# Amalgams of Constraint Satisfaction Problems

**Andrei A. Bulatov**
Oxford University Computing Laboratory
Oxford, UK
andrei.bulatov@comlab.ox.ac.uk

**Eugeny S. Skvortsov**
Ural State University, Ekaterinburg, Russia
Skvortsov@dc.ru

## Abstract

Many of standard practical techniques of solving constraint satisfaction problems use various decomposition methods to represent a problem as a combination of smaller ones. We study a general method of decomposing constraint satisfaction problems in which every constraint is represented as a disjunction of two or more simpler constraints defined, possibly, on smaller sets of values. We call a problem an amalgam if it can be decomposed in this way. Some particular cases of this construction have been considered in [Cohen *et al.*, 1997; 2000b; 2000a] including amalgams of problems with disjoint sets of values, and amalgams of independent problems. In this paper, we concentrate on constraint classes determined by relational clones, and study amalgams of such classes in the general case of arbitrary finite sets of values. We completely characterise amalgams of this form solvable in polynomial time and provide efficient algorithms.

## 1 Introduction

In the Constraint Satisfaction Problem (CSP) the aim is to find an assignment to a set of variables subject specified constraints. CSP provides a generic approach to various combinatorial problems frequently appearing in artificial intelligence and computer science, including the propositional satisfiability problem, in which the variables must be assigned Boolean values, graph-theoretical problems, scheduling problems, temporal and spatial reasoning, database theory, and many others.

The general CSP is NP-complete [Montanari, 1974]. However, the time complexity of many practical problems can be considerably decreased by representing a problem as a combination of smaller problems. Such decomposition methods have been intensively studied and implemented in most of existing constraint solvers. Most of them deals with the hypergraph associated with a problem instance (see, e.g. [Gottlob *et al.*, 2000; Dechter and Pearl, 1989; Freuder, 1990]), and to date are highly developed. Another possibility, decomposition of constraints themselves [Cohen *et al.*, 1997; 2000b; 2000a], remains almost uninvestigated. The present paper focuses on this class of decomposition methods.

In many cases of interest we may restrict the form of allowed constraints by specifying a constraint language, that is a set of allowed constraints. Every constraint is specified by a relation, so, a constraint language is just a set of relations over the set of values. One of the most natural ways to decompose such a restricted CSP is to represent every its constraint as a disjunction of two or more simpler constraints on, possibly, smaller sets of values. We say that a constraint language, $\Gamma$, is the *amalgam* of $\Gamma_1, \Gamma_2$ if every constraint in $\Gamma$ is the disjunction of a constraint from $\Gamma_1$ and a constraint from $\Gamma_2$. In this case we call $\Gamma_1, \Gamma_2$ the *components* of $\Gamma$.

The main research direction in the study of amalgams is, of course, developing efficient algorithms solving the corresponding constraint satisfaction problem. However, as we shall see the complexity of the CSP arising from an amalgam is not determined automatically by the complexity of its components. Therefore, the first research problem we deal with is the *tractability problem*: under which conditions the problem arising from an amalgam is tractable. (A problem is called tractable if it can be solved in polynomial time.) Then we tackle the *algorithmic problem*: design efficient algorithms for tractable amalgams.

As is naturally expected, algorithms for amalgams tend to use algorithms for their components, especially if ones are already known. Unfortunately, in general, the connection between constraint languages and their amalgam cannot be expressed by usual constraint techniques, and strongly depends on properties of particular constraint languages. Thus, solutions to both research problems are expected to be nontrivial.

Amalgams have been introduced in [Cohen *et al.*, 1997], though another name was used. In [Cohen *et al.*, 2000a], amalgams were considered in the simplest case when the sets of values for languages $\Gamma_1, \Gamma_2$ are disjoint. In this case, for any amalgam, the corresponding problem is trivially reducible to the problems over the components, that solves both the algorithmic problem and the tractability problem: an amalgam gives rise to a tractable problem if and only its components do. Certain properties of interaction of constraint languages may yield a reduction of an amalgam to its components. In [Cohen *et al.*, 1997] and later in [Cohen *et al.*, 2000b], several such properties, so-called *independence* of constraint languages, have been identified. A number of previously unknown tractable constraint languages have been represented as amalgams of very simple and well studied independent constraint languages.

It has been shown in [Jeavons, 1998b] that the tractability of the constraint satisfaction problem arising from a constraint language implies the tractability of the problem for the *relational clone* generated by the language. Therefore, a reasonable strategy is to concentrate on relational clones rather than arbitrary constraint languages. In this paper we solve the tractability and algorithmic problems for amalgams of relational clones. We completely characterise tractable amalgams and provide efficient algorithms in this case. The characterisation criteria is stated in terms of, first, properties of components of an amalgam, and second, the tractability of a certain constraint language on a 2-element set of values. The latter language reflects the interaction of the components. Thus, both the characterisation and the reducing algorithms use Schaefer's Dichotomy Theorem for Boolean constraints [Schaefer, 1978].

Throughout the paper we heavily use the algebraic technique for CSP developed in [Jeavons, 1998b; Jeavons *et al.*, 1998].

# 2 Preliminaries

## 2.1 Basic Definitions

Let $A$ be a finite set. The set of all $n$-tuples of elements of $A$ is denoted $A^n$. A subset of $A^n$ is called an $n$-ary *relation* on $A$, and any set of finitary relations on $A$ is called a *constraint language* on $A$.

**Definition 1** *Let $\Gamma$ be a constraint language on a set $A$. $\mathrm{CSP}(\Gamma)$ is the combinatorial decision problem whose instance is a triple $\mathcal{P} = (V; A; \mathcal{C})$ in which $V$ is a set of variables, and $\mathcal{C}$ is a set of* constraints*, that is pairs of the form $C = \langle s, \varrho\rangle$ where $s$ is a list of variables of length $m_C$, called the* constraint scope*, and $\varrho \in \Gamma$ is an $m_C$-ary relation, called the* constraint relation*. The question is whether there exists a solution to $\mathcal{P}$, that is a mapping $\varphi: V \to A$ such that $\varphi(s) \in \varrho$ for all $\langle s, \varrho\rangle \in \mathcal{C}$.*

**Example 1** An instance of GRAPH $q$-COLORABILITY consists of a graph $G$. The question is whether the vertices of $G$ can be labelled with $q$ colours so that adjacent vertices are assigned different colours.

This problem corresponds to $\mathrm{CSP}(\{\neq_A\})$ where $A$ is a $q$-element set (of colours) and $\neq_A = \{(a,b) \in A^2 \mid a \neq b\}$ is the disequality relation on $A$.

A constraint language $\Gamma$ is said to be *tractable* [*NP-complete*] if the problem $\mathrm{CSP}(\Gamma)$ is tractable [NP-complete].

Now we introduce the central notion of the paper.

**Definition 2** *Let $\Gamma_A, \Gamma_B$ be constraint languages on sets $A$, $B$ respectively. The* amalgam *of $\Gamma_A, \Gamma_B$ is defined to be the constraint language on $A \cup B$*

$$\Gamma_A \dot{\bowtie} \Gamma_B = \{\varrho_A \cup \varrho_B \mid \varrho_A \in \Gamma_A,\ \varrho_B \in \Gamma_B,\ \varrho_A, \varrho_B \textit{ are of the same arity}\}.$$

**Example 2** Let $A = \{a, c, d\}, B = \{b, c, d\}$, and $\Gamma_A = \{\varrho_A\}, \Gamma_B = \{\varrho_B\}$ where $\varrho_A, \varrho_B$ are partial orders:

$$\varrho_A = \begin{pmatrix} a & c & d & c & a & a \\ a & c & d & d & d & c \end{pmatrix}, \qquad \varrho_B = \begin{pmatrix} b & c & d & c & d & d \\ b & c & d & b & b & c \end{pmatrix}.$$

Columns of the matrices represent tuples of the relations. Then $\Gamma_A \dot{\bowtie} \Gamma_B = \{\varrho_A \cup \varrho_B\}$ where $\varrho_A \cup \varrho_B$ is a quasiorder.

**Example 3 ([Cohen *et al.*, 2000b])** Let $\mathbb{Z}$ be the set of all integers, and $\Gamma$ the set of congruences of the form $x \equiv a \pmod m$ treated as unary relations. The Chinese Remainder Theorem implies that $\Gamma$ is tractable. The amalgam $\Gamma \dot{\bowtie} \Gamma$ consists of expressions of the form $x \equiv a \pmod m \vee x \equiv b \pmod n$, and by results of [Cohen *et al.*, 2000b] is tractable. For example, this means that we are able to recognised in polynomial time the consistency of the system

$$\begin{aligned} &x_1 \equiv 1 \pmod 4 \vee x_1 \equiv 0 \pmod 3 \\ &x_2 \equiv 2 \pmod 3 \vee x_2 \equiv 2 \pmod 5 \\ &x_3 \equiv 5 \pmod 7. \end{aligned}$$

We are concerned with the following two problems.

**Problem 1 (tractability problem)** *When is* $\mathrm{CSP}(\Gamma_1 \dot{\bowtie} \Gamma_2)$ *tractable?*

**Problem 2 (algorithmic problem)** *Find a polynomial time algorithm for tractable* $\mathrm{CSP}(\Gamma_1 \dot{\bowtie} \Gamma_2)$.

In fact, properties of the amalgam do not strongly depend neither on properties of the original constraint languages nor on the way they interact. For example, if $\Gamma_1, \Gamma_2$ contains no relations of the same arity then their amalgam is empty. We, therefore, should restrict the class of constraint languages to be studied.

## 2.2 Relational Clones

For any problem in $\mathrm{CSP}(\Gamma)$, there may be some sets of variables whose possible values subject to certain constraints which are not elements of $\Gamma$. These constraints are said to be *implicit* and arise from interaction of constraints specified in the problem [Jeavons, 1998a].

To describe implicit constraint relations we make use of the natural correspondence between relations and predicates: for an $n$-ary relation $\varrho$ on a set $A$, the $n$-ary predicate $P_\varrho$ is true on a tuple $\mathbf{a}$ if and only if $\mathbf{a} \in \varrho$. Usually, we will not distinguish a relation and the corresponding predicate, and freely use both terminology. An existential first order formula $\exists y_1, \ldots, y_m \Phi(x_1, \ldots, x_n, y_1, \ldots, y_m)$ is said to be *primitive positive (pp-)* if its quantifier-free part $\Phi$ is a conjunction of atomic formulas.

**Definition 3** *A relation is an* implicit relation *of a constraint language $\Gamma$ on a set $A$ if it can be expressed by a pp-formula involving relations from $\Gamma$ and the equality relation $=_A$.*

*A constraint language $\Gamma$ is said to be a* relational clone *if it contains all its implicit constraint relations.*

*The relational clone $\langle\Gamma\rangle$ of all implicit constraint relations of $\Gamma$ is called the relational clone* generated *by $\Gamma$.*

**Example 4** The intersection of relations of the same arity, and Cartesian product are expressible via pp-formulas:

• Let $\varrho, \sigma$ be $n$-ary relations. Then $P_{\varrho\cap\sigma}(x_1, \ldots, x_n) = P_\varrho(x_1, \ldots, x_n) \wedge P_\sigma(x_1, \ldots, x_n)$.

• Let $\varrho$ be $n$-ary, and $\sigma$ $m$-ary relations. Then $P_{\varrho\times\sigma} = P_\varrho(x_1, \ldots, x_n) \wedge P_\sigma(x_{n+1}, \ldots, x_{n+m})$.

The notion of a relational clone considerably simplifies the analysis of constraint satisfaction problems in view of the following result that links the complexity of a constraint language and the relational clone it generates.

**Theorem 1 ([Jeavons, 1998b])** *Let $\Gamma$ be a constraint language on a finite set, and $\Gamma' \subseteq \langle\Gamma\rangle$ finite. Then* $\mathrm{CSP}(\Gamma')$ *is polynomial time reducible to* $\mathrm{CSP}(\Gamma)$.

This result motivates restricting Problems 1,2 to the class of relational clones.

**Example 3 (continuation)** Reconsider the constraint language $\Gamma$ from Example 3. Results of [Cohen *et al.*, 2000b] implies that $\langle\Gamma\rangle \dot{\bowtie} \langle\Gamma\rangle$ is also tractable. This larger amalgam includes, e.g., constraints of the form $x \equiv a \pmod{m} \vee y \equiv b \pmod{n}$, which are not members of $\Gamma \dot{\bowtie} \Gamma$.

### 2.3 Invariance Properties of Constraints

Another advantage of considering relational clones is that they often admit a concise description in terms of algebraic invariance properties [Pöschel and Kalužnin, 1979; Jeavons, 1998b]. An ($m$-ary) operation $f$ on a set $A$ *preserves* an $n$-ary relation $\varrho$ on $A$ (or $\varrho$ is *invariant* under $f$, or $f$ is a *polymorphism* of $\varrho$) if for any $(a_{11}, \ldots, a_{n1}), \ldots, (a_{1m}, \ldots, a_{nm}) \in \varrho$ the tuple $(f(a_{11}, \ldots, a_{1m}), \ldots, f(a_{n1}, \ldots, a_{nm}))$ belongs to $\varrho$. For a given set of operations, $C$, the set of all relations invariant under every operation from $C$ is denoted by $\mathsf{Inv}\ C$. Conversely, for a set of relations, $\Gamma$, the set of all operations preserving every relation from $\Gamma$ is denoted by $\mathsf{Pol}\ \Gamma$. Every relational clone can be represented in the form $\mathsf{Inv}\ C$ for a certain set of operations $C$ [Pöschel and Kalužnin, 1979]. Therefore, in view of Theorem 1 the complexity of a finite constraint language depends only on its polymorphisms.

We need operations of some particular types that give rise to tractable problem classes.

**Definition 4** *Let $A$ be a finite set. An operation $f$ on $A$ is called*

- *a* constant *operation if there is $c \in A$ such that $f(x_1, \ldots, x_n) = c$, for any $x_1, \ldots, x_n \in A$;*
- *a* semilattice *operation*[1], *if it is binary and satisfies the following three conditions: (a) $f(x, x) = x$, (b) $f(x, f(y, z)) = f(f(x, y), z)$, (c) $f(x, y) = f(y, x)$, for any $x, y, z \in A$;*
- *a* majority *operation if it is ternary, and $f(x, x, y) = f(x, y, x) = f(y, x, x) = x$, for any $x, y \in A$;*
- affine *if $f(x, y, z) = x - y + z$, for any $x, y, z \in A$, where $+, -$ are the operations of an Abelian group.*

**Proposition 1 ([Jeavons *et al.*, 1998; 1997])** *If $\Gamma$ is a constraint language on a finite set, and* $\mathsf{Pol}\ \Gamma$ *contains an operation of one of the following types: constant, semilattice, affine, majority; then $\Gamma$ is tractable.*

The complexity of constraint languages on a 2-element set is completely characterised in [Schaefer, 1978]. This outstanding result is known as Schaefer's Dichotomy Theorem. By making use of Proposition 1 the algebraic version of Schaefer's theorem can be derived [Jeavons *et al.*, 1997].

[1]Note that in some earlier papers [Jeavons, 1998b; Jeavons *et al.*, 1998] the term *ACI operation* is used.

**Theorem 2 (Schaefer, [Schaefer, 1978])** *A constraint language $\Gamma$ on a 2-element set is tractable if and only if* $\mathsf{Pol}\ \Gamma$ *contains one of the operations listed in Proposition 1. Otherwise $\Gamma$ is NP-complete.*

## 3 Tractable amalgams

In this section we give a complete solution of Problems 1,2 for amalgams of relational clones. Throughout the section $A$,$B$ are finite sets, $D = A \cap B$, and $R_A, R_B$ are relational clones on $A, B$ respectively. First, we reduce Problems 1,2 to the case when $|D| = 1$ and prove NP-completeness results in this case. Then we concentrate on tractable cases and present a solving algorithm for these cases.

### 3.1 The result

We assume that $R_A, R_B$ contain empty relations of any arity, because otherwise $\mathsf{Pol}\ (R_A \dot{\bowtie} R_B)$ contains a constant operation, and therefore $R_A \dot{\bowtie} R_B$ is tractable by Proposition 1.

The case when $D$ is empty was completely investigated in [Cohen *et al.*, 2000a].

**Proposition 2 ([Cohen *et al.*, 2000a])** *If $D = \varnothing$ then* $\mathrm{CSP}(R_A \dot{\bowtie} R_B)$ *is polynomial time reducible to* $\mathrm{CSP}(R_A)$, $\mathrm{CSP}(R_B)$. *Hence, $R_A \dot{\bowtie} R_B$ is tractable if and only if both $R_A, R_B$ are tractable.*

So, we assume $D$ to be non-empty.

**Proposition 3 ([Jeavons *et al.*, 1997; Jeavons, 1998b])** *If $f$ is a unary polymorphism of a constraint language $\Gamma$ then the problem class* $\mathrm{CSP}(f(\Gamma))$ *where $f(\Gamma) = \{f(\varrho) \mid \varrho \in \Gamma\}$, $f(\varrho) = \{(f(a_1), \ldots, f(a_n)) \mid (a_1, \ldots, a_n) \in \varrho\}$, is polynomial time equivalent to* $\mathrm{CSP}(\Gamma)$.

By $f\big|_C$ we denote the restriction of a (unary) operation $f$ onto a set $C$. The following statement is straightforward.

**Proposition 4** *(1) If $f$ is a unary polymorphism of $R_A \dot{\bowtie} R_B$ then $f\big|_A$, $f\big|_B$ are polymorphisms of $R_A$, $R_B$, respectively.*

*(2)* $\mathrm{CSP}(R_A \dot{\bowtie} R_B)$ *is polynomial time equivalent to* $\mathrm{CSP}(f\big|_A(R_A) \dot{\bowtie} f\big|_B(R_B))$.

*(3) If the amalgam $R_A \dot{\bowtie} R_B$ is tractable then there is a polymorphism $f$ of $R_A \dot{\bowtie} R_B$ such that $|f(D)| = 1$. Otherwise, $R_A \dot{\bowtie} R_B$ is NP-complete.*

Therefore, we may restrict ourselves to the case $|D| = 1$; let $D = \{c\}$. In this case we need more notation and terminology.

**Definition 5** *The relational clone $R_A$ [$R_B$] is said to be* monolithic *if $\overline{R}_A = \langle R_A \cup \{D\}\rangle$ [$\overline{R}_B = \langle R_B \cup \{D\}\rangle$] contains no unary relation $E$ with $c \notin E$.*

Let $R_A^+$ denote the relational clone generated by the set $R_A \cup \{\varrho \cup \{\overline{c}\} \mid \varrho \in R_A\}$ where $\overline{c} = (c, \ldots, c)$[2]. Obviously, $\overline{R}_A \subseteq \mathrm{R}_A^+$. The clone $R_B^+$ is defined analogously.

[2]We do not specify the length of the tuple $\overline{c} = (c, \ldots, c)$, because it is always clear from the context.

In the following definition we introduce an auxiliary relational clone on the 2-element set $Z = \{A, B\}$ that describes interaction of the components of the amalgam. For an ($n$-ary) relation $\varrho \in R_A$, let $\widetilde{\varrho} = \{(a_1, \dots, a_n, A) \mid (a_1, \dots, a_n) \in \varrho_A\} \cup \{(c, \dots, c, B)\}$. We also define the corresponding predicate $\widetilde{P}_\varrho(x_1, \dots, x_n, y)$ which is true if and only if $(x_1, \dots, x_n, y) \in \widetilde{\varrho}$. The variable $y$ will be called *distinguished*.

**Definition 6** *The* meta-clone $\mathbb{R}_A$ *is defined to be the set of all relations on $Z$ expressible through a pp-formula of the form $\exists x_1, \dots, x_n z_1, \dots, z_k \Phi(x_1, \dots, x_n, y_1, \dots, y_m, z_1, \dots, z_k)$ where $\Phi$ is a conjunction of predicates $\widetilde{P}_\varrho$ for $\varrho \in R_A^+$, and $y_1, \dots, y_m, z_1, \dots, z_k$ are the distinguished variables.*

*The meta-clone $\mathbb{R}_B$ is defined in a similar way.*

**Lemma 1** *Meta-clones $\mathbb{R}_A, \mathbb{R}_B$ are relational clones on $Z$.*

We now in a position to state the main result of this paper.

**Theorem 3** *Let $|D| = 1$. Then*

*1. if $R_A$ is monolithic then $R_A \dot{\bowtie} R_B$ is tractable if and only if $R_B^+$ is tractable;*

*2. if both $R_A, R_B$ are not monolithic then $R_A \dot{\bowtie} R_B$ is tractable if and only if $R_A^+$, $R_B^+$, and $\mathbb{R}_A \cup \mathbb{R}_B$ are tractable.*

It is not hard to see that $R_A^+, R_B^+ \subseteq \langle R_A \dot{\bowtie} R_B \rangle$. Therefore, by Theorem 1, $R_B^+$ is tractable whenever $R_A \dot{\bowtie} R_B$ is tractable. The converse inclusion follows from Lemma 2. Then, it can be proved by straightforward calculation that CSP($\mathbb{R}_A \cup \mathbb{R}_B$) is polynomial time reducible to CSP(R$_A \dot{\bowtie} R_B$). Finally, in Sections 3.2,3.3, we show how to reduce CSP($R_A \dot{\bowtie} R_B$) to CSP($\mathbb{R}_A \cup \mathbb{R}_B$), CSP($R_A^+$), and CSP($R_B^+$).

**Lemma 2** *If $R_A$ is monolithic then* CSP$(R_A \dot{\bowtie} R_B)$ *is polynomial time reducible to* CSP$(R_B^+)$.

**Proof:** Let $R_A$ be monolithic. We set out the proof in several claims.

CLAIM 1. For any $a \in A - \{c\}$, there is a unary polymorphism $f_a$ of $\overline{R}_A$ such that $f_a(a) = c$.

Let $g_1, \dots, g_k$ be the list of all unary polymorphisms of $\overline{R}_A$, and $E = \{a_1 = g_1(a), \dots, a_k = g_k(a)\}$. Suppose that $c \notin E$. Since $R_A$ is monolithic, $E \notin \overline{R}_A$. Therefore, there is a polymorphism $h(x_1, \dots, x_n)$ such that $h(b_1, \dots, b_n) \notin E$ for some $b_i \in E$, say, $b_i = a_{j_i}$. The operation $h'(x) = h(g_{j_1}(x), \dots, g_{j_n}(x))$ is also a polymorphism of $\overline{R}_A$ [Pöschel and Kalužnin, 1979], and $h'(a) \notin E$, a contradiction.

CLAIM 2. $\overline{R}_A$ is invariant under the constant operation $c$.

Since $D = \{(c)\} \in \overline{R}_A$, for any $a \in A - \{c\}$, we have $f_a(c) = c$. This implies $|f_{a_1}(A)| \le |A| - 1$ for any $a_1 \in A - \{c\}$. Then choosing an element $a_2 \in f_{a_1}(A) - \{c\}$ we get $|f_{a_2}(f_{a_1}(A))| \le |A| - 2$. Continuing the process we eventually obtain an operation $f(x) = f_{a_l}(\dots f_{a_1}(x) \dots)$ such that $|f(A)| = 1$. As $f(c) = c$, $f$ is the required constant operation.

Claim 2 implies that if $\varrho \in \overline{R}_A$ is non-empty then $\overline{c} \in \varrho$.

CLAIM 3. The operation $g$ on $A \cup B$ where $g(x) = c$ if $x \in A$ and $g(x) = x$ if $x \in B$ is a polymorphism of $R_A \dot{\bowtie} R_B$.

Take $\varrho \in R_A \dot{\bowtie} R_B$ and $\mathbf{d} \in \varrho = \varrho_A \cup \varrho_B$, $\varrho_A \in R_A$, $\varrho_B \in R_B$. If $\mathbf{d} \in \varrho_B$ then $g(\mathbf{d}) = \mathbf{d} \in \varrho$. If $\mathbf{d} \in \varrho_A$ then $\varrho_A \neq \varnothing$, $\overline{c} \in \varrho_A \subseteq \varrho$, and therefore $g(\mathbf{d}) = \overline{c} \in \varrho$.

Finally, for any $\varrho = \varrho_A \cup \varrho_B \in R_A \dot{\bowtie} R_B$, $g(\varrho) = \varrho_B$ or $g(\varrho) = \varrho_B \cup \{\overline{c}\}$. This means that $g(R_A \dot{\bowtie} R_B) = R_B^+$, and by Proposition 3, $CSP(R_A \dot{\bowtie} R_B)$ is polynomial time equivalent to CSP$(R_B^+)$. □

## 3.2 The Split Problem

We transform a problem over an amalgam so that the interaction of its components becomes more transparent.

**Lemma 3** *If $R_A$ [$R_B$] is not monolithic then there is a relation $\sigma_A \in R_A$ [$\sigma_B \in R_B$] of arity $l_A$ [$l_B$] such that $\overline{c} \notin \sigma_A$ [$\overline{c} \notin \sigma_B$].*

**Proof:** Let $E \in \overline{R}_A$ be such that $c \notin E$. Since $E$ belongs to the clone generated by $R_A \cup \{D\}$, it can be expressed by a pp-formula

$$P_E(x) = \exists y_1, \dots, y_n, z_1, \dots, z_m \; \Phi(x, y_1, \dots, y_n, z_1, \dots, z_m) \wedge P_D(z_1) \wedge \dots \wedge P_D(z_m)$$

where $\Phi$ uses predicates from $R_A$. As is easily seen, the relation $P_\sigma(x, z_1, \dots, z_m) = \exists y_1, \dots, y_n \Phi(x, y_1, \dots, y_n, z_1, \dots, z_m)$ satisfies the required conditions. □

Let $\mathcal{P} = (V; A \cup B; \mathcal{C})$ be a problem instance from CSP($R_A \dot{\bowtie} R_B$). Fix a representation $\varrho = \varrho_A \cup \varrho_B$, $\varrho_A \in R_A$, $\varrho_B \in R_B$ for every constraint relation $\varrho$. The *split problem* $\widetilde{\mathcal{P}}$ is defined to be the triple $(\widetilde{V}; A \cup B; \widetilde{\mathcal{C}})$ (see Fig.1) in which

- $\widetilde{V} = V_A \cup V_B \cup V'_A \cup V'_B$ where for every $v \in V$ there are $v^A \in V_A$ and $v^B \in V_B$, and for every constraint $C \in \mathcal{C}$ there are $u_C^1, \dots, u_C^{l_A} \in V'_A$ and $w_C^1, \dots, w_C^{l_B} \in V'_B$; let us denote $\overline{u}_C, \overline{w}_C$ the sequences $u_C^1, \dots, u_C^{l_A}$, $w_C^1, \dots, w_C^{l_B}$ respectively;
- $\widetilde{\mathcal{C}} = \mathcal{C}_A \cup \mathcal{C}_B \cup \mathcal{C}'$ where for every $C = \langle s, \varrho \rangle \in \mathcal{C}$, $s = (v_1, \dots, v_n)$, we include 3 constraints
  - $\langle (v_1^A, \dots, v_n^A, \overline{u}_C), \varrho'_A \rangle \in \mathcal{C}_A$, $\varrho'_A = (\varrho \times \sigma_A) \cup \{\overline{c}\}$;
  - $\langle (v_1^B, \dots, v_n^B, \overline{w}_C), \varrho'_B \rangle \in \mathcal{C}_B$, $\varrho'_B = (\varrho \times \sigma_B) \cup \{\overline{c}\}$;
  - $\langle (\overline{u}_C, \overline{w}_C), \sigma \rangle \in \mathcal{C}'$ where $\sigma$ is the $l_A + l_B$-ary relation $(\sigma_A \times \{\overline{c}\}) \cup (\{\overline{c}\} \times \sigma_B)$.

The following lemma is straightforward.

**Lemma 4** *$\mathcal{P}$ is equivalent to $\widetilde{\mathcal{P}}$.*

On the one hand, the restrictions of $\widetilde{\mathcal{P}}$ on $V_A \cup V'_A$, $V_B \cup V'_B$ are problems from CSP($R_A^+$), CSP($R_B^+$) respectively. On the other hand, the implicit constraint relation arising on the set $V'_A$ in this problem can be treated as a relation on the 2-element set $\{\sigma_A, D^{l_A}\}$, and correspond, in fact, to the relations from $\mathbb{R}_A$. Therefore, the meta-clones describe the interaction of two parts of the split problem.

## 3.3 Solving the Split Problem

Let us assume that $R_A^+, R_B^+, \mathbb{R}_A \cup \mathbb{R}_B$ are tractable, and $R_A, R_B$ are not monolithic. By Theorem 2, there is either (1) constant, or (2) semilattice, or (3) majority, or (4) affine operation $f$ such that every relation from $\mathbb{R}_A \cup \mathbb{R}_B$ is invariant under $f$. We show that under this assumption the split problem

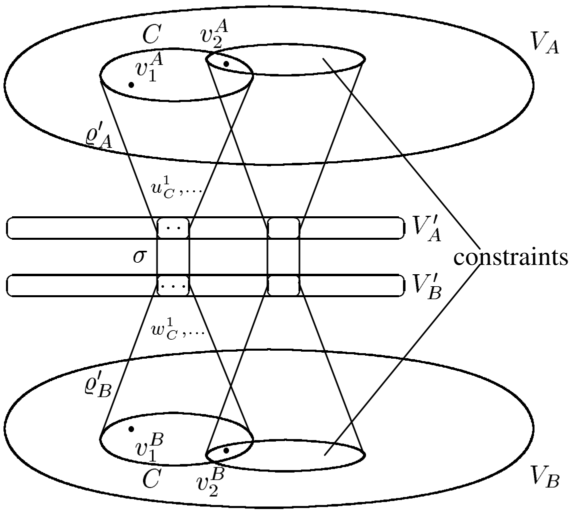


Figure 1:

$\widetilde{\mathcal{P}}$ can be reduced to problems from $\mathrm{CSP}(R_A^+)$, $\mathrm{CSP}(R_B^+)$ in polynomial time.

We divide $\widetilde{\mathcal{P}}$ into two subproblems $\widetilde{\mathcal{P}}_A = (V_A \cup V'_A; A; \mathcal{C}_A)$ and $\widetilde{\mathcal{P}}_B = (V_B \cup V'_B; B; \mathcal{C}_B)$ from $\mathrm{CSP}(R_A^+)$, $\mathrm{CSP}(R_B^+)$ respectively. Recall also that $\mathcal{C} = \{C_1, \dots, C_q\}$. Then set

$$\begin{aligned}\mathcal{S}_A &= \{(\varphi(u_{C_1}^1), \dots, \varphi(u_{C_1}^{l_A}), \varphi(u_{C_2}^1), \dots, \varphi(u_{C_q}^{l_A})) \mid \\ &\qquad \varphi \text{ is a solution to } \widetilde{\mathcal{P}}_A\}, \\ \mathcal{S}_B &= \{(\varphi(w_{C_1}^1), \dots, \varphi(w_{C_1}^{l_B}), \varphi(w_{C_2}^1), \dots, \varphi(w_{C_q}^{l_B})) \mid \\ &\qquad \varphi \text{ is a solution to } \widetilde{\mathcal{P}}_B\}.\end{aligned}$$

Let also mappings $\psi_A : \sigma_A \cup \{\overline{c}\} \to Z$, $\psi_B : \sigma_B \cup \{\overline{c}\} \to Z$ be defined as follows

$$\psi_A(\mathbf{d}) = \begin{cases} A, & \text{if } \mathbf{d} \in \sigma_A, \\ B, & \text{if } \mathbf{d} = \overline{c}, \end{cases} \qquad \psi_B(\mathbf{d}) = \begin{cases} A, & \text{if } \mathbf{d} = \overline{c}, \\ B, & \text{if } \mathbf{d} \in \sigma_B. \end{cases}$$

Then $\overline{\mathcal{S}}_A = \psi_A(\mathcal{S}_A) = \{(\psi_A(a_{C_1}^1, \dots, a_{C_1}^{l_A}), \dots, \psi_A(a_{C_q}^1, \dots, a_{C_q}^{l_A})) \mid (a_{C_1}^1, \dots, a_{C_q}^{l_A}) \in \mathcal{S}_A\}$ is a $q$-ary relation from $\mathbb{R}_A$, and $\overline{\mathcal{S}}_B = \psi_B(\mathcal{S}_B) = \{(\psi_B(b_{C_1}^1, \dots, b_{C_1}^{l_B}), \dots, \psi_B(b_{C_q}^1, \dots, b_{C_q}^{l_B})) \mid (b_{C_1}^1, \dots, b_{C_q}^{l_B}) \in \mathcal{S}_B\}$ is a $q$-ary relation from $\mathbb{R}_B$. A solution to $\widetilde{\mathcal{P}}$ exists if and only if there are solutions $\varphi_A \in \mathcal{S}_A$, $\varphi_B \in \mathcal{S}_B$ such that $\psi_A\varphi_A = \psi_B\varphi_B$ or, equivalently, if $\overline{\mathcal{S}}_A \cap \overline{\mathcal{S}}_B \neq \varnothing$. The $i$th component of a tuple $\mathbf{d}$ will be denoted by $\mathbf{d}[i]$. Fix tuples $\mathbf{a} \in \sigma_A$, $\mathbf{b} \in \sigma_B$.

CASE 1. $\mathbb{R}_A \cup \mathbb{R}_B$ is invariant under a constant operation.

Without loss of generality we may assume that the constant operation maps $Z$ to $A$. Then either $\overline{\mathcal{S}}_A$ [$\overline{\mathcal{S}}_B$] is empty, or $(A, \dots, A) \in \overline{\mathcal{S}}_A$ [$(A, \dots, A) \in \overline{\mathcal{S}}_B$]. If $\overline{\mathcal{S}}_A = \varnothing$ or $\overline{\mathcal{S}}_B = \varnothing$ then $\mathcal{S}_A$ or $\mathcal{S}_B$ is empty, consequently, $\widetilde{\mathcal{P}}$ has no solution. If $\overline{\mathcal{S}}_A, \overline{\mathcal{S}}_B \neq \varnothing$ then $\widetilde{\mathcal{P}}_A$ has a solution $\varphi_A \in \mathcal{S}_A$ such that $\varphi_A(u_C^i) = \mathbf{a}[i]$ for all $C \in \mathcal{C}$, $1 \le i \le l_A$, and $\widetilde{\mathcal{P}}_B$ has a solution $\varphi_B \in \mathcal{S}_B$ with $\varphi_B(w_C^j) = c$ for all $C \in \mathcal{C}$, $1 \le j \le l_B$. As is easily seen, $\psi_A\varphi_A = \psi_B\varphi_B$.

We need some additional notation. Let $\varrho$ be an $n$-ary relation, and $I = \{i_1, \dots, i_k\} \subseteq \{1, \dots, n\}$; then $\pi_I(\varrho)$ denotes the $k$-ary relation $\{\pi_I(\mathbf{d}) \mid \mathbf{d} \in \varrho\}$ where $\pi_I(\mathbf{d}) = (\mathbf{d}[i_1], \dots, \mathbf{d}[i_k])$. For a problem instance $\mathcal{P} = (W; D; \mathcal{C})$ and a constraint $\langle s, \varrho \rangle$, we denote the problem instance $(W; D; \mathcal{C} \cup \{\langle s, \varrho \rangle\})$ by $\mathcal{P} \cup \{\langle s, \varrho \rangle\}$.

CASE 2. $\mathbb{R}_A \cup \mathbb{R}_B$ is invariant under a semilattice operation $f$.

Without loss of generality we may assume that $f(A, A) = f(A, B) = f(B, A) = A$, $f(B, B) = B$. Then [Jeavons *et al.*, 1997] any relation $\varnothing \neq \varrho \in \mathbb{R}_A \cup \mathbb{R}_B$ contains the tuple $\max_\varrho$ where $\max_\varrho[i] = A$ if $A \in \pi_{\{i\}}(\varrho)$, and $\max_\varrho[i] = B$ otherwise. We employ the following algorithm.

**Until** no new constraint added, **for** each $C \in \mathcal{C}$ **do**

- **Solve** the problems $\widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_C, \sigma_A \rangle\}$, $\widetilde{\mathcal{P}}_B \cup \{\langle \overline{w}_C, \{\overline{c}\} \rangle\}$.
- **If** one of them has no solution **then replace** $\widetilde{\mathcal{P}}_A$ with $\widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_C, \{\overline{c}\} \rangle\}$ and $\widetilde{\mathcal{P}}_B$ with $\widetilde{\mathcal{P}}_B \cup \{\langle \overline{w}_C, \sigma_B \rangle\}$.
- **Solve** the problems $\widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_C, \{\overline{c}\} \rangle\}$, $\widetilde{\mathcal{P}}_B \cup \{\langle \overline{w}_C, \sigma_B \rangle\}$.
- **If** one of them has no solution **then replace** $\widetilde{\mathcal{P}}_A$ with $\widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_C, \sigma_A \rangle\}$ and $\widetilde{\mathcal{P}}_B$ with $\widetilde{\mathcal{P}}_B \cup \{\langle \overline{w}_C, \{\overline{c}\} \rangle\}$.

Notice that this algorithm is quite analogous to *establishing 1-consistency* [Jeavons *et al.*, 1998; Dechter and van Beek, 1997; Cooper, 1989]. Let $\widetilde{\mathcal{P}}'_A, \widetilde{\mathcal{P}}'_B$ be the obtained problems, $\mathcal{S}'_A$, $\mathcal{S}'_B$ defined for $\widetilde{\mathcal{P}}'_A, \widetilde{\mathcal{P}}'_B$ in the same way as $\mathcal{S}_A, \mathcal{S}_B$, and $\overline{\mathcal{S}}'_A = \psi_A(\mathcal{S}'_A)$, $\overline{\mathcal{S}}'_B = \psi_B(\mathcal{S}'_B)$. Clearly, $\overline{\mathcal{S}}_A \cap \overline{\mathcal{S}}_B \neq \varnothing$ if and only if $\overline{\mathcal{S}}'_A \cap \overline{\mathcal{S}}'_B \neq \varnothing$. Moreover, since $\pi_{\{i\}}\overline{\mathcal{S}}'_A = \pi_{\{i\}}\overline{\mathcal{S}}'_B$, for any $i$, we have $\max_{\overline{\mathcal{S}}'_A} = \max_{\overline{\mathcal{S}}'_B} \in \overline{\mathcal{S}}'_A \cap \overline{\mathcal{S}}'_B$ whenever $\overline{\mathcal{S}}'_A, \overline{\mathcal{S}}'_B \neq \varnothing$. Thus, $\widetilde{\mathcal{P}}$ has a solution if and only if $\widetilde{\mathcal{P}}'_A, \widetilde{\mathcal{P}}'_B$ have solutions.

CASE 3. $\mathbb{R}_A \cup \mathbb{R}_B$ is invariant under a majority operation $f$.

In this case any ($n$-ary) relation $\varrho \in \mathbb{R}_A \cup \mathbb{R}_B$ is *2-decomposable* [Jeavons *et al.*, 1998], that is every $n$-tuple $\mathbf{d}$ such that, for any $1 \le i, j \le n$, $(\mathbf{d}[i], \mathbf{d}[j]) \in \pi_{\{i,j\}}\varrho$, belongs to $\varrho$. Therefore, $\overline{\mathcal{S}}_A \cap \overline{\mathcal{S}}_B \neq \varnothing$ if and only if there is a tuple $\mathbf{d}$ such that $(\mathbf{d}[i], \mathbf{d}[j]) \in \pi_{\{i,j\}}(\overline{\mathcal{S}}_A) \cap \pi_{\{i,j\}}(\overline{\mathcal{S}}_B)$, for any $1 \le i, j \le q$.

The latter condition is encoded by the constraint satisfaction problem $(W; Z; \mathcal{C}'')$ in which $W = \{z_1, \dots, z_q\}$, and for any $1 \le i, j \le q$, there is the constraint $\langle (z_i, z_j), \pi_{\{i,j\}}(\overline{\mathcal{S}}_A) \cap \pi_{\{i,j\}}(\overline{\mathcal{S}}_B) \rangle$. Since all the constraint relations are in $\mathbb{R}_A \cup \mathbb{R}_B$, this problem can be solved in polynomial time (see [Jeavons *et al.*, 1998]). We just have to find relations of the form $\pi_{\{i,j\}}(\overline{\mathcal{S}}_A)$, $\pi_{\{i,j\}}(\overline{\mathcal{S}}_B)$. To this end we consider the problems

$$\begin{aligned}\widetilde{\mathcal{P}}_A^{i,j}(A, A) &= \widetilde{\mathcal{P}}_A \cup \{\langle (\overline{u}_{C_i}, \sigma_A \rangle, \langle \overline{u}_{C_j}, \sigma_A \rangle\}, \\ \widetilde{\mathcal{P}}_A^{i,j}(A, B) &= \widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_{C_i}, \sigma_A \rangle, \langle \overline{u}_{C_j}, \{\overline{c}\} \rangle\}, \\ \widetilde{\mathcal{P}}_A^{i,j}(B, A) &= \widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_{C_i}, \{\overline{c}\} \rangle, \langle \overline{u}_{C_j}, \sigma_A \rangle\}, \\ \widetilde{\mathcal{P}}_A^{i,j}(B, B) &= \widetilde{\mathcal{P}}_A \cup \{\langle \overline{u}_{C_i}, \{\overline{c}\} \rangle, \langle \overline{u}_{C_j}, \{\overline{c}\} \rangle\},\end{aligned}$$

and include a tuple $(X, Y)$ in $\pi_{\{i,j\}}(\overline{\mathcal{S}}_A)$ if and only if the problem $\widetilde{\mathcal{P}}_A^{i,j}(X, Y)$ has a solution. The relation $\pi_{\{i,j\}}(\overline{\mathcal{S}}_B)$ can be found in the same way.

CASE 4. $\mathbb{R}_A \cup \mathbb{R}_B$ is invariant under an affine operation $f$.

This case is, in fact, impossible. To show this, notice that the relation $\varrho = (\sigma_A \times A^{l_A}) \cup \{\overline{c}\}$ belongs to $R_A^+$. Then the relation

$$\tau = \begin{pmatrix} A & A & B \\ A & B & B \end{pmatrix}$$

is expressible by the pp-formula $P_\tau(x_1, x_2) = \exists y_1, y_2, z$ $\widetilde{P}_\varrho(y_1, y_2, z) \wedge \widetilde{P}_{\sigma_A}(y_1, x_1) \wedge \widetilde{P}_{\sigma_A}(y_2, x_2)$, and, therefore, belongs to $\mathbb{R}_A$. However, $f$ does not preserve $\tau$.

**Example 5** Let $X = \{0, 1, 2, 3\}$, $Y = \{\overline{0}, \overline{1}, \overline{2}, \overline{3}\}$, $D = \{c\}$. Consider the relations on $A = X \cup D$, $B = Y \cup D$:

$$\varrho_A = \begin{pmatrix} c & c & c & c & c & c & c & c & c & c & c & c \\ 0 & 0 & 2 & 2 & 1 & 1 & 3 & 3 & 0 & 1 & 2 & 3 \\ 1 & 3 & 1 & 3 & 0 & 2 & 0 & 2 & c & c & c & c \end{pmatrix},$$

$$\varrho_B = \begin{pmatrix} \overline{0} & \overline{0} & \overline{2} & \overline{2} & \overline{1} & \overline{1} & \overline{3} & \overline{3} & \overline{0} & \overline{0} & \overline{2} & \overline{2} & \overline{1} & \overline{1} & \overline{3} & \overline{3} & \overline{0} & \overline{1} & \overline{2} & \overline{3} \\ c & c & c & c & c & c & c & c & c & c & c & c & c & c & c & c & c & c & c & c \\ \overline{0} & \overline{2} & \overline{0} & \overline{2} & \overline{1} & \overline{3} & \overline{1} & \overline{3} & \overline{1} & \overline{3} & \overline{1} & \overline{3} & \overline{0} & \overline{2} & \overline{0} & \overline{2} & c & c & c & c \end{pmatrix}.$$

We are going to show that the amalgam $\{\varrho_A\} \dot{\bowtie} \{\varrho_B\}$ is tractable. To this end we notice that $\varrho_A$ $[\varrho_B]$ is invariant with respect to $f_A, h_A$ $[f_B, g_B, h_B]$ such that

- $f_A(x,y) = x$ if $x = c$, $f_A(x,y) = y$ if $x \in X$; $f_B(x,y) = x$ if $x \in Y$, $f_B(x,y) = y$ if $x = c$;
- $g_B(x,y) = x$ if $x = c$, $g_B(x,y) = y$ if $x \in Y$;
- $h_A(x,y,z) = x -_X y +_X z$ if $x, y, z \in X$, $h_A(x,y,z) = x$ otherwise;
- $h_B(x,y,z) = x -_Y y +_Y z$ if $x, y, z \in Y$, $h_A(x,y,z) = x$ otherwise

where $+_X$, $+_Y$ denote addition modulo 4 in $X, Y$ respectively. We define the relational clones $R_A, R_B$ to be $\mathsf{Inv}\,(\{f_A, h_A\})$, $\mathsf{Inv}\,(\{f_B, g_B, h_B\})$. Clearly, $\varrho_A \in R_A$, $\varrho_B \in R_B$. We show that $R_A \dot{\bowtie} R_B$ is tractable.

As is easily seen, the operations $f_A, h_A$ preserve the relation $X$, and $f_B, g_B, h_B$ preserve $Y$. Therefore, $X \in R_A \subseteq \overline{R}_A$, $Y \in R_B \subseteq \overline{R}_B$, hence, $R_A, R_B$ are not monolithic. It is straightforward to show that every relation from $\mathbb{R}_A \cup \mathbb{R}_B$ is invariant with respect to the semilattice operation $f$ where $f(A,B) = f(B,A) = f(B,B) = B$, $f(A,A) = A$.

By Schaefer's Theorem, $\mathbb{R}_A \cup \mathbb{R}_B$ is tractable, therefore, to show the tractability of $R_A \dot{\bowtie} R_B$ we have to show the tractability of $R_A^+, R_B^+$. For any $\varrho \in R_A$, the operations $f_A, h_A$ preserve the relation $\varrho \cup \{\overline{c}\}$, and, for any $\varrho \in R_B$, $f_B, g_B, h_B$ preserve $\varrho \cup \{\overline{c}\}$. Hence, $R_A^+ = R_A$, $R_B^+ = R_B$.

We sketch a solving algorithm for $\mathrm{CSP}(R_A)$ and leave to the reader verifying its soundness. Let $\mathcal{P} = (V; A; \mathcal{C}) \in \mathrm{CSP}(R_A)$.

- **Set** $W = \{v \in V \mid$ for any $\langle s, \varrho \rangle \in \mathcal{C}$ such that $v \in s$, $c \in \pi_{\{v\}}(\varrho)\}$, and $U = V - W$.
- Every constraint relation of $\mathcal{P}\big|_U = (U; X; \mathcal{C}\big|_U)$ where, for any $\langle s, \varrho \rangle \in \mathcal{C}$, there is $\langle s \cap U, \pi_{s \cap U}(\varrho) \rangle \in \mathcal{C}\big|_U$ is invariant with respect to $x -_X y +_X z$. **Find** a solution $\psi: U \to X$ to $\mathcal{P}\big|_U$ by making use of standard algorithms from number theory.
- **Set** $\varphi(v) = \psi(v)$ if $v \in U$ and $\varphi(v) = c$ otherwise. The mapping $\varphi: V \to A$ is a solution to $\mathcal{P}$.

The algorithm for $\mathrm{CSP}(R_B)$ is analogous.

## References


[Cohen *et al.*, 1997] D.A. Cohen, P.G. Jeavons, and M. Koubarakis. Tractable disjunctive constraints. In *Proceedings 3rd International Conference on Constraint Programming—CP'97 (Linz, October 1997)*, volume 1330 of *Lecture Notes in Computer Science*, pages 478–490. Springer-Verlag, 1997.

[Cohen *et al.*, 2000a] D.A. Cohen, P.G. Jeavons, and R.L. Gault. New tractable constraint classes from old. In *Proceedings 6th International Conference on Principles and Practice of Constraint Programming—CP'2000*, volume 1894 of *Lecture Notes in Computer Science*, pages 160–171. Springer-Verlag, 2000.

[Cohen *et al.*, 2000b] D.A. Cohen, P.G. Jeavons, P. Jonsson, and M. Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47:826–853, 2000.

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

[Dechter and Pearl, 1989] R. Dechter and J. Pearl. Tree clustering for constraint networks. *Artificial Intelligence*, 38:353–366, 1989.

[Dechter and van Beek, 1997] R. Dechter and P. van Beek. Local and global relational consistency. *Theoretical Computer Science*, 173(1):283–308, 1997.

[Freuder, 1990] E.C. Freuder. Complexity of $k$-tree structured constraint satisfaction problems. In *Proceedings of 8th National Conference on Artificial Intelligence AAAI-90*, pages 4–9, 1990.

[Gottlob *et al.*, 2000] G. Gottlob, L. Leone, and F. Scarcello. A comparison of structural CSP decomposition methods. *Artificial Intelligence*, 124(2):243–282, 2000.

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

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

[Jeavons, 1998a] P.G. Jeavons. Constructing constraints. In *Proceedings 4th International Conference on Constraint Programming—CP'98 (Pisa, October 1998)*, volume 1520 of *Lecture Notes in Computer Science*, pages 2–16. Springer-Verlag, 1998.

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

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

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

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