# Tractable conservative Constraint Satisfaction Problems

Andrei A. Bulatov
Oxford University Computing Laboratory, Oxford, UK
*e-mail: Andrei.Bulatov@comlab.ox.ac.uk*

## Abstract

*In a constraint satisfaction problem (CSP) the aim is to find an assignment of values to a given set of variables, subject to specified constraints. The CSP is known to be NP-complete in general. However, certain restrictions on the form of the allowed constraints can lead to problems solvable in polynomial time. Such restrictions are usually imposed by specifying a constraint language. The principal research direction aims to distinguish those constraint languages which give rise to tractable CSPs from those which do not.*

*We achieve this goal for the widely used variant of the CSP, in which the set of values for each individual variable can be restricted arbitrarily. Restrictions of this type can be expressed by including in a constraint language all possible unary constraints. Constraint languages containing all unary constraints will be called conservative. We completely characterize conservative constraint languages that give rise to CSP classes solvable in polynomial time. In particular, this result allows us to obtain a complete description of those (directed) graphs $H$ for which the* LIST $H$-COLORING *problem is polynomial time solvable.*

## 1. Introduction

In a Constraint Satisfaction Problem (CSP) the aim is to find an assignment of values to a given set of variables, subject to constraints on the values which can be assigned simultaneously to certain specified subsets of variables. A CSP can also be expressed as the problem of deciding whether a given conjunctive formula has a model, or as the problem of deciding whether there exists a homomorphism between two relational structures.

The general CSP is NP-complete [30]. However, many practical and theoretical problems can be expressed in terms of CSP using constraints of a certain restricted form so that the restricted problems can be shown to be tractable (i.e., in **PTIME**). As constraints are usually specified by relations, the constraint satisfaction problem can therefore be restricted by specifying a *constraint language*, that is a set of relations that are allowed to be used as constraints. In the homomorphism form of the CSP this is equivalent to restricting of a possible form of the target structure. The problem of distinguishing those constraint languages which give rise to a tractable problems from those which do not is acknowledged to be very deserving and attracts much attention (see e.g., [2, 13, 20]). We shall call this problem the *classification problem*.

The classification problem was completely solved for Boolean constraint satisfaction problem (that is over a 2-element domain) [35] which is equivalent to the propositional SATISFIABILITY problem. In [2], this problem was solved for CSPs over a 3-element set of values. In the case of a bigger domain the classification problem is believed to be very hard and, in spite of substantial progress achieved during last decade [8, 10, 14, 20, 23, 26, 24, 28], it withstands all attacks.

Remarkably, all studied CSP classes turn out to be either tractable or NP-complete. Dichotomy results of this type are of particular interest in the study of CSP, because, on the one hand, they determine the precise complexity of a problem, and on the other hand, the a priory existence of a dichotomy result cannot be taken for granted. In [20], it was conjectured that a dichotomy theorem holds for the complexity of an arbitrary restricted CSP, and in [10], a possible criteria for tractable problems has been suggested. For a short survey of dichotomy results the reader is referred to [13].

In this paper we study the classification problem for the well-known and widely used variant of the constraint satisfaction problem, in which the set of values for each individual variable can be restricted arbitrarily. As is easily seen, the availability of this type of restrictions is equivalent to including into a constraint language all possible unary relations, or including into the target structure all possible unary predicates. A constraint language satisfying this condition is said to be *conservative*.

Constraint satisfaction problems related to some particular types of conservative constraint languages have already been studied. In [11], a dichotomy theorem has been proved for such languages containing all permutation relations; and in [6], conservative constraint languages related to binary operations have been studied.

We completely characterize conservative constraint languages that give rise to tractable constraint satisfaction problems, and provide a polynomial time algorithm (which is quite non-trivial) solving the problem in the tractable cases. In particular, this algorithm allows one to solve numerous problems that cannot be solved by known algorithms. We also prove that in all other cases the constraint satisfaction problem is NP-complete.

It is somewhat unexpected that, despite the result is applicable to CSPs over sets of arbitrary size, the form of the criteria it states is quite similar to that found by T.Schaefer [35] for SATISFIABILITY problem. By contrast with the earlier results, which are concerned with CSPs either over a very small set of values [35, 2, 13], or using constraints of a very restricted form [20, 23, 19, 26, 24, 6, 9, 3], or solvable by some particular algorithm [20, 15], the result of this paper yields, probably, the first dichotomy theorem for a really wide class of constraint satisfaction problems, involving constraints of all types subject to no restriction (but finiteness) on the size of the set of values.

We apply the main result to $H$-COLORING problem, an intensively studied combinatorial problem that can be naturally formulated as a restricted CSP [23, 21, 17]. In $H$-COLORING the question is whether there exists a homomorphism from a given graph $G$ to a fixed graph $H$. This problem is equivalent to the CSP for a constraint language consisting of one binary relation. The conservative version of $H$-COLORING problem is called LIST $H$-COLORING [29, 18, 19]. In this problem the homomorphism sought is subject to restrictions on possible images of vertices of the input graph. Therefore, a complete description of those directed graphs $H$ for which LIST $H$-COLORING is solvable in polynomial time can be deduced from our result.

The result, the solving algorithm, and the proofs heavily use the algebraic approach to CSP developed in [26, 24, 10, 8, 7]. This method builds on the fact that one can extract much information about the complexity of a restricted constraint satisfaction problem from knowing certain operations, called polymorphisms, connected to the constraint relations.

Proofs of auxiliary results can be found in the full version of the paper [5].

## 2. Constraint Satisfaction Problem

Let $A$ be a finite set. An *$n$-ary relation* on $A$ is a set of $n$-tuples of elements from $A$; we use $R_A$ to denote the set of all finitary relations on $A$. A *constraint language* is a subset of $R_A$ (finite or infinite).

**Definition 1** *The* constraint satisfaction problem over a constraint language $\Gamma \subseteq R_A$, *denoted* $\mathrm{CSP}(\Gamma)$, *is defined to be the decision problem with instance $\mathcal{P} = (V; A; \mathcal{C})$, where*

- *$V$ is a finite set of* variables,
- *$A$ is a* set of values *(sometimes called a* domain*), and*
- *$\mathcal{C}$ is a set of* constraints, *in which each constraint $C \in \mathcal{C}$ is a pair $\langle s, \varrho\rangle$ with $s$ a list of variables of length $m_C$, called the* constraint scope, *and $\varrho$ an $m_C$-ary relation on $A$, belonging to $\Gamma$, called the* constraint relation.

*The question is whether there exists a* solution *to $\mathcal{P}$, that is, a mapping $\varphi\colon V \to A$ such that, for each constraint in $\mathcal{C}$, the image of the constraint scope is a member of the constraint relation.*

The size of a problem instance is the length of encoding of all tuples in all constraint relations.

**Example 1** (SATISFIABILITY) An instance of standard propositional SATISFIABILITY problem [22, 31] is specified by giving a propositional formula $\mathcal{F} = F_1 \wedge \cdots \wedge F_n$ in CNF and asking whether $\mathcal{F}$ is satisfiable. As is easily seen, it is equivalent to the CSP instance $(V, \{0, 1\}, \mathcal{C})$ where $V$ is the set of all variables appearing in $\mathcal{F}$ and $\mathcal{C} = \{\langle s_1, \varrho_1\rangle, \ldots, \langle s_n, \varrho_n\rangle\}$ where $s_i$ is the list of variables underlying the literals from $F_i$, and $\varrho_i$ consists of all tuples that make $F_i$ true. □

**Example 2** (H-COLORING) Let $H$ be a (directed) graph. In the $H$-COLORING problem we are asked whether there is a homomorphism from a given graph $G$ to $H$.

Every (directed) graph $H = (V(H), E(H))$ corresponds to a binary relation $\varrho_H$: $(a, b) \in \varrho_H$ iff $(a, b)$ is an edge of $H$. Then every instance $G = (V(G); E(G))$ of $H$-COLORING problem corresponds to the instance $(V(G); V(H); \{\langle(a, b), \varrho_H\rangle \mid (a, b) \in E(G)\})$ of $\mathrm{CSP}(\varrho_H)$.

The GRAPH $q$-COLORABILITY problem is a particular case of $H$-COLORING in which $H$ is the $q$-vertex complete graph without loops. Then $\varrho_H$ is the *disequality* relation $\neq_{V(H)} = \{(a, b) \in (V(H))^2 \mid a \neq b\}$. □

Several different definitions of the Constraint Satisfaction Problem have appeared in literature, some of them are equivalent to that given above, some are not. In this paper we concentrate on one of the standard versions of the CSP which we will call the *conservative CSP*.

**Definition 2** *The* conservative constraint satisfaction problem over a constraint language $\Gamma \subseteq R_A$, *denoted* c-$\mathrm{CSP}(\Gamma)$, *is defined to be the decision problem with instance $\mathcal{P} = (V; A; \mathcal{L}; \mathcal{C})$, where $V, A, \mathcal{C}$ are defined as for the usual CSP, and $\mathcal{L}$ is a collection of subsets $L_v$ $(v \in V)$ from $A$. The question is whether there exists a solution $\varphi$ to $\mathcal{P}$, defined as before, such that $\varphi(v) \in L_v$ for all $v \in V$.*

**Example 3** (LIST-H-COLORING) Let $H$ be a (directed) graph. In LIST $H$-COLORING problem we are given a graph $G$ and, for each vertex $v$ of $G$, a list $L_v$ of vertices of $H$. The question is whether there is a homomorphism $\varphi$ from $G$ to $H$ such that $\varphi(v) \in L_v$ for every vertex $v$ of $G$. Clearly, LIST $H$-COLORING can be represented in the form of the conservative CSP as in Example 2. □

Notice that, in fact, for any constraint language $\Gamma \subseteq R_A$, the problem c-CSP$(\Gamma)$ is equivalent to $\mathrm{CSP}(\Gamma \cup 2^A)$, where $2^A$ denotes the set of all subsets from $A$. A constraint language $\Gamma$ with $2^A \subseteq \Gamma$ will be called *conservative*. Therefore, we may get rid of conservative constraint satisfaction problems, and, instead, consider problems of the form $\mathrm{CSP}(\Gamma)$ for conservative constraint languages $\Gamma$. We are concerned with the following research problems.

**Problem 1 (Classification problem)** *Characterize conservative constraint languages, $\Gamma$, for which $\mathrm{CSP}(\Gamma)$ is tractable.*

**Problem 2 (Algorithmic problem)** *Find efficient (polynomial time) algorithms for tractable problems $\mathrm{CSP}(\Gamma)$, $\Gamma$ conservative.*

**Problem 3 (Meta-problem)** *Find an efficient (polynomial time) algorithm, if any, that decides whether a conservative constraint language is tractable or not.*

In the important particular case of Boolean CSP (i.e., SATISFIABILITY problem), and in the case of CSP over a 3-element domain, all these problems have been completely solved [35, 28, 2, 4] for arbitrary constraint languages, not only conservative. The result for SATISFIABILITY is known as Schaefer's Dichotomy Theorem. We revisit those results later. Now we just notice that in both cases, for every constraint language $\Gamma$, $\mathrm{CSP}(\Gamma)$ is either tractable or NP-complete, that allows us to pose another research problem.

**Problem 4 (Dichotomy problem)** *Is it true that every problem of the form $\mathrm{CSP}(\Gamma)$, $\Gamma$ conservative, is either tractable or NP-complete?*

## 3. Polymorphisms and tractability

The algebraic approach to the study of CSP uses, in addition to relations, arbitrary *operations* on the set of values. The set of all finitary operations on $A$ (that is mappings $f\colon A^n \to A$, $n \in \mathbb{N}$) will be denoted by $O_D$.

Any operation on $A$ can be extended in a standard way to an operation on tuples over $A$, as follows. The components of an $m$-tuple $\mathbf{a}$ are denoted $\mathbf{a}[1], \ldots, \mathbf{a}[m]$. For any $n$-ary operation $f \in O_A$, and any collection of tuples $\mathbf{a}_1, \ldots, \mathbf{a}_n \in A^m$, define

$$f(\mathbf{a}_1, \ldots, \mathbf{a}_n) = (f(\mathbf{a}_1[1], \ldots, \mathbf{a}_n[1]), \ldots, f(\mathbf{a}_1[m], \ldots, \mathbf{a}_n[m])).$$

**Definition 3** *For any $m$-ary relation $\varrho \in R_A$, and any $n$-ary operation $f \in O_A$, if $f(\mathbf{a}_1, \ldots, \mathbf{a}_n) \in \varrho$ for all choices of $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \varrho$, then $\varrho$ is said to be* invariant *under $f$, and $f$ is called a* polymorphism *of $\varrho$.*

**Example 4** Let $\varrho$ be the solution space of a system of linear equations over a field $F$. Then the operation $m(x, y, z) = x - y + z$ is a polymorphism of $\varrho$. Indeed, let $A \cdot \mathbf{x} = \mathbf{b}$ be a system defining $\varrho$, and $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \varrho$. Then

$$A \cdot m(\mathbf{x}, \mathbf{y}, \mathbf{z}) = A \cdot \mathbf{x} - A \cdot \mathbf{y} + A \cdot \mathbf{z} = \mathbf{b} - \mathbf{b} + \mathbf{b} = \mathbf{b}.$$

In fact, the converse can also be shown: if $\varrho$ is invariant under $m$ then it is the solution space of a certain system of linear equations. □

The set of all relations that are invariant under each operation from some set $C \subseteq O_A$ will be denoted by $\mathsf{Inv}\ C$. The set of all operations that are polymorphisms of every relation from some set $\Gamma \subseteq R_A$ will be denoted by $\mathsf{Pol}\ \Gamma$. The operators $\mathsf{Inv}$ and $\mathsf{Pol}$ form a Galois correspondence between $R_A$ and $O_A$ (see Proposition 1.1.14 of [33]). A basic introduction to this correspondence can be found in [32], and a comprehensive study in [33]. The following result describes a connection between polymorphisms and the complexity of CSP.

**Theorem 1 ([24])** *Let $\Gamma_1$ and $\Gamma_2$ be constraint languages over a finite set, such that $\Gamma_2$ is finite. If $\mathsf{Pol}\ \Gamma_1 \subseteq \mathsf{Pol}\ \Gamma_2$ then $\mathrm{CSP}(\Gamma_2)$ is polynomial time reducible to $\mathrm{CSP}(\Gamma_1)$.*

The theorem amounts to say that the complexity of $\mathrm{CSP}(\Gamma)$, $\Gamma$ finite, is completely determined by the polymorphisms of $\Gamma$. Moreover, this is true for all investigated infinite constraint languages.

A number of results on the complexity of constraint satisfaction problems have been obtained via this approach (e.g., [2, 3, 6, 8, 10, 9, 14, 26, 24]. In particular, it was proved that operations of certain types yield tractability.

**Theorem 2 ([26, 24, 6, 3])** *If one of the following operations is a polymorphism of a constraint language $\Gamma$ over a finite set $A$, then $\mathrm{CSP}(\Gamma)$ is tractable:*

- *a* semilattice *operation, that is a binary operation $f$ satisfying the conditions: (a) $f(a, a) = a$; (b) $f(a, b) = f(b, a)$, (c) $f(f(a, b), c) = f(a, f(b, c))$, for all $a, b \in A$;*
- *a* majority *operation, that is a ternary operation $g$ such that $g(a, a, b) = g(a, b, a) = g(b, a, a) = a$, for all $a, b \in A$.*
- *a* Mal'tsev *operation, that is a ternary operation $h$ such that $h(a, a, b) = h(b, a, a) = b$, for all $a, b \in A$.*

Moreover, Schaefer's Dichotomy Theorem [35], when appropriately re-stated, easily follows from Theorems 1,2 and well known algebraic results [34] (see [24]).

**Theorem 3 ([35, 24])** *For any* $\Gamma \subseteq R_{\{0,1\}}$, $\mathrm{CSP}(\Gamma)$ *is tractable when* $\mathsf{Pol}\ \Gamma$ *contains one of the following:*

- *the constant* 0 *or the constant* 1 *operation;*
- *the conjunction or the disjunction operation (both of them are semilattice operations);*
- *the majority operation* $(x \vee y) \wedge (y \vee z) \wedge (z \vee x)$*;*
- *the Mal'tsev operation* $x - y + z \pmod 2$ *(it is also called the* affine *operation).*

*In all other cases* $\mathrm{CSP}(\Gamma)$ *is NP-complete.*

Originally, this theorem was stated in terms of propositional formulas in CNF and their clauses rather than that of operations. The correspondence between the listed operations and properties of CNFs is the following. The constant 0 and 1 operations correspond to 0- and 1-valid formulas respectively; the disjunction and conjunction correspond to Horn and anti-Horn formulas; the majority operation correspond to bijunctive formulas; and the affine operation correspond to systems of linear equations.

Schaefer's Dichotomy Theorem completely solves Problems 1, 2, and 4 for Boolean constraint languages. Problem 3 has been explicitly solved in [28].

## 4. A dichotomy theorem for conservative constraint languages

First, we show that the NP-completeness part of our dichotomy theorem can be easily deduced from the existing results.

It has been observed [26, 27, 10] that there are two benchmark problems that explain the NP-completeness of all known NP-complete CSPs. These are the problem $\mathrm{CSP}(\{\varrho\})$,

$$\varrho = \left( \begin{array}{cccccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{array} \right),$$

in Boolean case (the tuple are written vertically), and $\mathrm{CSP}(\{\neq_A\})$ in non-Boolean case.

An operation $f$ is a polymorphisms of one those relations if and only if $f$ is an *essentially unary surjective operation*, that is $f(x_1, \ldots, x_n) = g(x_i)$ where $g$ is a bijection [26, 27]. Therefore, by Theorem 1, if every polymorphism of a constraint language, $\Gamma$, is an essentially unary surjective operation, then $\mathrm{CSP}(\Gamma)$ is NP-complete.

Note that every ($n$-ary) polymorphism $f$ of a conservative constraint language satisfies the condition: $f(a_1, \ldots, a_n) \in \{a_1, \ldots, a_n\}$, for all $a_1, \ldots, a_n \in A$. Such an operation is said to be *conservative*. For example, the operations listed in Theorem 3, excluding the constant ones, are conservative. By $f|_B$ we denote the restriction of an operation $f$ onto a set $B$. The following result from [10] provides a necessary condition for the tractability of $\mathrm{CSP}(\Gamma)$.

**Proposition 1 ([10])** *Let* $\Gamma$ *be a constraint language on a finite set* $A$*, and a subset* $B \subseteq A$, $|B| \geq 2$*, treated as a unary relation belongs to* $\Gamma$*. If, for any* $f \in \mathsf{Pol}\ \Gamma$, $f|_B$ *is an essentially unary surjective operation, then* $\mathrm{CSP}(\Gamma)$ *is NP-complete.*

A conservative language $\Gamma$ satisfies the conditions of Proposition 1 for any $B \subseteq A$. Therefore, if $\mathrm{CSP}(\Gamma)$ is tractable then, for any 2-element subset $B = \{0, 1\} \subseteq A$, there exists an operation $f^B$ such that $f^B|_B$ is not an essentially unary surjective operation. Well known properties of Boolean operations [34], imply that $f^B$ can be chosen such that $f^B|_B$ is either a semilattice (that is conjunction or disjunction) operation, or the majority operation $(x \vee y) \wedge (y \vee z) \wedge (z \vee x)$, or the Mal'tsev operation $x - y + z \pmod 2$. The main result of this paper asserts that this property is also sufficient for the tractability of $\mathrm{CSP}(\Gamma)$.

**Theorem 4** *The problem* $\mathrm{CSP}(\Gamma)$ *for a conservative constraint language* $\Gamma$ *is tractable if and only if, for any 2-element subset* $B \subseteq A$ *(without loss of generality, we may assume* $B = \{0, 1\}$*), there exists an operation* $f^B \in \mathsf{Pol}\ \Gamma$ *such that* $f^B|_B$ *is either a semilattice operation, or the majority operation* $(x \vee y) \wedge (y \vee z) \wedge (z \vee x)$*, or the Mal'tsev operation* $x - y + z \pmod 2$*. Otherwise* $\mathrm{CSP}(\Gamma)$ *is NP-complete.*

In [10, 8] a possible criteria has been conjectured characterizing tractable problems $\mathrm{CSP}(\Gamma)$ for an arbitrary constraint language, $\Gamma$, over a finite set. It can be shown that, the criteria stated in Theorem 4 is equivalent to that from [10, 8] in the particular case of conservative constraint languages.

Theorem 4 completely solves the classification problem, and give a positive solution to the dichotomy problem. It also provides a polynomial time algorithm to recognize tractable conservative constraint satisfaction problems. Indeed, to verify the condition stated in the theorem for a conservative constraint language, $\Gamma$, one just have to calculate all binary and ternary polymorphisms of $\Gamma$, their number is bounded with a constant when the size of $A$ is fixed; and then, for each 2-element subset $B \subseteq A$, check all the computed polymorphisms looking for an operation $f$ such that $f|_B$ is of the specified types. Finally, in Section 5 we provide a polynomial time algorithm solving $\mathrm{CSP}(\Gamma)$ for a conservative $\Gamma$ whenever it is tractable.

We apply Theorem 4 to characterize those digraphs $H$ for which LIST $H$-COLORING problem is tractable. Such a characterization has been obtained in [18, 19] for undirected graphs. For an undirected graph $H$, LIST $H$-COLORING is tractable if and only if $H$ is *bi-arc*. Remarkably, bi-arc graphs can be described in terms of polymorphisms: a graph

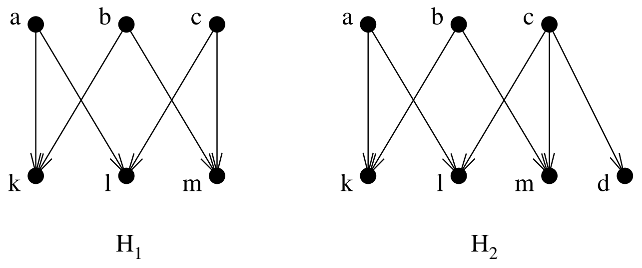


**Figure 1.**

$H$ is bi-arc if and only if $\varrho_H$ is symmetric and has a majority polymorphism [1, 19]. To our best knowledge, up to now, no progress has been made in studying LIST $H$-COLORING for directed graphs. Theorem 4 allows us to obtain a complete classification of the complexity of LIST $H$-COLORING problem in this case.

**Corollary 1** LIST $H$-COLORING *problem for a digraph $H$ is tractable if and only if $\varrho_H$ satisfies the conditions of Theorem 4. Otherwise it is NP-complete.*

The problem of describing the class of digraphs specified in Corollary 1 in graph-theoretic terms remains open.

**Example 5** Consider the LIST $H$-COLORING problems for the graphs shown in Fig. 1. It is not hard to see that $\mathsf{Pol}\ (\{\varrho_{H_1}\})$ contains a binary commutative operation $f$, such that the restriction of $f$ onto any 2-element set is a semilattice operation. For example, $f$ can be chosen to be defined by the equalities: $f(a,b) = b$, $f(b,c) = c$, $f(c,a) = a$, $f(k,l) = l$, $f(l,m) = m$, $f(m,k) = k$, and $f(x,y) = f(y,x) = x$ if $x \in \{a,b,c\}, y \in \{k,l,m\}$. By Corollary 1, LIST $H_1$-COLORING is tractable. In the next section we show that LIST $H_2$-COLORING is NP-complete. $\Box$

## 5. Algorithm

By contrast with the NP-completeness part of Theorem 4, the tractability part is very difficult. The solving algorithm is quite non-trivial, and the proofs of auxiliary results are elaborate and take many pages. We outline the basic principles of the algorithm and give its detailed description, but omit all the proofs.

### 5.1. Polymorphisms and coloured graphs

**Coloured edges.** We first demonstrate that every tractable conservative constraint language is related to a certain edge-coloured graph.

Let $\Gamma$ be a conservative constraint language over a finite set $A$, and $\Gamma$ satisfy the conditions of Theorem 4. We consider the graph $\mathcal{G}_\Gamma(A)$, an edge-coloured digraph with vertex set $A$. An edge $(a,b)$ is present and is coloured *red* if there is an operation $f \in \mathsf{Pol}\ \Gamma$ such that $f|_{\{a,b\}}$ is a semilattice operation with $f(a,b) = f(b,a) = f(b,b) = b$, $f(a,a) = a$. Edges $(a,b),(b,a)$ are present and are coloured *yellow* if neither $(a,b)$ nor $(b,a)$ can be coloured red and there is $f \in \mathsf{Pol}\ \Gamma$ such that $f|_{\{a,b\}}$ is a majority operation. Edges $(a,b),(b,a)$ are present and are coloured *blue* if none of them can be coloured red or yellow, and there is $f \in \mathsf{Pol}\ \Gamma$ such that $f|_{\{a,b\}}$ is an affine operation. Thus, for each pair $a,b \in A$, at least one of $(a,b)$, $(b,a)$ is an edge of $\mathcal{G}_\Gamma(A)$; if $(a,b)$ is a yellow or blue then $(b,a)$ is also an edge of the same colour; while if $(a,b)$ is red then the edge $(b,a)$ may do not exist. We will usually omit the subscript $\Gamma$ when the constraint language will be clear from the context. For a subset $B \subseteq A$, the graph $\mathcal{G}(B)$ is the subgraph of $\mathcal{G}(A)$ spanned on $B$.

In fact, only few operations are required.

**Proposition 2** *There are operations $f(x,y), g(x,y,z)$, $h(x,y,z) \in \mathsf{Pol}\ \Gamma$ such that, for every 2-element subset $\{a,b\}$,*

- *$f|_{\{a,b\}}$ is a semilattice operation whenever $(a,b)$ or $(b,a)$ is a red edge, and $f|_{\{a,b\}}(x,y) = x$ otherwise;*
- *$g|_{\{a,b\}}$ is a majority operation if $(a,b),(b,a)$ are yellow, $g|_{\{a,b\}}(x,y,z) = x$ if $(a,b)$ is blue, and $g|_{\{a,b\}}(x,y,z) = f|_{\{a,b\}}(f|_{\{a,b\}}(x,y),z)$ if $(a,b)$ or $(b,a)$ is red;*
- *$h|_{\{a,b\}}$ is the affine operation if $(a,b)$ is blue, $h|_{\{a,b\}}(x,y,z) = x$ if $(a,b)$ is yellow, and $h|_{\{a,b\}}(x,y,z) = f|_{\{a,b\}}(f|_{\{a,b\}}(x,y),z)$ if $(a,b)$ or $(b,a)$ is red.*

We replace $\Gamma$ with a larger constraint language $\Gamma' = \mathsf{Inv}\ (\{f,g,h\})$. As the operations $f,g,h$ witness, $\Gamma'$ still satisfies the conditions of Theorem 4. Moreover, $\Gamma'$ contains all relations that can appear when processing a problem instance from CSP($\Gamma$) or CSP($\Gamma'$). The operation $f$ uniquely determines the orientation of red edges, therefore, if $(a,b)$ is a red edge of $\mathcal{G}_{\Gamma'}(A)$ then $(b,a)$ is not present.

**Example 6** We now are able to show that the problem LIST $H_2$-COLORING, where $H_2$ is the graph from Fig. 1, is NP-complete. If this problem were tractable then there would exist polymorphisms $f,g,h$ of $\varrho = \varrho_{H_2}$ specified in Proposition 2. We show that the restrictions of these operations onto $B = \{d,k\}$ cannot be a semilattice, a majority, or an affine operation. Notice first that if $f|_B$ is a semilattice operation with $f(k,d) = f(d,k) = d$ [$f(k,d) = f(d,k) = k$] then so are $f|_{\{a,c\}}, f|_{\{b,c\}}, f|_{\{k,l\}}, f|_{\{k,m\}}$ and $f(a,c) = c, f(b,c) = c, f(k,l) = l, f(k,m) = m$ [$f(a,c) = a, f(b,c) = b, f(k,l) = k, f(k,m) = k$]. Indeed, for example, since $f$ is a polymorphism of $\varrho$ and

$f(k,d) = f(d,k) = d$,

$$f\left(\begin{pmatrix}a\\k\end{pmatrix},\begin{pmatrix}c\\d\end{pmatrix}\right),\quad f\left(\begin{pmatrix}c\\d\end{pmatrix},\begin{pmatrix}a\\k\end{pmatrix}\right)$$

must be equal to $\begin{pmatrix}d\\c\end{pmatrix}$, which means $f(a,c) = f(c,a) = c$. Analogously, if $g|_B$ is a majority operation or $h|_B$ is an affine operation then so are $g|_{\{a,c\}}, g|_{\{b,c\}}, g|_{\{k,l\}}, g|_{\{k,m\}}$ or $h|_{\{a,c\}}, h|_{\{b,c\}}, h|_{\{k,l\}}, h|_{\{k,m\}}$.

Suppose that $f|_B$ is a semilattice operation with $f(d,k) = d$. Then we have

$$f\left(\begin{pmatrix}a\\k\end{pmatrix},\begin{pmatrix}b\\m\end{pmatrix}\right) = \begin{pmatrix}e\\m\end{pmatrix},\ f\left(\begin{pmatrix}a\\l\end{pmatrix},\begin{pmatrix}b\\k\end{pmatrix}\right) = \begin{pmatrix}e\\l\end{pmatrix}$$

where $e = f(a,b)$. The first equality implies $f(a,b) = b$ while the second implies $f(a,b) = a$, a contradiction.

Then, let $g|_B$ is a majority operation, and $e$ denote $g(a,b,c)$. Then the pairs

$$\begin{aligned} g\left(\begin{pmatrix}a\\k\end{pmatrix},\begin{pmatrix}b\\k\end{pmatrix},\begin{pmatrix}c\\d\end{pmatrix}\right) &= \begin{pmatrix}e\\k\end{pmatrix},\\ g\left(\begin{pmatrix}a\\l\end{pmatrix},\begin{pmatrix}b\\k\end{pmatrix},\begin{pmatrix}c\\l\end{pmatrix}\right) &= \begin{pmatrix}e\\l\end{pmatrix},\\ g\left(\begin{pmatrix}a\\k\end{pmatrix},\begin{pmatrix}b\\m\end{pmatrix},\begin{pmatrix}c\\m\end{pmatrix}\right) &= \begin{pmatrix}e\\m\end{pmatrix} \end{aligned}$$

belong to $\varrho$, and hence, $e \in \{a,b\}$ follows from the first equality, $e \in \{a,c\}$, $e \in \{b,c\}$ from the second and the third equalities respectively, a contradiction.

Finally, it can be shown in a similar way that $h$ also cannot be properly defined. □

**Known cases.** If all edges of $\mathcal{G}(A)$ are coloured with the same colour, the tractability of CSP$(\Gamma)$ follows from the earlier results.

CASE 1. All the edges of $\mathcal{G}(A)$ are blue.
In this case $h$ is the affine operation on every two-element set, hence it satisfies the conditions $h(a,b,b) = h(b,b,a) = a$ for any $a,b \in A$. Thus, $h$ is a Mal'tsev operation. The tractability of CSP$(\Gamma)$ follows from the results of [3].

CASE 2. All edges of $\mathcal{G}(A)$ are yellow.
In this case $g$ is the majority operation on every two-element set, hence it satisfies the conditions $g(a,a,b) = g(a,b,a) = g(b,a,a) = a$ for any $a,b \in A$. Therefore $g$ is a majority operation on $A$, and the tractability of CSP$(\Gamma)$ follows from the results of [26, 20].

CASE 3. All edges of $\mathcal{G}(A)$ are red.
In this case $g,h$ can be neglected as they do not affect the set $\mathsf{Pol}\ \Gamma'$. The operation $f$ is idempotent, commutative, and satisfies the condition $f(a,b) \in \{a,b\}$. Therefore, $f$ is a commutative conservative operation. As is proved in [6], CSP$(\Gamma')$ is tractable.

**Connectedness.** We use several forms of connectedness in $\mathcal{G}(A)$ defined as existing a path (directed or not) consisting of edges of some restricted set of colours. Vertices $a,b$ are said to be *r-* [*r+y-*, *r+b-*] *connected* if there is an undirected path from $a$ to $b$ consisting of red [red and yellow, red and blue] edges. They are said to be *strongly r-* [*r+y-*, *r+b-*] *connected* if there is a directed path from $a$ to $b$ consisting of red [red and yellow, red and blue] edges. We then define *r-connected components*, and *strongly r-* [*r+y-*, *r+b-*] *connected components* in a natural way.

For strongly r-connected components $C,D$ of $\mathcal{G}(A)$ we write $C <_r D$ if there are $a \in C$, $b \in D$ such that $(a,b)$ is a red edge. The transitive closure $\prec_r$ of $<_r$ is a partial order on the set of strongly r-connected components. A component maximal with respect to this order is said to be *maximal strongly r-connected component*. In a similar way we define the partial orders $\prec_b, \prec_y$ on the sets of strongly r+b- and r+y-connected components, and also *maximal strongly r+b-connected components* and *maximal strongly r+y-connected components*. Note that we use the same symbols for any subset $B \subseteq A$. It will always be clear from the context which graph is meant.

## 5.2. Partial solutions and bounded relational width

One of the most frequently used propagation algorithms for constraint satisfaction problems tries to guess a solution to the whole problem by findind 'partial' solutions for certain sets of variables. In some cases this strategy is successful.

We use $\underline{n}$ to denote the set $\{1,\ldots,n\}$. For an $n$-ary relation $\varrho$, $\mathbf{a} \in \varrho$, and $I = \{i_1,\ldots,i_k\} \subseteq \underline{n}$, by $\mathrm{pr}_I\mathbf{a}$ we denote the tuple $(\mathbf{a}[i_1],\ldots,\mathbf{a}[i_k])$, and by $\mathrm{pr}_I\varrho$ the set $\{\mathrm{pr}_I\mathbf{b} \mid \mathbf{b} \in \varrho\}$. We will write $\mathrm{pr}_i\varrho$ instead of $\mathrm{pr}_{\{i\}}$. Clearly, $\varrho \subseteq \mathrm{pr}_1\varrho \times \ldots \times \mathrm{pr}_n\varrho$; we say that $\varrho$ is a *subdirect product* of $A_1,\ldots,A_n$ if $\mathrm{pr}_i\varrho = A_i$ for any $i$.

If $\langle s,\varrho\rangle$ is a constraint in a CSP, $s = (v_1,\ldots,v_n)$, $\varrho$ is an $n$-ary relation, and $W = (v_{i_1},\ldots,v_{i_k}) \subseteq s$, then it will be convenient to use $\mathrm{pr}_W\varrho$ instead of $\mathrm{pr}_{\{i_1,\ldots,i_k\}}\varrho$.

Let $\mathcal{P} = (V;A;\mathcal{C})$ be an instance of the CSP, and $W \subseteq V$. By $\mathcal{P}_W$ we denote the *restricted problem*, that is the problem instance defined as $(W;A;\mathcal{C}')$ where, for every $\langle s,\varrho\rangle \in \mathcal{C}$, there is $\langle s',\varrho'\rangle \in \mathcal{C}'$ with $s' = s \cap W$, and $\varrho' = \mathrm{pr}_{s'}\varrho$. Every solution of $\mathcal{P}_W$ is said to be a *partial solution* of $\mathcal{P}$ on $W$. We use $\mathcal{S}_W$ to denote the set of all partial solutions on $W$. Notice that $\mathcal{S}_W$ can be treated as a $|W|$-ary relation.

The problem instance $\mathcal{P}$ is said to be *$k$-minimal* if, for any $k$-element subset $W$ of $V$, there is a constraint $\langle s,\varrho\rangle \in \mathcal{C}$ such that $W \subseteq s$, and for any $\langle s,\varrho\rangle \in \mathcal{C}$, we have $\mathrm{pr}_{W\cap s}\varrho = \mathrm{pr}_{W\cap s}\mathcal{S}_W$.

Any problem instance $\mathcal{P}$ can be transformed to an equivalent $k$-minimal instance $\mathcal{P}'$. To do this we have just to repeat the following procedure until the obtained problem instance coincides with the previous one: for each $k$-element set $W \subseteq V$, calculate the relation $\mathcal{S}_W$, and then, for each constraint $\langle s, \varrho \rangle \in \mathcal{C}$, remove from $\varrho$ every tuple $\mathbf{a}$ such that $\mathrm{pr}_{s \cap W} \mathbf{a} \notin \mathrm{pr}_{s \cap W} \mathcal{S}_W$. The obtained instance $\mathcal{P}'$ is $k$-minimal and the time complexity of this procedure is $O(m^2 n^k)$ where $n$ is the number of variables and $m$ is the total number of tuples in the constraint relations. The obtained instance is referred to as *associated* with $\mathcal{P}$. As is easily seen the solution set of the associated instance is equal to that of the original instance. The algorithm transforming a problem to an equivalent $k$-minimal problem will be called $k$-MINIMALITY.

Constraint satisfaction problems for which the $k$-minimality implies existence of a solution (provided that there are no empty constraint relations) are said to be of the *relational width* $k$. Problems of this and similar types have been widely studied in computer science, see, for example, [12, 16, 20, 25].

A solution to a CSP of the relational width $k$ can be found by a simple algorithm:

**Algorithm** $k$-WIDTH

INPUT: Problem instance $\mathcal{P} = (V; A; \mathcal{C})$ where $A = \{a_1, \ldots, a_k\}$ and $V$ is ordered in some way.

OUTPUT: a solution to $\mathcal{P}$ if there is one, 'NO' otherwise.

*Step 1.* **Invoke** $k$-MINIMALITY($\mathcal{P}$)
*Step 2.* **if** $V = \varnothing$ **then output** a solution
**otherwise set** $v$ to be the fist variable from $V$
*Step 3.* **for** $i = 1$ **to** $k$ **do**
*Step 3.1* **set** $\mathcal{P}_i = (V - \{v\}, A, \mathcal{C}_i)$ to be such that, for any $\langle s, \varrho \rangle \in \mathcal{C}$, there is $\langle s, \varrho \rangle \in \mathcal{C}_i$ if $v \notin s$ and $\langle s - \{v\}, \varrho' \rangle \in \mathcal{C}_i$ where $\varrho' = \{\mathrm{pr}_{s-\{v\}} \mathbf{a} \mid \mathbf{a} \in \varrho, \mathbf{a}[v] = a_i\}$
*Step 3.2* **invoke** $k$-MINIMALITY($\mathcal{P}_i$)
*Step 3.3* **if** the obtained $k$-minimal problem contains no empty constraint relations **then** $k$-WIDTH($\mathcal{P}_i$)
**endfor**

### 5.3. Auxiliary constructions

The main idea of our algorithm is to split a problem into several smaller subproblems, usually selected in a sophisticated way, so that the existence of a solution to each subproblem ensures the existence of a solution to the whole problem, and if some subproblem has no solution then the problem can be tightened by excluding tuples from the contraint relations. Let $\mathcal{P} = (V; A; \mathcal{C})$ be a problem instance from CSP$(\Gamma)$. Without loss of generality $\mathcal{P}$ can be assumed to be 3-minimal. We use $\mathcal{G}_v$ to denote the graph $\mathcal{G}(\mathcal{S}_v)$, $v \in V$. We distinguish three types of subproblems depending on the structure of the graphs $\mathcal{G}_v$.

**Maximal strongly r+b-connected components** We use this type of subproblems when there is $v \in V$ such that $\mathcal{G}_v$ is not strongly r+b-connected.

If $\varrho \in \Gamma'$ is a subdirect product of $A_1, \ldots, A_n$, $j \in \underline{n}$, and $B$ is a strongly r+b-connected component of $\mathcal{G}(A_i)$, then we set

$$J_\varrho(\mathrm{rb}, j, B) = \{i \in \underline{n} \mid \text{for any } \mathbf{a}, \mathbf{b} \in \varrho \text{, such that } \mathbf{a}[j] \in B \text{, the element } \mathbf{b}[j] \text{ belongs to } B \text{ if and only if } \mathbf{a}[i], \mathbf{b}[i] \text{ are strongly r+b-connected in } \mathcal{G}(A_i)\}$$

For any $i \in J_\varrho(\mathrm{rb}, j, B)$, we use $B_i$ to denote the strongly r+b-connected component of $\mathcal{G}(A_i)$ such that, for any $\mathbf{a} \in \varrho$, $\mathbf{a}[i] \in B_i$ whenever $\mathbf{a}[j] \in B$; in particular, $B_j = B$.

**Lemma 1** *If $B$ is a maximal strongly r+b-connected component of $\mathcal{G}(A_j)$ then, for any $i \in J_\varrho(\mathrm{rb}, j, B)$, $B_i$ is a maximal strongly r+b-connected component of $\mathcal{G}(A_i)$.*

We then extend this notation for the problem instance $\mathcal{P}$. Take a variable $v \in V$, a strongly connected component $B$ of $\mathcal{G}_v$, and set $J_1$ to be the union of $J_\varrho(\mathrm{rb}, v, B)$ where $\varrho$ runs over all constraint relations whose scope contains $v$. Note that, since $\mathcal{P}$ is 3-minimal, if for a certain $w \in J_1$, $v, w$ are contained in the scopes of constraints $\langle s_1, \varrho_1 \rangle, \langle s_2, \varrho_2 \rangle$ then, for any $\mathbf{a} \in \varrho_1$, $\mathbf{b} \in \varrho_2$ such that $\mathbf{a}[v], \mathbf{b}[v] \in B$, the elements $\mathbf{a}[w], \mathbf{b}[w]$ are also strongly r+b-connected. This allows us to define $B_w$ for every $w \in J_1$ in the same way as for relations. Suppose that $J_k$ is already defined. Then we set $J_{k+1}$ to be the union of all sets of the form $J_\varrho(\mathrm{rb}, w, B_w)$ where $w \in J_k$ and $\langle s, \varrho \rangle$ is a constraint such that $w \in s$. Finally, $J(\mathrm{rb}, v, B)$ is defined to be the largest set among the sets $J_k$. Note that if $\varphi$ is a solution to $\mathcal{P}$ such that $\varphi(v) \in B$ then $\varphi(w) \in B_w$ for any $w \in J(\mathrm{rb}, v, B)$.

We now define the subproblem corresponding to $J(\mathrm{rb}, v, B)$. The problem $\mathcal{P}_{\mathrm{rb},v,B}$ is defined to be the problem instance $(W; A; \mathcal{C}')$ where $W = J(\mathrm{rb}, v, B)$ and, for every $C = \langle s, \varrho \rangle \in \mathcal{C}$, there is $C' = \langle s \cap W; \varrho' \rangle \in \mathcal{C}$ such that

$$\varrho' = \{\mathrm{pr}_{s \cap W} \mathbf{a} \mid \mathbf{a} \in \varrho,\ \mathbf{a}[w] \in B_w \text{ for any } w \in s \cap W\}.$$

**Lemma 2** *If, for any $v \in V$, and any maximal strongly r+b-connected component $B$ of $\mathcal{G}_v$, the restricted problem $\mathcal{P}_{\mathrm{rb},v,B}$ has a solution, then $\mathcal{P}$ also has a solution.*

Moreover, if the conditions of Lemma 2 hold then a solution to $\mathcal{P}$ can be found in polynomial time by the algorithm 3-WIDTH.

Thus if all the subproblems have solutions then Lemma 2 allows one to solve the problem $\mathcal{P}$. Otherwise if $\mathcal{P}_{\mathrm{rb},v,B}$ has no solution then $\mathcal{P}$ can be tightened by removing from every $\varrho$, $\langle s, \varrho \rangle \in \mathcal{C}$, such that $w \in s \cap J(\mathrm{rb}, v, B)$, every tuple $\mathbf{a}$ with $\mathbf{a}[w] \in B_w$.

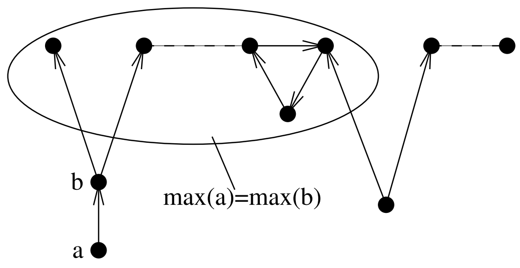


**Figure 2. Maximal elements. Red edges are drawn solid, yellow edges are dotted, blue edges are omitted**

**Maximal strongly r-connected components.** This type of subproblems is used if every $\mathcal{G}_v$ is strongly r+b-connected, but, for a certain $v \in V$, there is an r-connected component of $\mathcal{G}_v$ which is not strongly r-connected.

Let $\varrho \in \Gamma'$ be a subdirect product of $A_1, \ldots, A_n$. For each $A_i$, we use $A'_i$ to denote the union of all maximal strongly r-connected components.

**Lemma 3** *The relation* $\varrho' = \varrho \cap (A'_1 \times \ldots \times A'_n)$ *is a subdirect product of* $A'_1, \ldots, A'_n$.

Elements from $A'_i$ are said to be *maximal*. For any $a \in A_i$, we define the set of corresponding maximal elements as follows:

$$\max(a) = \{b \in A'_i \mid \text{there is } c \in A'_i \text{ such that } a \text{ is strongly r-connected with } c \text{ and } c, b \text{ are strongly r+y-connected in } \mathcal{G}(A'_i)\}$$

(see Fig. 2). Analogously, for any $\mathbf{a} \in \varrho$, we set

$$\max(\mathbf{a}) = \{\mathbf{b} \in \varrho \mid \mathbf{b}[i] \in \max(\mathbf{a}[i]) \text{ for any } i \in \underline{n}\}.$$

We introduce another order-like relation on $A_i$, $i \in \underline{n}$. For $a, b \in A_i$, we write $a \sqsubseteq b$ iff $\max(a) \subseteq \max(b)$. Clearly, $\sqsubseteq$ is a quasi-order. This quasi-order gives rise in the standard way, to the equivalence relation $\sqsubseteq \cap \sqsubseteq^{-1}$; the classes of this equivalence relation are said to be the $\sqsubseteq$-classes. They consist of elements $a, b \in A_i$ such that $a \sqsubseteq b$ and $b \sqsubseteq a$. Moreover, $\sqsubseteq$ induces an partial order on the collection of $\sqsubseteq$-classes. We will concern with maximal $\sqsubseteq$-classes.

For a maximal $\sqsubseteq$-class we use a construction analogous to that for a strongly r+b-connected component. If $j \in \underline{n}$ and $B$ is a maximal $\sqsubseteq$-class of $\mathcal{G}(A_j)$ then

$$J_\varrho(\mathrm{m}, j, B) = \{i \in \underline{n} \mid \text{for any } \mathbf{a}, \mathbf{b} \in \varrho, \text{ such that } \mathbf{a}[j] \in B, \text{ the element } \mathbf{b}[j] \text{ belongs to } B \text{ if and only if } \mathbf{a}[i], \mathbf{b}[i] \text{ are in the same maximal } \sqsubseteq\text{-class of } \mathcal{G}(A_i)\}.$$

Sets of the form $J(\mathrm{m}, v, B)$ for the problem instance $\mathcal{P}$ can be defined similarly to those in the previous case. The union of all maximal $\sqsubseteq$-classes of $\mathcal{G}_v$ we denote by $\mathcal{S}_v^0$.

In this case the subproblems of $\mathcal{P}$ correspond to the sets of the form $J(\mathrm{m}, v, B)$ and are defined as follows. By $\mathcal{P}^+_{v,B}$ we denote the constraint satisfaction problem $(W; A; \mathcal{C}')$ where $W = J(\mathrm{m}, v, B)$ and, for each constraint $\langle s, \varrho\rangle \in \mathcal{C}$, we include $C' = \langle s \cap W, \varrho'\rangle$ in $\mathcal{C}'$, with

$$\varrho' = \{\mathrm{pr}_{s\cap W}\mathbf{a} \mid \mathbf{a} \in \varrho,\ \mathbf{a}[w] \in \mathbf{a}[w] \in \mathbf{C}_w \text{ for } w \in s \cap W\}.$$

where $C_w = \{a \in \mathcal{S}_w \mid B_w \sqsubseteq a\}$.

**Lemma 4** *If every problem of the form* $\mathcal{P}^+_{v,B}$ *has a solution* $\varphi$ *such that every* $\varphi(w)$, $w \in J(\mathrm{m}, v, B)$, *is maximal, and* $\mathcal{P}$ *has a solution, then there is a solution* $\psi$ *to* $\mathcal{P}$ *such that* $\psi(w)$ *is maximal for any* $w \in V$.

Having a formal description of the algorithm we shall be able to show that if a problem of the form $\mathcal{P}^+_{v,B}$ has no solution of the required type, then $\mathcal{P}$ can be tightened. Otherwise, Lemma 4 allows us to reduce $\mathcal{P}$ to the smaller problems $\mathcal{P}^+_{v,B}$, and the problem $\mathcal{P}^{\max} = (V, C, \mathcal{C}')$ in which every constraint relation $\langle s, \varrho\rangle$ is replaced with $\langle s, \varrho'\rangle$, $\varrho' = \{\mathbf{a} \in \varrho \mid \text{for any } v \in s, \mathbf{a}[v] \text{ is maximal }\}$.

However, Lemma 4 cannot help in the case when, for any $v \in V$, $\mathcal{S}_v = \mathcal{S}_v^0$. In this case we need another method.

Denote the set of maximal elements in $\mathcal{S}_v$ by $\mathcal{S}_v^{\max}$. By definition, $\mathcal{S}_v^{\max}$ is strongly r+y-connected. First suppose that, for any $v \in V$, it is also strongly r+b-connected. Then $\mathcal{P}^{\max}$ is 3-minimal and satisfies the conditions of Lemma 8 and can be solved by the algorithm 3-WIDTH.

Then, suppose that there is $v \in V$ such that $\mathcal{S}_v^{\max}$ is not strongly r+b-connected (and, therefore, is not r+b-connected). We consider the sets of the form $J_{\max}(\mathrm{rb}, v, B)$ defined as $J(\mathrm{rb}, v, B)$ for the problem $\mathcal{P}^{\max}$, where $B$ is a strongly r+b-connected component of $\mathcal{S}_v^{\max}$. As usual, having fixed $v$ and $B$ we get a strongly r+b-connected component $B_w$ for each $w \in J_{\max}(\mathrm{rb}, v, B)$.

We define the subproblems of $\mathcal{P}$ corresponding to the sets of the form $J_{\max}(\mathrm{rb}, v, B)$ as follows. By $\mathcal{P}^{\max}_{v,B}$ we denote the constraint satisfaction problem $(W; A; \mathcal{C}')$ where $W = J_{\max}(\mathrm{rb}, v, B)$ and, for each constraint $\langle s, \varrho\rangle \in \mathcal{C}$, we include $C' = \langle s \cap W, \varrho'\rangle$ in $\mathcal{C}'$, with

$$\varrho' = \{\mathrm{pr}_{s\cap W}\mathbf{a} \mid \mathbf{a} \in \varrho,\ \mathbf{a}[w] \in \mathcal{S}_{w,B} \text{ for } w \in s \cap W\},$$

where $\mathcal{S}_{w,B}$ is the set of all $a \in \mathcal{S}_w$ such that for every $b \in \max(a)$ and such that $a \prec b$ belongs to $B_w$. Note that $\max(\mathcal{S}_{w,B}) = B_w$.

**Lemma 5** *Suppose that*

*(1)* $\mathcal{P}$ *is 3-minimal;*

*(2) for any* $v \in V$, $\mathcal{S}_v = \mathcal{S}_v^0$*;*

*(3) there is* $v \in V$ *such that* $\mathcal{S}_v^{\max}$ *is not strongly r+b-connected;*

*(4) for any* $v \in V$ *and strongly r+b-connected components* $B$ *of* $\mathcal{S}_v^{\max}$, *the problem* $\mathcal{P}^{\max}_{v,B}$ *has a solution* $\psi_{v,B}$ *such that*

$\psi_{v,B}(w) \in B_w$ *for any* $w \in J_{\max}(\mathrm{rb}, v, B)$.

*Then* $\mathcal{P}$ *has a solution* $\varphi$ *such that* $\varphi(v)$ *belongs to a maximal strongly r-connected component for any* $v \in V$.

**Strongly r+y-connected components.** We use this type of subproblems if, for any $v \in V$, the graph $\mathcal{G}_v$ is strongly r+b-connected and every its r-connected component is strongly r-connected.

In this case any subdirect product $\varrho \in \Gamma'$ of $A_1, \ldots, A_n$ satisfies a very strong condition. We define sets $J_\varrho(\mathrm{ry}, j, B)$, $j \in \underline{n}$, $B$ is an r+y-connected component of $A_j$, in the same way as for strongly r+b-connected components:

$J_\varrho(\mathrm{ry}, j, B) = \{i \in \underline{n} \mid$ for any $\mathbf{a}, \mathbf{b} \in \varrho$, such that $\mathbf{a}[j] \in B$, the element $\mathbf{b}[j]$ belongs to $B$ if and only if $\mathbf{a}[i], \mathbf{b}[i]$ are r+y-connected in $\mathcal{G}(A_i)$

The relation $\varrho$ is said to be ry-*rectangular* if for any $\mathbf{a} \in \varrho$, any $j \in \underline{n}$, and any $\mathbf{b} \in \mathrm{pr}_I \varrho$, where $I = J_\varrho(\mathrm{ry}, j, B)$, $B$ is the r+y-connected component containing $\mathbf{a}[j]$, and $\mathbf{b}[j] \in B$, the tuple $\mathbf{c}$ belongs to $\varrho$:

$$\mathbf{c}[i] = \begin{cases} \mathbf{b}[i], & \text{if } i \in J_\varrho(\mathrm{ry}, j, B), \\ \mathbf{a}[i], & \text{otherwise.} \end{cases}$$

**Lemma 6** *Let* $\varrho \in \Gamma'$ *be a subdirect product of* $A_1, \ldots, A_n$ *such that, for any* $i \in \underline{n}$, $\mathcal{G}(A_i)$ *is strongly r+b-connected and every its r-connected component is strongly r-connected. Then* $\varrho$ *is* ry-*rectangular.*

Lemma 6 reduces the problem $\mathcal{P}$ to a collection of subproblems $\mathcal{P}_{\mathrm{ry},v,B}$, defined analogously to those for strongly r+b-connected components, and a *skeleton problem*. The latter problem is defined as follows. We may assume that each problem of the form $\mathcal{P}_{\mathrm{ry},v,B}$ has a solution $\mathcal{S}_{v,B}$ found by the algorithm. Moreover, if $w \in J(\mathrm{ry}, v, B)$, then $J(\mathrm{ry}, v, B) = J(\mathrm{ry}, w, B_w)$ where $B_w$ is the r+y-connected component corresponding to $B$, $\mathcal{P}_{\mathrm{ry},v,B} = \mathcal{P}_{\mathrm{ry},w,B_w}$, and $\mathcal{S}_{v,B} = \mathcal{S}_{w,B_w}$. Then the skeleton problem is the problem $\mathcal{P}^s = (V; A; \mathcal{C}^s)$ obtained by removing all tuples from the constraint relations but those consisting of the solutions $\mathcal{S}_{v,B}$: for every $C = \langle s, \varrho \rangle \in \mathcal{C}$ there is $C^s = \langle s, \varrho^s \rangle \in \mathcal{C}^s$ where $\mathbf{a} \in \varrho^s$ if and only if $\mathbf{a} \in \varrho$ and, for any $v \in s$, any r+y-connected component $B$, we have $\mathrm{pr}_{s \cap J(\mathrm{ry},v,B)} \mathbf{a} = \mathrm{pr}_{s \cap J(\mathrm{ry},v,B)} \mathcal{S}_{v,B}$.

**Lemma 7** *If, for any* $v \in V$ *and any r+y-connected component,* $B$*, of* $\mathcal{G}_v$*, the problem* $\mathcal{P}_{\mathrm{ry},v,B}$ *has a solution, then the skeleton problem has a solution if and only if* $\mathcal{P}$ *has a solution. Moreover, a solution to* $\mathcal{P}$ *can be chosen to be a solution of the skeleton problem.*

Note that if $\mathcal{S}^s_v$ denotes the partial solution to the skeleton problem on $v \in V$, then every edge of $\mathcal{G}(\mathcal{S}_v)$ is blue. Therefore the skeleton problem can be solved by the algorithm MALTSEV described in [3]. If one of the subproblems $\mathcal{P}_{\mathrm{ry},v,B}$ has no solution then $\mathcal{P}$ can be tightened by removing from every constraint relation $\varrho$, $\langle s, \varrho \rangle \in \mathcal{C}$, every tuple $\mathbf{a}$ such that $\mathbf{a}[w] \notin B_w$ for $w \in s \cap J(\mathrm{ry}, v, B)$.

In the only case that is not yet considered, for any $v \in V$, $\mathcal{G}_v$ is strongly r+b- and strongly r+y-connected.

**Lemma 8** *If, for any* $v \in V$, $\mathcal{G}_v$ *is strongly r+b- and strongly r+y-connected, then* $\mathcal{P}$ *is of relational width 3.*

Now we in are a position to give a formal description of the algorithm.

### 5.4. Algorithm

**Algorithm** CONSERV

INPUT: Problem instance $\mathcal{P} = (V; A; \mathcal{C})$.

OUTPUT: a solution to $\mathcal{P}$ and the tightened problem instance if $\mathcal{P}$ has a solution, $(\varnothing, \varnothing)$ otherwise.

*Step 1.* **Invoke** 3-MINIMALITY($\mathcal{P}$)
*Step 2.* **Find** (strongly) r+y-, r+b-, r- connected components of $\mathcal{S}_v$, $v \in V$.
*Step 3.* **ifcase:**
  *there exists* $v \in V$ *such that* $\mathcal{G}_v$ *is not strongly r+b-connected*:
*Step 3.1.* **do**
*Step 3.1.1.* **for each** $v \in V$, **each** maximal strongly r+b-connected component $B \subseteq \mathcal{S}_v$, **do**
*Step 3.1.1.1.* $(\mathcal{S}_{v,B}, \mathcal{P}'_{\mathrm{rb},v,B})$ :=CONSERV($\mathcal{P}_{\mathrm{rb},v,B}$)
*Step 3.1.1.2.* for every constraint $C = \langle s, \varrho \rangle$ of $\mathcal{P}$, **remove** from $\varrho$ every tuple $\mathbf{a}$ such that there is $v \in s$, $B \subseteq \mathcal{S}_v$, $W = J(\mathrm{r+b}, v, B)$ with $\mathrm{pr}_{W \cap s} \mathbf{a} \notin \varrho'$ where $C' = \langle W \cap s, \varrho' \rangle$ is the corresponding constraint of $\mathcal{P}'_{\mathrm{rb},v,B}$
  **endfor**
*Step 3.1.2.* **If** $\mathcal{P}$ changed **then return**(CONSERV($\mathcal{P}$))
*Step 3.1.3.* **Otherwise return**(3-WIDTH($\mathcal{P}$))
  **enddo**
  *there exists* $v \in V$ *and an r-connected component* $C$ *of* $\mathcal{G}_v$ *such that* $C$ *is not strongly r-connected and, for any* $w \in V$, $\mathcal{S}_w = \mathcal{S}^0_w$:
*Step 3.2.* **do**
*Step 3.2.1.* **If** for any $v \in V$, $\mathcal{S}^{\max}_v$ is strongly r+b-connected, **then return**(3-WIDTH($\mathcal{P}^{\max}$))
*Step 3.2.2.* **Otherwise for each** $v \in V$, **each** strongly r+b-connected component $B \subseteq \mathcal{S}^{\max}_v$, **do**
*Step 3.2.2.1.* $(\mathcal{S}_{v,B}, \mathcal{P}'_{v,B})$ :=CONSERVATIVE($\mathcal{P}^{\max}_{v,B}$)
*Step 3.2.2.2.* for every constraint $C = \langle s, \varrho \rangle$ of $\mathcal{P}$, **remove** from $\varrho$ every tuple $\mathbf{a}$ such that there is $v \in s$, $B \subseteq \mathcal{S}_v$, $W = J_{\max}(\mathrm{rb}, v, B)$ with $\mathrm{pr}_{W \cap s} \mathbf{a} \notin \varrho'$ and $\mathbf{a}[v] \in B$ where $C' = \langle W \cap s, \varrho' \rangle$ is the corresponding constraint of $\mathcal{P}'_{v,B}$

**endfor**
*Step 3.2.3.* **if** $\mathcal{P}$ changed **then return**(CONSERV($\mathcal{P}$))
*Step 3.2.4.* **otherwise return**(3-WIDTH($\mathcal{P}^{\max}$))
**enddo**
*there exists $v \in V$ and an r-connected component $C$ of $\mathcal{G}_v$ such that $C$ is not strongly r-connected and $w \in V$ such that $\mathcal{S}_w \neq \mathcal{S}_w^0$*:
*Step 3.3.* **do**
*Step 3.3.1.* **if** for any $v \in V$, $\mathcal{S}_v = \mathcal{S}_v^0$ **then return**(3-WIDTH($\mathcal{P}^{\max}$)
*Step 3.3.2.* **for each** $v \in V$, **each** maximal $\sqsubseteq$-class $B \subseteq \mathcal{S}_v$, **do**
*Step 3.3.2.1.* $(\mathcal{S}_{v,B}, \mathcal{P}'_{v,B})$ :=CONSERV($\mathcal{P}^{+}_{v,B}$)
*Step 3.3.2.2.* for every constraint $C = \langle s, \varrho\rangle$ of $\mathcal{P}$, **remove** from $\varrho$ every tuple $\mathbf{a}$ such that there is $v \in s$, $B \subseteq \mathcal{S}_v$, $W = J(\mathrm{m}, v, B)$ with $\mathrm{pr}_{W\cap s}\mathbf{a} \notin \varrho'$ where $C' = \langle W \cap s, \varrho'\rangle$ is the corresponding constraint of $\mathcal{P}'_{v,B}$
**endfor**
*Step 3.3.3.* **If** $\mathcal{P}$ changed **then return**(CONSERV($\mathcal{P}$))
*Step 3.3.4.* **Otherwise return**(CONSERV($\mathcal{P}^{\max}$)
**enddo**
*there exists $v \in V$ such that $\mathcal{G}_v$ is not r+y-connected*:
*Step 3.4.* **do**
*Step 3.4.1.* **for each** $v \in V$, each r+y-connected component $B$ **do**
*Step 3.4.1.1.* $(\mathcal{S}_{v,B}, \mathcal{P}'_{\mathrm{ry},v,B})$ :=CONSERV($\mathcal{P}_{\mathrm{ry},v,B}$)
*Step 3.4.1.2.* for every constraint $C = \langle s, \varrho\rangle$ of $\mathcal{P}$, **remove** from $\varrho$ all tuples $\mathbf{a}$ such that there is $v \in s$, $B \subseteq \mathcal{S}_v$, $W = J(\mathrm{ry}, v, B)$ with $\mathrm{pr}_{W\cap s}\mathbf{a} \notin \varrho'$ where $C' = \langle W \cap s, \varrho'\rangle$ is the corresponding constraint of $\mathcal{P}'_{\mathrm{ry},v,B}$
**endfor**
*Step 3.4.2.* **If** $\mathcal{P}$ changed **then return**(CONSERV($\mathcal{P}$))
*Step 3.4.3.* **else return**(MALTSEV($\mathcal{P}^s$))
**enddo**
*$\mathcal{S}_v$ is strongly r+b- and r+y-connected for any $v \in V$*:
*Step 3.5.* **return**(3-WIDTH($\mathcal{P}$))
**endifcase**
*Step 4.* **Invoke** 3-MINIMALITY($\mathcal{P}$)

## 6. Conclusion

We have continued the research aiming to distinguish those constraint languages that give rise to tractable constraint satisfaction problems from those which do not. We achieve this goal in the case of conservative constraint languages which, besides its own impotance, provide a possible prototype for the most general case of arbitrary constraint languages.

The main result of the paper has become possible due to the algebraic approach to the constraint satisfaction problem. This idea being originally developed for unifying and expanding known tractable classes has grown up to one the most prolific methods of studying the constraint satisfaction problem. It allows one to tackle problems which can hardly be solved by other existing methods, and we strongly believe that eventually it will make it possible to completely solve the indicated research problem.

## References

[1] R. Brewster, T. Feder, P. Hell, J. Huang, and G. MacGillivray. Near-unanimity functions and varieties of graphs. manuscript.

[2] A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings 43rd IEEE Symposium on Foundations of Computer Science, FOCS'02*, pages 649–658, Vancouver, Canada, November 2002.

[3] A. Bulatov. Mal'tsev constraints are tractable. Technical Report PRG-RR-02-05, Computing Laboratory, University of Oxford, Oxford, UK, 2002.

[4] A. Bulatov. Tractable constraints on a three-element set. Technical Report PRG-RR-02-06, Computing Laboratory, University of Oxford, Oxford, UK, 2002.

[5] A. Bulatov. Tractable conservative constraint satisfaction problems. Technical Report PRG-RR-03-01, Computing Laboratory, University of Oxford, Oxford, UK, 2003.

[6] A. Bulatov and P. Jeavons. Tractable constraints closed under a binary operation. Technical Report PRG-TR-12-00, Computing Laboratory, University of Oxford, Oxford, UK, 2000.

[7] A. Bulatov and P. Jeavons. Algebraic approach to multi-sorted constraints. Technical Report PRG-RR-01-18, Computing Laboratory, University of Oxford, Oxford, UK, 2001.

[8] A. Bulatov and P. Jeavons. Algebraic structures in combinatorial problems. Technical Report MATH-AL-4-2001, Technische universität Dresden, Dresden, Germany, 2001.

[9] A. Bulatov, P. Jeavons, and A. Krokhin. The complexity of maximal constraint languages. In *Proceedings of the 33rd Annual ACM Simposium on Theory of Computing*, pages 667–674, Hersonissos, Crete, Greece, July 2001. ACM Press.

[10] A. Bulatov, A. Krokhin, and P. Jeavons. Constraint satisfaction problems and finite algebras. In *Proceedings of 27th International Colloquium on Automata, Languages and Programming—ICALP'00*, volume 1853 of *Lecture Notes in Computer Science*, pages 272–282. Springer-Verlag, 2000.

[11] D. Cohen, M. Cooper, and P. Jeavons. Characterising tractable constraints. *Artificial Intelligence*, 65(2):347–361, 1994.

[12] M. Cooper. An optimal k-consistency algorithm. *Artificial Intelligence*, 41:89–95, 1989.

[13] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 2001.

[14] V. Dalmau. Constraint satisfaction problems in non-deterministic logarithmic space. In *Proceedings 29th International Colloquium on Automata, Languages and Programming, ICALP'02*, volume 2380 of *Lecture Notes in Computer Science*, pages 414–425. Springer-Verlag, 2002.

[15] V. Dalmau and J. Pearson. Set functions and width 1 problems. In *Proceedings 5th International Conference on Constraint Programming, CP'99*, volume 1713 of *Lecture Notes in Computer Science*, pages 159–173. Springer-Verlag, 1999.

[16] R. Dechter. From local to global consistency. *Artificial Intelligence*, 55(1):87–107, 1992.

[17] M. Dyer and C. Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17:260–289, 2000.

[18] T. Feder, P. Hell, and J. Huang. List homomorphisms and circular arc graphs. *Combinatorica*, 19:487–505, 1999.

[19] T. Feder, P. Hell, and J. Huang. Bi-arc graphs, and the complexity of list homomorphisms. *Journal of Graph Theory*, 42(1):61–80, 2003.

[20] T. Feder and M. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM Journal of Computing*, 28:57–104, 1998.

[21] A. Galuccio, P. Hell, and J. Nešetřil. The complexity of $h$-colouring of bounded degree graphs. *Discrete Math.*, pages 101–109, 2000.

[22] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA., 1979.

[23] P. Hell and J. Nešetřil. On the complexity of $H$-coloring. *Journal of Combinatorial Theory, Ser.B*, 48:92–110, 1990.

[24] P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

[25] P. Jeavons, D. Cohen, and M. Cooper. Constraints, consistency and closure. *Artificial Intelligence*, 101(1-2):251–265, 1998.

[26] P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44:527–548, 1997.

[27] P. Jeavons, D. Cohen, and J. Pearson. Constraints and universal algebra. *Annals of Mathematics and Artificial Intelligence*, 24:51–67, 1998.

[28] P. Kolaitis and M. Vardi. Conjunctive-query containment and constraint satisfaction. *J. Comput. Syst. Sci.*, 61:302–332, 2000.

[29] J. Kratochvil and Z. Tuza. Algorithmic complexity of list colorings. *Discrete Applied Mathematics*, 50:297–302, 1994.

[30] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 1974.

[31] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[32] N. Pippenger. *Theories of Computability*. Cambridge University Press, Cambridge, 1997.

[33] R. Pöschel and L. Kalužnin. *Funktionen- und Relationenalgebren*. DVW, Berlin, 1979.

[34] E. Post. *The two-valued iterative systems of mathematical logic*, volume 5 of *Annals Mathematical Studies*. Princeton University Press, 1941.

[35] T. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th ACM Symposium on Theory of Computing (STOC'78)*, pages 216–226, 1978.