EPiC Series in Computing

Volume 68, 2020, Pages 37–49

ANDREI-60. Automated New-era Deductive Reasoning Event in Iberia

Reconstructing in the Constraint Satisfaction Problem

Evgeny Dantsin

Roosevelt University, Chicago, IL, United States
edantsin@roosevelt.edu

Abstract

It is a long-standing problem in graph theory to prove or disprove the *reconstruction conjecture*, also known as the Kelly-Ulam conjecture. This conjecture states that every simple graph on at least three vertices is *reconstructible*, which means that the isomorphism class of such a graph is uniquely determined by the isomorphism classes of its vertex-deleted subgraphs. In this paper, the notion of reconstructing is extended from graphs to instances of the constraint satisfaction problem (CSP): an instance is *reconstructible* if its isomorphism class is uniquely determined by the isomorphism classes of its constraint-deleted subinstances. Questions of interest include not only questions about reconstructible CSP instances but also about CSP instances with reconstructible properties and parameters such as the existence of solutions or the number of solutions. It is shown in the paper that such questions can be answered using techniques borrowed and adapted from graph reconstruction. In particular, Lovász's method of counting graph homomorphisms [Lov72] is adapted to characterize CSP instances for which the number of solutions is reconstructible.

1 Introduction

What does "reconstructing" mean? In this paper, the word "reconstructing" and its derivatives have the meaning illustrated by the following example. Let s be a finite sequence of n elements and k be an integer less than n. Let $M_k(s)$ be the multiset of all subsequences obtained from s by deleting k elements. For example, if s is the string 0110 of four bits, then

$$\begin{array}{rcl} M_1(s) & = & \{110, 010, 010, 110\}; \\ M_2(s) & = & \{10, 10, 11, 00, 01, 01\}. \end{array}$$

We say that s is *k-reconstructible* if $M_k(s)$ uniquely determines s, i.e. for every sequence s' different from s, we have $M_k(s') \neq M_k(s)$. In our example, it is straightforward to check that 0110 is 1-reconstructible and not 2-reconstructible.

The example with sequences is generalized in two directions. First, we can consider objects that are more general than sequences, such as matrices, graphs, structures, etc. Second, we can consider a set of objects with a more general equivalence relation than identity, for example, a set of graphs with the isomorphism relation on them. In this case, we say that an object s is *k-reconstructible* from the corresponding multiset of its sub-objects $s_1, \ldots, s_m$ if the equivalence class of s is uniquely determined by the equivalence classes of $s_1, \ldots, s_m$.

Given a reconstruction setting like above, it is natural to ask what objects are reconstructible and what are not. Notice that even if an object is not reconstructible, some of its properties and parameters can be still reconstructible. For example, when considering graphs up to isomorphism, the number of edges of a graph is reconstructible even if it is unknown whether the graph itself is reconstructible or not. Therefore, it is also natural to ask what properties and parameters of objects are reconstructible.

The graph reconstruction conjecture and its variations. Perhaps the best known examples of reconstructing are concerned with graphs and isomorphisms between them. Let G be a simple graph on vertices $v_1, \ldots, v_n$. Consider the subgraphs

$$G - v_1, \ldots, G - v_n$$

where $G - v_i$ is obtained from G by removing v_i and its incident edges; these subgraphs are called the *vertex-deleted* subgraphs of G. The *deck* of G is a multiset of n *cards* that represent the vertex-deleted subgraphs: the ith card is the isomorphism class of $G - v_i$. We say that G is *reconstructible* if the deck of G uniquely determines the isomorphism class of G, which means that if a graph G' has the same deck as G then G' is isomorphic to G. What graphs are reconstructible?

In the 1940s, Kelly and Ulam proposed the *graph reconstruction conjecture*, also known as the *Kelly-Ulam conjecture* or *Ulam's conjecture*. The conjecture states that every simple graph on at least three vertices is reconstructible. It is still an open problem whether the conjecture is true or not, but many interesting results were obtained in attempts to prove or disprove it. For example, we know that all graphs in many well known classes are reconstructible, including regular graphs, Eulerian graphs, trees, disconnected graphs [Kel57]. Considering randomly chosen graphs, almost all graphs are reconstructible [Mül76]. Moreover, almost all graphs are reconstructible using only three cards from their decks [Bol90]. Computerized verification shows that all graphs with up to 11 vertices are reconstructible [McK97].

Properties or parameters of a graph are called *isomorphism invariants* if they are preserved under isomorphisms. An isomorphism invariant is called *reconstructible* if it is uniquely determined by the deck of the graph. A simple example of a reconstructible invariant is the number of edges. Indeed, notice that if a graph has n vertices then each edge of the graph appears in exactly $n - 2$ cards. Other examples of reconstructible invariants include connectivity, the degree sequence, the Tutte polynomial, the chromatic number, the number of Hamiltonian cycles, and planarity, see a survey in [LS16].

A number of variations of the graph reconstruction conjecture have been proposed and studied. Some of them have been disproved, for example, for directed graphs [Sto77], for hypergraphs [Koc87], and for infinite graphs with finite degrees [BEH+17]. Other variations have still been neither proved nor disproved, for example, the *set reconstruction conjecture* proposed by Harary in [Har64]: every simple graph with at least four vertices is reconstructible from the set (not the multiset) of its cards.

Edge reconstruction. There is another variant of graph reconstruction proposed by Harary in [Har64]. For a graph G with edges $e_1, \ldots, e_m$, the *edge-deleted subgraphs* of G are the subgraphs

$$G - e_1, \ldots, G - e_m$$

where $G - e_i$ is obtained from G by removing e_i. The *edge-deck* of a graph is the multiset of the isomorphism classes of its edge-deleted subgraphs. A graph G is *edge-reconstructible* if

the edge-deck of G uniquely determines the isomorphism class of G. The *edge-reconstruction conjecture* [Har64] states that every simple graph with at least four edges is edge-reconstructible. It was shown in [Gre71] that the edge-deck of a graph with at least four edges and no isolated vertices uniquely determines the deck of this graph. Therefore, if the Kelly-Ulam conjecture is true, then the edge-reconstruction conjecture is true. The notion of edge reconstruction can be extended to hypergraphs [Ber72]. The edge-deck for a hypergraph and the edge-reconstruction conjecture for hypergraphs are defined in literally the same way as for graphs. It is open for both edge-reconstruction conjectures whether they are true or not.

Lovász showed in [Lov72] that a graph on n vertices with m edges is edge-reconstructible from its deck if m is sufficiently large compared with n, namely if $m > \frac{1}{2}\binom{n}{2}$. The proof uses the inclusion-exclusion principle to count homomorphism between graphs. Müller improved Lovász's bound to $2^{m-1} > n!$ in [Mül77]; this result also follows from the sufficient condition for edge reconstruction given by Nash-Williams in [NW78].

Reconstructing other combinatorial objects. In reconstruction of sequences (described in the beginning of this section), we are interested in whether or not a sequence of length n is determined by its $\binom{n}{k}$ subsequences of length k. The main results on reconstruction of sequences are lower and upper bounds on k [MMS+91, KR97, DS03]. Reconstruction of matrices from submatrices is defined similarly to reconstruction of sequences [KLS09]. Another related type of reconstruction is motivated by DNA sequence assembly: reconstruction of jigsaw puzzles [MR15, NPS17, BBN19].

Reconstruction of the same objects can be defined differently, depending on how we define isomorphism between these objects. For example, reconstruction of Boolean formulas in conjunctive normal form (CNFs) from their clause-deleted subformulas is defined in [DW18] based on the following isomorphism: two CNFs are *isomorphic* if one of them can be obtained from the other by renaming variables and flipping literals. In the current paper, CNFs are viewed as CSP instances, which defines a different type of isomorphism between them, see Section 2 for details. Since the types of isomorphism differ, the corresponding notions of CNF reconstruction are different too.

The notion of reconstruction defined by Alon, Caro, Krasikov, and Roditty in [ACKR89] generalizes edge reconstruction of graphs and hypergraphs, see details in Section 3. They consider a set X and a group Γ acting on X. This action determines an equivalence relation on the subsets of X. A subset $Y \subseteq X$ of cardinality m is called *k-reconstructible* if the equivalence class of Y is uniquely determined by the equivalence classes of all subsets of Y of cardinality $m - k$. Subsets of X represent graphs; equivalent subsets represent isomorphic graphs. A k-reconstructible subset of cardinality m represents a graph G with m edges such that the isomorphism class of G is uniquely determined by the isomorphism classes of the subgraphs obtained from G by removing k edges. The main results of [ACKR89] are sufficient conditions for k-reconstruction. When applying to edge reconstruction of graphs, these conditions improve bounds in [Mül77, NW78] mentioned above.

What is done in this paper. The paper defines the notion of reconstructing for instances of the constraint satisfaction problem (CSP): an instance ϕ with m constraints is *k-reconstructible* if its isomorphism class is uniquely determined by the isomorphism classes of all $\binom{m}{k}$ subinstances obtained from by ϕ by removing k constraints. A CSP instance is *reconstructible* if it is 1-reconstructible.

What CSP instances are k-reconstructible and what are not? Theorem 4 gives a sufficient condition for k-reconstruction of CSP instances; its proof is based on the combinatorial tech-

nique used by Alon, Caro, Krasikov, Roditty in [ACKR89]. Namely, let ϕ be a CSP instance with m constraints over n variables taking values from a domain of cardinality d. Suppose that

$$2^{m-k} > \frac{n! \cdot d!}{|Aut(\phi)|}$$

where $|Aut(\phi)|$ denotes the order of the automorphism group of ϕ. Then ϕ is k-reconstructible.

A property or parameter of CSP instances is called an *invariant* if it is preserved under isomorphism. For example, the existence of solutions and the number of solutions are invariants. The existence of nontrivial automorphisms and the automorphism group order are other examples. Consider a CSP instance ϕ and some invariant, say, the number of solutions. Is this invariant uniquely determined by the k-deck of ϕ? If ϕ is k-reconstructible, then the obvious answer is "yes". Can we answer this question if ϕ is not k-reconstructible or we do not know whether ϕ is k-reconstructible?

Theorem 6 characterizes CSP instances for which the number of solutions is reconstructible. To illustrate how this characterization works, the theorem is applied to the graph coloring problem (`COL`) and to the Boolean satisfiability problem (`SAT`). The application to `COL` gives a simple proof of the known result that the number of graph colorings is reconstructible, which follows from Tutte's theorem in [Tut79]. The application to `SAT` shows that the number of satisfying assignments for a Boolean formula in CNF is reconstructible.

Organization of the paper. Section 2 contains definitions for the basic notions used in the paper: CSP instances and isomorphism between them. Reconstruction of CSP instances is defined in Section 3. This section also gives a sufficient condition for CSP instances to be k-reconstructible. Section 4 contains the theorem about reconstructing the number of solutions and its applications to `COL` and `SAT`. Concluding remarks are made in Section 5.

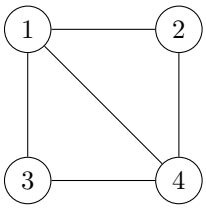
2 Isomorphism Between CSP Instances

In this section, we define the constraint satisfaction problem (CSP), consider its connection to homomorphisms between finite relational structures, and define isomorphism between CSP instances. The definitions are illustrated with two running examples, the problem of coloring a graph with k colors and the satisfiability problem for Boolean formulas in k-CNF, that are also used in the next sections.

2.1 CSP: Definitions and Examples

In the definitions below and throughout the paper, we use the following notation for tuples and relations. Let f be a function from a set A to a set B. For any k-tuple $\mathbf{x} = (x_1, \ldots, x_k)$ in A^k, we write $f(\mathbf{x})$ to denote the k-tuple $(f(x_1), \ldots, f(x_k))$ in B^k. For any k-ary relation $R \subseteq A^k$, we write $f(R)$ to denote the subset $\{f(\mathbf{x}) \mid \mathbf{x} \in R\}$ of B^k.

Definition 1 (constraint satisfaction)**.** Let V be a finite set of *variables*. Let D be a finite set of *values*; this set is called the *domain*. A *constraint* of arity k over V and D is a pair $C = (\mathbf{x}, R)$ where $\mathbf{x}$ is a k-ary tuple of variables and R is a k-ary relation on D. The tuple $\mathbf{x}$ is called the *scope* of C; the relation R is called the *constraint relation* of C. An *assignment* is a function $f : V \to D$. We say that f *satisfies* C if $f(\mathbf{x}) \in R$.



$$V = \{x_1, x_2, x_3, x_4\},\ D = \{b, g, r\},\ \mathcal{C} = \{C_1 \ldots, C_5\}$$

$$C_1 = ((x_1, x_2), \{b \neq g, g \neq b, b \neq r, r \neq b, g \neq r, r \neq g\})$$
$$C_2 = ((x_1, x_3), \{b \neq g, g \neq b, b \neq r, r \neq b, g \neq r, r \neq g\})$$
$$C_3 = ((x_1, x_4), \{b \neq g, g \neq b, b \neq r, r \neq b, g \neq r, r \neq g\})$$
$$C_4 = ((x_2, x_4), \{b \neq g, g \neq b, b \neq r, r \neq b, g \neq r, r \neq g\})$$
$$C_5 = ((x_3, x_4), \{b \neq g, g \neq b, b \neq r, r \neq b, g \neq r, r \neq g\})$$

Figure 1: An instance of COL represented as a CSP instance.

Definition 2 (CSP)**.** The *constraint satisfaction problem* (abbreviated *CSP*) is the following decision problem. A *CSP instance* is a triple $(V, D, \mathcal{C})$ where $\mathcal{C}$ is a finite set of constraints over V and D. A *solution* to this instance is an assignment that satisfies all constraints in $\mathcal{C}$. The question is whether a given instance has a solution.

Many well-known decision problems can be represented as special cases of the CSP. Consider two examples.

Example 1. In the *graph coloring problem* (COL), we are given a simple graph G and we determine whether G has a *k-coloring*, i.e., a labeling of its vertices with at most k colors. The instance (G, k) is represented by the following CSP instance $(V, D, \mathcal{C})$. Let $u_1, \ldots, u_n$ be the vertices of G and let m be the number of edges of G. We define V to be a set of n variables, say $x_1, \ldots, x_n$, and we define D to be a set of k elements viewed as colors. Thus, any labeling of the vertices with colors is represented by an assignment to the variables of V. With each edge $\{u_i, u_j\}$, we associate the constraint $x_i \neq x_j$ where $i < j$. More formally, all constraints in $\mathcal{C}$ have the same constraint relation, namely, the $\neq$ relation on D. For all integers i, j such that $1 \leq i < j \leq n$, there is a constraint with the scope (x_i, x_j) if and only if there is an edge between v_i and v_j. In this representation of COL instances with CSP instances, a labeling for the vertices is a k-coloring if and only if the assignment that represents this labeling is a solution.

For example, let G be the graph in Fig. 1 and $k = 3$. The CSP instance representing (G, k) is shown on the right (the elements b, g, and r of D stand for three colors *blue*, *green*, and *red*).

Example 2. Another natural example is SAT, the satisfiability problem for Boolean formulas in CNF. Let F be a SAT instance, how can we represent F as an "equivalent" CSP instance $(V, D, \mathcal{C})$? The representation is shown using the concrete example of F in Fig. 2.

We define V to be the set of all variables appearing in F, define D to be the set of truth values, and define $\mathcal{C}$ to be a set of constraints where each constraint $C_i = (\mathbf{x}_i, R_i)$ is built from the ith clause in F as follows. First, we rearrange the literals in the clause so that all positive literals (if any) precede the negative ones. The scope $\mathbf{x}_i$ is defined to be the sequence of variables in the rearranged clause. Next, we consider all assignments to the variables in $\mathbf{x}_i$ and we represent them by the corresponding sequences of bits. The constraint relation in C_i is determined by the numbers of positive and negative literals in ith clause. Namely, let p and n be these numbers respectively. Then the relation consists of all tuples of length $(p + n)$ over $\{0, 1\}$ except the tuple that makes the rearranged literals false (p zeroes followed by n ones). Obviously, an assignment to the variables of F, viewed as a function from V to $\{0, 1\}$, satisfies F if and only if this assignment is a solution to $(V, D, \mathcal{C})$.

$$F = (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_4) \wedge (x_1 \vee x_3) \wedge (\overline{x_2} \vee \overline{x_4})$$

$$V = \{x_1, x_2, x_3, x_4\},\ D = \{0,1\},\ \mathcal{C} = \{C_1 \ldots, C_4\}$$

$$C_1 = ((x_2, x_1, x_3), \{0,1\}^3 \setminus \{(0,1,1)\})$$
$$C_2 = ((x_4, x_1, x_2), \{0,1\}^3 \setminus \{(0,1,1)\})$$
$$C_3 = ((x_1, x_3), \{0,1\}^2 \setminus \{(0,0)\})$$
$$C_4 = ((x_2, x_4), \{0,1\}^2 \setminus \{(1,1)\})$$

Figure 2: An instance of SAT represented as a CSP instance.

2.2 Equivalent Definition in Terms of Homomorphisms

It is known that the CSP can equivalently be defined in term of homomorphisms between finite relational structures [FV98, Jea98]. To consider this definition, we first define the notion of a relational structure.

Definition 3 (relational structures)**.** A *relational signature* σ is a finite set of *relation symbols*. Each relation symbol has an associated positive integer called the symbol's *arity*. A *relational structure* $\mathcal{A}$ is a triple (A, σ, I) where A is a nonempty set called the *domain*, σ is a relational signature, and I is an *interpretation function* that maps each k-ary relation symbol of σ to a k-ary relation on A. A relational structure is called *finite* if its domain is finite.

Definition 4 (homomorphisms and isomorphisms)**.** Let $\mathcal{A}$ and $\mathcal{B}$ be relational structures with the same signature. Let A and B be the domains of $\mathcal{A}$ and $\mathcal{B}$ respectively. A function $h : A \to B$ is called a *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$ if h has the following property: for each k-ary relation symbol interpreted as a relation $R^{\mathcal{A}}$ in $\mathcal{A}$ and as a relation $R^{\mathcal{B}}$ in $\mathcal{B}$, we have

$$\mathbf{x} \in R^{\mathcal{A}} \quad \Rightarrow \quad h(\mathbf{x}) \in R^{\mathcal{B}}$$

for all k-tuples $\mathbf{x} \in A^k$. A homomorphism h is an *isomorphism* if h is a bijection and h has a stronger property than above:

$$\mathbf{x} \in R^{\mathcal{A}} \quad \Leftrightarrow \quad h(\mathbf{x}) \in R^{\mathcal{B}}$$

for all k-tuples $\mathbf{x} \in A^k$. Structures $\mathcal{A}$ and $\mathcal{B}$ are *isomorphic* if there is an isomorphism from $\mathcal{A}$ to $\mathcal{B}$.

In the *homomorphism problem*, we are given structures $\mathcal{A}$ and $\mathcal{B}$ with the same signature and we determine whether there exists a homomorphism from $\mathcal{A}$ to $\mathcal{B}$. The restriction of homomorphism problem to finite relational structures is closely connected with the CSP: these two problems reduce to each other in a very natural way. The reductions are described below.

Let $\phi = (V, D, \mathcal{C})$ be a CSP instance and let $\{R_1, \ldots, R_s\}$ be the set of all constraint relations in $\mathcal{C}$. Notice that s may be less than the number of constraints in $\mathcal{C}$ since different constraints can have the same relation. We introduce relations symbols $r_1, \ldots, r_s$ corresponding to the constraint relations and define a signature σ_ϕ as the set of these relation symbols. For example, if ϕ is the CSP instance that represents graph coloring in Fig. 1, then σ_ϕ consists of the $\neq$ symbol only. If ϕ is the CSP instance representing satisfiability in Fig. 2, then σ_ϕ has three symbols: a symbol for the 3-ary relation $\{0,1\}^3 \setminus \{(0,1,1)\}$ and symbols for the binary relations $\{0,1\}^2 \setminus \{(0,0)\}$ and $\{0,1\}^2 \setminus \{(1,1)\}$.

Next, we define relational structures $\mathcal{V}_\phi$ and $\mathcal{D}_\phi$. Both have the signature σ_ϕ defined in the paragraph above. The domain of $\mathcal{V}_\phi$ is V and the domain of $\mathcal{D}_\phi$ is D. The symbols $r_1, \ldots, r_s$ are interpreted in $\mathcal{D}_\phi$ as the constraint relations $R_1, \ldots, R_s$ (for which these symbols were introduced). In the structure $\mathcal{V}_\phi$, each symbol r_i is interpreted as a relation consisting of tuples of variables. Namely, the interpretation of r_i consists of all tuples $\mathbf{x}$ such that $\mathcal{C}$ contains a constraint $(\mathbf{x}, R_i)$. For example, if ϕ is the CSP instance in Fig. 1, then $\mathcal{V}_\phi$ has one relation that consists of the five tuples (x_i, x_j) appearing in the constraints. If ϕ is the CSP instance in Fig. 2, then $\mathcal{V}_\phi$ has three relations: one 3-ary relation with the tuples (x_2, x_1, x_3) and (x_4, x_1, x_2) and two binary relations with one tuple in each.

Definition 5 (relational structures associated with CSP instances)**.** Let ϕ be a CSP instance. We say that the relational structures $\mathcal{V}_\phi$ and $\mathcal{D}_\phi$ described above are *associated* with ϕ. We call $\mathcal{V}_\phi$ the *structure on variables* and call $\mathcal{D}_\phi$ the *structure on the domain*.

Proposition 1. Let ϕ be a CSP instance. An assignment f for ϕ is a solution to ϕ if and only if f is a homomorphism from $\mathcal{V}_\phi$ to $\mathcal{D}_\phi$.

Proof. Obviously follows from the definitions of CSP instances and associated relational structures above. □

According to this proposition, the mapping $\phi \mapsto (\mathcal{V}_\phi, \mathcal{D}_\phi)$ is a polynomial-time reduction of the CSP to the homomorphism problem. This mapping has the inverse for finite relational structures: each pair of such structures determines a CSP instance associated with this pair. This inverse is a polynomial-time reduction of the homomorphism problem restricted to finite relation structures to the CSP.

2.3 Isomorphisms and Automorphisms

Informally, two CSP instances are *isomorphic* if one of them is obtained from the other by permuting variables and permuting values of the domain.

Definition 6 (isomorphism between CSP instances)**.** Let $\phi = (V, D, \mathcal{C}_\phi)$ and $\psi = (V, D, \mathcal{C}_\psi)$ be CSP instances on the same set of variables and the same domain. Let α be a permutation of V and β be a permutation of D. We say that the pair (α, β) is an *isomorphism* from ϕ to ψ if for each tuple $\mathbf{x}$ over V and for each relation R on D,

$$(\mathbf{x}, R) \in \mathcal{C}_\phi \;\Leftrightarrow\; (\alpha(\mathbf{x}), \beta(R)) \in \mathcal{C}_\psi.$$

We say that CSP instances ϕ and ψ are *isomorphic*, written $\phi \simeq \psi$, if there is an isomorphism from ϕ to ψ. The set of all CSP instances isomorphic to ϕ is called the *isomorphism class* of ϕ and denoted by $[\phi]$.

Definition 7 (automorphisms of CSP instances)**.** An isomorphism (α, β) from a CSP instance $\phi = (V, D, \mathcal{C})$ to itself is an *automorphism* of ϕ if (α, β) preserves the set of constraints:

$$(\mathbf{x}, R) \in \mathcal{C} \;\Leftrightarrow\; (\alpha(\mathbf{x}), \beta(R)) \in \mathcal{C}$$

for each tuple $\mathbf{x}$ over V and for each relation R on D. The set of all automorphisms of ϕ forms a group under composition; this group is denoted $Aut(\phi)$.

The following simple statements give equivalent definitions in terms of relational structures associated with CSP instances. The statements use the notions of isomorphisms and automorphisms for relational structures; these notions are defined in the standard way, see for example [Hod97].

Proposition 2. Let $\phi = (V, D, \mathcal{C}_\phi)$ and $\psi = (V, D, \mathcal{C}_\psi)$ be CSP instances. Let α be a permutation of V and β be a permutation of D. The pair (α, β) is an isomorphism from ϕ to ψ if and only if α is an isomorphism from $\mathcal{V}_\phi$ to $\mathcal{V}_\psi$ and β is an isomorphism from $\mathcal{D}_\phi$ to $\mathcal{D}_\psi$.

Proof. Suppose that (α, β) is an isomorphism from ϕ to ψ. Then α is a bijection between the scopes of all constraints in ϕ and the scopes of all constraints in ψ. Moreover, if scopes $\mathbf{x}$ and $\mathbf{y}$ in ϕ have the same relation R, then their images $\alpha(\mathbf{x})$ and $\alpha(\mathbf{y})$ have the same relation $\beta(R)$ in ψ. Therefore, α is an isomorphism from $\mathcal{V}_\phi$ to $\mathcal{V}_\psi$. Likewise, β is a bijection between the tuples of all relations appearing in ϕ and the tuples of all relations appearing in ψ and, hence, β is an isomorphism from $\mathcal{D}_\phi$ to $\mathcal{D}_\psi$. The converse is proved by similar straightforward reasoning. □

Proposition 3. Let $\phi = (V, D, \mathcal{C})$ be a CSP instance. Let α be a permutation of V and β be a permutation of D. The pair (α, β) is an automorphism of ϕ if and only if α is an automorphism of $\mathcal{V}_\phi$ and β is an automorphism of $\mathcal{D}_\phi$.

Proof. Similar to Proposition 2. □

3 Reconstruction of CSP Instances

Informally, a CSP instance ϕ is *k-reconstructible* if its isomorphism class is uniquely determined by the isomorphism classes of all subinstances obtained from ϕ by removing k constraints.

Definition 8 (k-deck)**.** Let $\phi = (V, D, \mathcal{C})$ be a CSP instance and let $m = |\mathcal{C}|$. A *k-card* of ϕ is the isomorphism class of a CSP instance obtained from ϕ by removing k constraints from $\mathcal{C}$ where $1 \le k < m$. The *k-deck* of ϕ is the multiset of all k-cards of ϕ; this multiset consists of $\binom{m}{k}$ elements. If $k = 1$, we call k-cards and k-decks *cards* and *decks*.

Definition 9 (k-reconstruction)**.** A CSP instance ϕ is *k-reconstructible* if for every CSP instance ψ that has the same k-deck as ϕ, we have $\phi \simeq \psi$. We say that ϕ is *reconstructible* if ϕ is 1-reconstructible.

The following theorem gives a sufficient condition for k-reconstruction of CSP instances; its proof is essentially an application of the technique used by Alon, Caro, Krasikov, Roditty in [ACKR89].

Theorem 4. Let $\phi = (V, D, \mathcal{C})$ be a CSP instance with $|V| = n$, $|D| = d$, and $|\mathcal{C}| = m$. Let $1 \le k < m$. Suppose that

$$2^{m-k} \;>\; \frac{n! \cdot d!}{|Aut(\phi)|} \tag{1}$$

where $|Aut(\phi)|$ denotes the order of the automorphism group of ϕ. Then ϕ is k-reconstructible.

Proof. We use the following notion of *k-reconstructible subsets* from [ACKR89]. Let X be a set and Γ be a group acting on X. This action determines the following equivalence relation on the subsets of X: subsets $Y_1, Y_2 \subseteq X$ are *equivalent* if Γ contains an element g that moves Y_1 to Y_2, i.e., Y_2 is obtained from Y_1 by applying the permutation on X corresponding to g. Let $Y \subseteq X$ consist of m elements and let $1 \le k < m$. Consider all subsets of cardinality $m - k$ of Y. The multiset of their equivalence classes is called the *k-deck* of Y. We say that Y is *k-reconstructible* if its k-deck determines the equivalence class of Y.

Corollary 2.4 in [ACKR89] gives the following sufficient condition for a subset to be k-reconstructible. Let $Y \subseteq X$ be a subset of cardinality m. Let Γ_Y be the subgroup of Γ that

consists of all "automorphisms" of Y, i.e., the subgroup of all elements of Γ moving Y to itself. Let Γ and Γ_Y denote the orders of the groups. If

$$2^{m-k} > |\Gamma|/|\Gamma_Y| \tag{2}$$

then Y is k-reconstructible.

To apply this condition to CSP instances, we need to define X and Γ so that the subsets of X represent CSP instances and the equivalence relation on X represents isomorphism between CSP instances. Consider the set of all CSP instances $(V, D, \mathcal{C})$ where V and D are fixed. We define X to be the set of all possible constraints over V and D, i.e., X consists of all pairs $(\mathbf{x}, R)$ where $\mathbf{x}$ is a tuple of variables and R is a relation on D such that $\mathbf{x}$ and R have the same arity. Thus, there is a bijection between the subsets of X and all CSP instances over V and D. Next, we define Γ to be the direct product of the symmetric group on V and the symmetric group on D. This group acts on X: for an element $g = (\alpha, \beta)$ of Γ and for an element $C = (\mathbf{x}, R)$ of X, we define $g \cdot C$ to be $(\alpha(\mathbf{x}), \beta(R))$. The group action partitions the power set of X into equivalence classes in exactly the same way as described above. Subsets $Y_1, Y_2 \subseteq X$ are equivalent if and only if the CSP instances corresponding to Y_1 and Y_2 are isomorphic.

Consider a subset $Y \subseteq X$, an element (α, β) of Γ that moves Y to itself, and the CSP instance $\phi = (V, D, \mathcal{C})$ where $\mathcal{C}$ is Y. By Definition 7, the pair (α, β) is an automorphism of the CSP instance $(V, D, \mathcal{C})$ where $\mathcal{C}$ is Y. Therefore, the subgroup Γ_Y in (2) is the automorphism group of this instance. Also, by our definition of Γ, its order is the product of the orders of the symmetric groups on V and D. Hence, the claim (1) follows from (2). □

Corollary 5. In the setting of Theorem 4, if

$$2^{m-k} > \frac{n!}{ord(Aut(\mathcal{V}_\phi))} \cdot \frac{d!}{ord(Aut(\mathcal{D}_\phi))}$$

then ϕ is k-reconstructible.

Proof. It follows from Proposition 3 that for each CSP instance ϕ, the automorphism group $Aut(\phi)$ is the direct product:

$$Aut(\mathcal{V}_\phi) \times Aut(\mathcal{D}_\phi)$$

where $Aut(\mathcal{V}_\phi)$ and $Aut(\mathcal{D}_\phi)$ are the automorphism groups of the relational structures $\mathcal{V}_\phi$ and $\mathcal{D}_\phi$ respectively. □

4 Number of Solutions

In this section, we consider the question of whether or not a given property or parameter of a CSP instance ϕ is uniquely determined by the k-deck of ϕ.

Definition 10 (k-reconstruction for properties and parameters)**.** Let f be a function defined on all CSP instances. Given a CSP instance ϕ, we say that f is *k-reconstructible on ϕ* if we have $f(\phi) = f(\psi)$ for every CSP instance ψ that has the same deck as ϕ. We say that f is *reconstructible on ϕ* if f is 1-reconstructible on ϕ.

A function f is called an *invariant* if f is preserved under isomorphism between CSP instances, i.e., $f(\phi) = f(\psi)$ whenever $\phi \simeq \psi$. Notice that if f is k-reconstructible on ϕ then f is an invariant on ϕ. How can we characterize CSP instances for which the converse is true as well? Theorem 6 below characterizes CSP instances for which the number of solutions is reconstructible. The characterization is given in terms of the *complements* of CSP instances.

Definition 11 (complements)**.** Let R be a k-ary relation on a set D. The *complement* of R, denoted by $\overline{R}$, is the set of all k-tuples over D that do not belong to R, i.e., $\overline{R} = D^k - R$. The *complement* of a constraint $C = (\mathbf{x}, R)$ is the constraint $\overline{C} = (\mathbf{x}, \overline{R})$. If $\mathcal{C}$ is a set of constraints, we write $\overline{\mathcal{C}}$ to denote the set of the complements of the constraints of $\mathcal{C}$. Let $\phi = (V, D, \mathcal{C})$ be a CSP instance. The *complement* of ϕ, denoted by $\overline{\phi}$, is the CSP instance $(V, D, \overline{\mathcal{C}})$.

Theorem 6. The number of solutions to a CSP instance ϕ is reconstructible if and only if the number of solutions to its complement $\overline{\phi}$ is reconstructible.

Proof. For every CSP instance $\phi = (V, D, \mathcal{C})$, we write $\#\phi$ or $\#(V, D, \mathcal{C})$ to denote the number of solutions to ϕ. This number can be counted using the following equality:

$$\#(V, D, \mathcal{C}) = \sum_{\mathcal{S} \subseteq \mathcal{C}} (-1)^{|\mathcal{S}|} \#(V, D, \overline{\mathcal{S}}). \tag{3}$$

This equality follows from the inclusion-exclusion principle applied to the set of solutions to ϕ. Also, when thinking of a solution to ϕ as a homomorphism from the structure on variables $\mathcal{V}_\phi$ to the structure on the domain $\mathcal{D}_\phi$, we can view (3) as a generalization of the similar formula in Lovász's paper [Lov72] used there for counting graph monomorphisms.

Now, we consider CSP instances $\phi = (V, D, \mathcal{C}_\phi)$ and $\psi = (V, D, \mathcal{C}_\psi)$ that have the same deck and we prove

$$\#\phi = \#\psi \quad \Leftrightarrow \quad \#\overline{\phi} = \#\overline{\psi}. \tag{4}$$

The numbers of solutions to ϕ and ψ can be counted using (3) and writing the right-hand side of (3) as the sum of the terms corresponding to all proper subsets of $\mathcal{C}$ plus the term corresponding to $\mathcal{C}$:

$$\#\phi = \sum_{\mathcal{S} \subset \mathcal{C}_\phi, \mathcal{S} \neq \mathcal{C}_\phi} (-1)^{|\mathcal{S}|} \#(V, D, \overline{\mathcal{S}_\phi}) + (-1)^{|\mathcal{C}_\phi|} \#(V, D, \overline{\mathcal{C}_\phi}) \tag{5}$$

$$\#\psi = \sum_{\mathcal{S} \subset \mathcal{C}_\psi, \mathcal{S} \neq \mathcal{C}_\psi} (-1)^{|\mathcal{S}|} \#(V, D, \overline{\mathcal{S}_\psi}) + (-1)^{|\mathcal{C}_\psi|} \#(V, D, \overline{\mathcal{C}_\psi}) \tag{6}$$

To prove (4), we show that the sums over proper subsets in the right-hand sides of (5) and (6) are equal:

$$\sum_{\mathcal{S} \subset \mathcal{C}_\phi, \mathcal{S} \neq \mathcal{C}_\phi} (-1)^{|\mathcal{S}|} \#(V, D, \overline{\mathcal{S}_\phi}) = \sum_{\mathcal{S} \subset \mathcal{C}_\psi, \mathcal{S} \neq \mathcal{C}_\psi} (-1)^{|\mathcal{S}|} \#(V, D, \overline{\mathcal{S}_\psi}). \tag{7}$$

Let $m = |\mathcal{C}_\phi|$ (since ϕ and ψ have the same deck, m is also the number of constraints in ψ), Let $1 \leq k < m$. Consider a subset $\mathcal{S}_\phi \subset \mathcal{C}_\phi$ of cardinality $m - k$ and the number

$$\#(V, D, \overline{\mathcal{S}_\phi})$$

occurring in the left-hand side of (7). Obviously, this number of solutions is uniquely determined by the isomorphism class of $(V, D, \overline{\mathcal{S}_\phi})$. Moreover, since a constraint and its complement uniquely determine each other, this number is uniquely determined by the isomorphism class of $(V, D, \mathcal{S}_\phi)$ as well. Hence, the sum in the left-hand side of (7) is uniquely determined by the k-deck of ϕ. Likewise, the sum in the right-hand of (7) is uniquely determined by the k-deck of ψ.

It is easy to see that if two CSP instances have the same deck then they have the same k-deck for every k less than the number of constraints. Therefore, the sums in both sides of (7)

are determined by the same multiset of isomorphism classes, which proves (7). Combining the equalities (5)-(7), we obtain

$$\#\phi - \#\psi \quad = \quad (-1)^m \#\overline{\phi} \; - \; (-1)^m \#\overline{\psi}.$$

Hence, $\#\phi = \#\psi$ if and only if $\#\overline{\phi} = \#\overline{\psi}$. □

The following corollaries illustrate how Theorem 6 can be applied. Corollary 7 gives a simple proof of the known result that the number of graph colorings is reconstructible, which follows from Tutte's theorem in [Tut79]. Corollary 8 shows that for Boolean formulas in CNF represented as CSP instances, the number of satisfying assignments is reconstructible.

Corollary 7. Let ϕ be a CSP instance that represents an instance of the graph coloring problem (`COL`) as described in Fig. 1. Then $\#\phi$ is reconstructible.

Proof. Let G_ϕ be the underlying graph and k_ϕ be the number of colors. Consider the complement $\overline{\phi}$ and a solution to it. This solution corresponds to a coloring of G_ϕ with at most k_ϕ colors such that in each connected component of G_ϕ, all vertices have the same color. Thus, the number of solutions to $\overline{\phi}$ is uniquely determined by k_ϕ and the number of connected components of G_ϕ. It is easy to see that both of these parameters are reconstructible from the deck of $\overline{\phi}$. By Theorem 6, $\#\phi$ is reconstructible. □

Corollary 8. Let ϕ be a CSP instance that represents an instance of `SAT` as described in Fig. 2. If ϕ contains at least three constraints then the number of solutions to ϕ is reconstructible.

Proof. According to our representation of `SAT` instances, each constraint in ϕ can be viewed a disjunction $\ell_1 \vee \ldots \vee \ell_k$ of literals. Then, by Definition 11, its complement in $\overline{\phi}$ represents the conjunction $\overline{\ell}_1 \wedge \ldots \wedge \overline{\ell}_k$. For every constraint C in $\overline{\phi}$, let $L(C)$ denote the set of literals represented by C. Let L denote the union $\cup L(C)$ over all constraints C in $\overline{\phi}$.

First, we consider the case that $\overline{\phi}$ has no solution. This case holds if and only if the set L contains a pair of complementary literals. It is clear that if $\overline{\phi}$ has at least three constraints, then each pair of constraints, up to isomorphism, appears together in some card. Hence, the existence of a pair of constraint representing complementary literals can be detected from the deck. Thus, $\#\overline{\phi}$ is reconstructible for every $\overline{\phi}$ that has at least three constraints and has no solution.

Next, we consider the case that $\overline{\phi}$ has a solution, which is equivalent to the case that L does not contain any pair of complementary literals. Let V be the set of variables of $\overline{\phi}$. Since L does not contain complementary literals, any assignment to the variables of V is a solution and $\#\overline{\phi} = 2^{|V|}$. By Definition 8, any card of $\overline{\phi}$ has the same set V as $\overline{\phi}$ itself. Therefore, $\#\overline{\phi}$ is reconstructible for every $\overline{\phi}$ that has a solution.

Combining the two cases and Theorem 6, we obtain the claim. □

5 Concluding Remarks

This paper defines the notion of k-reconstructible CSP instances and considers the question of what CSP instances and what their invariants are reconstructible. There are many other questions of interest, not mentioned in the paper. Here are some of them.

The notion of reconstruction depends on how isomorphism and automorphisms of CSP instances are defined. In this paper, an isomorphism is a pair of permutations: a permutation of the set of variables and a permutation of the domain. This definition seems the most natural

but still, there are other approaches to defining isomorphisms and symmetries of CSP instances, see for example [CJJ+06]. What about the corresponding notions of reconstruction, do they differ significantly from the notion defined in this paper?

Almost all graphs are reconstructible [Mül76] and, moreover, almost all graphs are reconstructible from their sub-decks consisting of three cards [Bol90]. These results are proved for $\mathbb{G}(n,p)$, the most common model of random graphs, where each possible edge in a graph on n vertices occurs independently with probability p. There are different models of random CSP instances [AMK+01], do similar results hold for some of them?

What could be possible applications of the notion of k-reconstructible CSP instances? Can this notion be useful for modeling some systems or processes, as reconstruction of jigsaw puzzles was useful for modeling DNA sequence assembly [MR15, NPS17, BBN19]? Or, can this notion be useful for solving some problems in the CSP field, as edge reconstruction was useful for proving upper bounds on the order of the automorphism group of a graph [KLT02]?

References

[ACKR89] Noga Alon, Yair Caro, Ilia Krasikov, and Yehuda Roditty. Combinatorial reconstruction problems. *Journal of Combinatorial Theory, Series B*, 47(2):153–161, 1989.

[AMK+01] Dimitris Achlioptas, Michael S. O. Molloy, Lefteris M. Kirousis, Yannis C. Stamatiou, Evangelos Kranakis, and Danny Krizanc. Random constraint satisfaction: A more accurate picture. *Constraints*, 6(4):329–344, 2001.

[BBN19] Paul Balister, Béla Bollobás, and Bhargav Narayanan. Reconstructing random jigsaws. In *Multiplex and Multilevel Networks*, pages 31–50. Oxford University Press, 2019.

[BEH+17] Nathan Bowler, Joshua Erde, Peter Heinig, Florian Lehner, and Max Pitz. A counterexample to the reconstruction conjecture for locally finite trees. *Bulletin of the London Mathematical Society*, 49(4):630–648, 2017.

[Ber72] Claude Berge. Isomorphism problems for hypergraphs. In *Proceedings of the 1st Working Seminar on Hypergraphs*, volume 411 of *Lecture Notes in Mathematics*, pages 1–12. Springer, 1972.

[Bol90] Béla Bollobás. Almost every graph has reconstruction number three. *Journal of Graph Theory*, 14(1):1–4, 1990.

[CJJ+06] David A. Cohen, Peter Jeavons, Christopher Jefferson, Karen E. Petrie, and Barbara M. Smith. Symmetry definitions for constraint satisfaction problems. *Constraints*, 11(2-3):115–137, 2006.

[DS03] Miroslav Dudík and Leonard Schulman. Reconstruction from subsequences. *Journal of Combinatorial Theory, Series A*, 103(2):337–348, 2003.

[DW18] Evgeny Dantsin and Alexander Wolpert. Reconstruction of boolean formulas in conjunctive normal form. In *Proceedings of the 24th International Computing and Combinatorics Conference, COCOON 2018*, volume 10976 of *Lecture Notes in Computer Science*, pages 592–601. Springer, 2018.

[FV98] Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.

[Gre71] D. L. Greenwell. Reconstructing graphs. *Proceedings of the American Mathematical Society*, 30(3):431–433, 1971.

[Har64] Frank Harary. On the reconstruction of a graph from a collection of subgraphs. In *Theory of Graphs and its Applications*, pages 47–52, 1964.

[Hod97] Wilfrid Hodges. *A Shorter Model Theory*. Cambridge University Press, 1997.

[Jea98] Peter Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1-2):185–204, 1998.

[Kel57] Paul J. Kelly. A congruence theorem for trees. *Pacific Journal of Mathematics*, 7(1):961–968, 1957.

[KLS09] Géza Kós, Péter Ligeti, and Péter Sziklai. Reconstruction of matrices from submatrices. *Mathematics of Computation*, 78(267):1733–1747, 2009.

[KLT02] Ilia Krasikov, Arieh Lev, and Bhalchandra D. Thatte. Upper bounds on the automorphism group of a graph. *Discrete Mathematics*, 256(1-2):489–493, 2002.

[Koc87] William L. Kocay. A family of nonreconstructible hypergraphs. *Journal of Combinatorial Theory, Series B*, 42(1):46–63, 1987.

[KR97] Ilia Krasikov and Yehuda Roditty. On a reconstruction problem for sequences. *Journal of Combinatorial Theory, Series A*, 77(2):344–348, 1997.

[Lov72] László Lovász. A note on the line reconstruction problem. *Journal of Combinatorial Theory, Series B*, 13:309–310, 1972.

[LS16] Josef Lauri and Raffaele Scapellato. *Topics in Graph Automorphisms and Reconstruction*. London Mathematical Society Lecture Note Series. Cambridge University Press, 2nd edition, 2016.

[McK97] Brendan D. McKay. Small graphs are reconstructible. *The Australasian Journal of Combinatorics*, 15:123–126, 1997.

[MMS+91] Bennet Manvel, Aaron Meyerowitz, Allen Schwenk, Ken Smith, and Paul Stockmeyer. Reconstruction of sequences. *Discrete Mathematics*, 94(3):209–219, 1991.

[MR15] Elchanan Mossel and Nathan Ross. Shotgun assembly of labeled graphs. *CoRR*, abs/1504.07682, 2015.

[Mül76] Vladimir Müller. Probabilistic reconstruction from subgraphs. *Commentationes Mathematicae Universitatis Carolinae*, 17(4):709–719, 1976.

[Mül77] Vladimir Müller. The edge reconstruction conjecture is true for graphs with more than $n \log_2 n$ edges. *Journal of Combinatorial Theory, Series B*, 22:281–283, 1977.

[NPS17] Rajko Nenadov, Pascal Pfister, and Angelika Steger. Unique reconstruction threshold for random jigsaw puzzles. *Chicago Journal Theoretical Computer Science*, 2017, 2017.

[NW78] Crispin Nash-Williams. The reconstruction problem. In *Selected Topics in Graph Theory*, chapter 8, pages 205–235. Academic Press, 1978.

[Sto77] Paul K. Stockmeyer. The falsity of the reconstruction conjecture for tournaments. *Journal of Graph Theory*, 1(1):19–25, 1977.

[Tut79] William T. Tutte. All the king's horses. a guide to reconstruction. In *Graph Theory and Related Topics*, pages 15–33. Academic Press, 1979.