Journal of Graph Theory*, Vol. 67, No. 1, pp. 69–82 (online), DOI: 10.1002/jgt.20514 (2011).

[16] Dyer, M., Flaxman, A. D., Frieze, A. M. and Vigoda, E.: Randomly coloring sparse random graphs with fewer colors than the maximum degree, *Random Structures & Algorithms*, Vol. 29, No. 4, pp. 450–465 (online), DOI: 10.1002/rsa.20129 (2006).

[17] Feghali, C., Johnson, M. and Paulusma, D.: A Reconfigurations Analogue of Brooks' Theorem and Its Consequences, *Journal of Graph Theory*, Vol. 83, No. 4, pp. 340–358 (online), DOI: 10.1002/jgt.22000 (2016).

[18] Gopalan, P., Kolaitis, P. G., Maneva, E. N. and Papadimitriou, C. H.: The Connectivity of Boolean Satisfiability: Computational and Structural Dichotomies, *SIAM Journal on Computing*, Vol. 38, No. 6, pp. 2330–2355 (online), DOI: 10.1137/07070440X (2009).

[19] Haas, R. and MacGillivray, G.: Connectivity and Hamiltonicity of Canonical Colouring Graphs of Bipartite and Complete Multipartite Graphs, *Algorithms*, Vol. 11, No. 4 (online), DOI: 10.3390/a11040040 (2018).

[20] Hatanaka, T., Ito, T. and Zhou, X.: The List Coloring Reconfiguration Problem for Bounded Pathwidth Graphs, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E98.A, No. 6, pp. 1168–1178 (online), DOI: 10.1587/transfun.E98.A.1168 (2015).

[21] Hatanaka, T., Ito, T. and Zhou, X.: The Coloring Reconfiguration Problem on Specific Graph Classes, *Combinatorial Optimization and Applications - 11th International Conference, COCOA 2017, Shanghai, China, December 16-18, 2017, Proceedings, Part I*, pp. 152–162 (online), DOI: 10.1007/978-3-319-71150-8_15 (2017).

[22] Hatanaka, T., Ito, T. and Zhou, X.: Parameterized complexity of the list coloring reconfiguration problem with graph parameters, *Theoretical Computer Science*, Vol. 739, pp. 65 – 79 (online), DOI: https://doi.org/10.1016/j.tcs.2018.05.005 (2018).

[23] Ito, T., Demaine, E. D., Harvey, N. J., Papadimitriou, C. H., Sideri, M., Uehara, R. and Uno, Y.: On the complexity of reconfiguration problems, *Theoretical Computer Science*, Vol. 412, No. 12, pp. 1054–1065 (online), DOI: 10.1016/j.tcs.2010.12.005 (2011).

[24] Ito, T., Kawamura, K. and Zhou, X.: An Improved Sufficient Condition for Reconfiguration of List Edge-Colorings in a Tree, *IEICE Transactions on Information and Systems*, Vol. 95-D, No. 3, pp. 737–745 (online), DOI: 10.1587/transinf.E95.D.737 (2012).

[25] Ito, T. and Suzuki, A.: Web survey on combinatorial reconfiguration, `http://www.ecei.tohoku.ac.jp/alg/coresurvey/`. Updated on November 9, 2017.

[26] Johnson, M., Kratsch, D., Kratsch, S., Patel, V. and Paulusma, D.: Finding Shortest Paths Between Graph Colourings, *Algorithmica*, Vol. 75, No. 2, pp. 295–321 (online), DOI: 10.1007/s00453-015-0009-7 (2016).

[27] Makino, K., Tamaki, S. and Yamamoto, M.: On the Boolean connectivity problem for Horn relations, *Discrete Applied Mathematics*, Vol. 158, No. 18, pp. 2024–2030 (online), DOI: 10.1016/j.dam.2010.08.019 (2010).

[28] Makino, K., Tamaki, S. and Yamamoto, M.: An exact algorithm for the Boolean connectivity problem for k-CNF, *Theoretical Computer Science*, Vol. 412, No. 35, pp. 4613 – 4618 (online), DOI: https://doi.org/10.1016/j.tcs.2011.04.041 (2011).

[29] Mouawad, A. E., Nishimura, N., Pathak, V. and Raman, V.: Shortest Reconfiguration Paths in the Solution Space of Boolean Formulas, *SIAM Journal on Discrete Mathematics*, Vol. 31, No. 3, pp. 2185–2200 (online), DOI: 10.1137/16M1065288 (2017).

[30] Nishimura, N.: Introduction to Reconfiguration, *Algorithms*, Vol. 11, No. 4 (online), DOI: 10.3390/a11040052 (2018).

[31] Osawa, H., Suzuki, A., Ito, T. and Zhou, X.: Complexity of Coloring Reconfiguration under Recolorability Constraints, *Proceedings of ISAAC 2017*, Leibniz International Proceedings in Informatics (LIPIcs), Vol. 92, pp. 62:1–62:12 (online), DOI: 10.4230/LIPIcs.ISAAC.2017.62 (2017).

[32] Osawa, H., Suzuki, A., Ito, T. and Zhou, X.: The Complexity of (List) Edge-Coloring Reconfiguration Problem, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. 101-A, No. 1, pp. 232–238 (online), DOI: 10.1587/transfun.E101.A.232 (2018).

[33] Schwerdtfeger, K. W.: A Computational Trichotomy for Connectivity of Boolean Satisfiability, *Journal on Satisfiability, Boolean Modeling and Computation*, Vol. 8, No. 3/4, pp. 173–195 (2014).

[34] van den Heuvel, J.: The complexity of change, *Surveys in Combinatorics 2013*, Cambridge University Press, pp. 127–160 (online), DOI: 10.1017/CBO9781139506748.005 (2013).

[35] Wrochna, M.: Homomorphism Reconfiguration via Homotopy, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, Leibniz International Proceedings in Informatics (LIPIcs), pp. 730–742 (online), DOI: 10.4230/LIPIcs.STACS.2015.730 (2015).

[36] Wrochna, M.: Reconfiguration in bounded bandwidth and tree-depth, *Journal of Computer and System Sciences*, Vol. 93, pp. 1–10 (online), DOI: https://doi.org/10.1016/j.jcss.2017.11.003 (2018).