INSTITUT FÜR INFORMATIK
Lehr- und Forschungseinheit für
Programmier- und Modellierungssprachen
Oettingenstraße 67, D–80538 München

Ludwig
Maximilians
Universität
München

A Modular Approach To Proving Confluence

Michael Marte

http://www.pms.informatik.uni-muenchen.de/publikationen
Forschungsbericht/Research Report PMS-FB-2001-18, Release 12th May 2002

A Modular Approach To Proving Confluence

Michael Marte[*]
Institut für Informatik, Universität München
Oettingenstr. 67, 80538 München, Germany
`marte@informatik.uni-muenchen.de`

12th May 2002

Abstract

We are interested in investigating the confluence properties of cooperating constraint solvers. To this end, we model solvers as reductions that transform constraint networks, we define the notion of *insensitivity* to a superset relation, and show that, if each solver of a given set of solvers is *insensitive* to the same terminating superset relation, then any combination of these solvers is confluent. By means of this modular approach, we study the relationship between confluence and maintaining certain levels of local consistency and we demonstrate the confluence of a solver for a global finite-domain constraint that consists of several reductions.

1 Introduction

We are interested in investigating the confluence properties of cooperating constraint solvers. If a system of constraint solvers is confluent, then the result of constraint propagation does not depend on how the solvers are scheduled. If it is either known to diverge or if its neither known to be confluent nor to diverge, then the question arises which scheduling strategy will perform best. This may be very inconvenient in empirical research as well as in application development as it potentially adds another dimension to the design space.

To establish confluence properties, we model solvers as reductions that transform constraint networks, we define the notion of *insensitivity* to a superset relation, and show that, if each solver of a given set of solvers is *insensitive* to the same terminating superset relation, then any combination of these solvers is confluent.

As a first application of our approach, we study the relationship between confluence and maintaining certain levels of local consistency. In particular, we investigate domain and interval consistency as defined by van Hentenryck et al. [VSD98] in the context of finite-domain constraint solving. It turns out that any combination of solvers where each solver maintains either level of local consistency is confluent.

Second, we apply our approach to a solver for a global finite-domain constraint that has been designed to model and solve track parallelization problems. This kind of problem occurs in school timetabling and consists in parallelizing the execution of task sets [Mar01]. The solver consists of several reductions and we demonstrate its confluence and the confluence of any subset of its reductions with a number of proofs linear in the number of reductions.

[*]This work was supported by the German Research Council (DFG).

This paper is organized as follows. Section 2 introduces some terminology. Section 3 introduces the concept of insensitivity and relates it to the concept of strong commutation. Section 4 presents our method for proving confluence. In Section 5, we provide a reduction system that captures the process of solving finite-domain constraints. Section 6 studies the relationship between local consistency and confluence. In Section 7, we present our solver for track parallelization and investigate its confluence properties. In Section 8, we present related work and compare to it. Section 9 summarizes and closes with perspectives for future work.

2 Preliminaries

We start by remembering some concepts that are required to talk about reduction systems and that will be used throughout the paper (cf. [BN98]).

A *reduction system* is a pair $(A, \rightarrow)$ where A is a set and $\rightarrow \subseteq A \times A$. $\rightarrow^=$ denotes the reflexive closure of $\rightarrow$. $\rightarrow^+$ denotes the transitive closure of $\rightarrow$. $\rightarrow^*$ denotes the reflexive transitive closure of $\rightarrow$. x is called *reducible* iff $\exists y.\ x \rightarrow y$. x is called *in normal form (irreducible)* iff it is not reducible. y is called *a normal form of* x iff $x \rightarrow^* y$ and y is in normal form. If x has a uniquely determined normal form, the latter is denoted by $x\downarrow$. We say that y is a *direct successor* of x iff $x \rightarrow y$. We say that y is a *successor* of x iff $x \rightarrow^+ y$. $x, y \in A$ are called *joinable* iff $\exists z.\ x \rightarrow^* z \leftarrow^* y$. We write $x \downarrow y$ to denote that x and y are joinable. $\rightarrow$ is called *locally confluent* iff $y \leftarrow x \rightarrow z$ implies $y \downarrow z$. It is called *confluent* iff $y \leftarrow^* x \rightarrow^* z$ implies $y \downarrow z$. $\rightarrow$ is called *terminating* iff there is no chain $a_0 \rightarrow a_1 \rightarrow \ldots$ that descends infinitely. It is called *convergent* iff it is terminating and confluent. Let $(A, \rightarrow_1)$ and $(A, \rightarrow_2)$ be reduction systems. We say that $\rightarrow_1$ and $\rightarrow_2$ *commute* iff $y \leftarrow_1^* x \rightarrow_2^* z$ implies $\exists u.\ y \rightarrow_2^* u \leftarrow_1^* z$. We say that $\rightarrow_1$ and $\rightarrow_2$ *commute strongly* iff $y \leftarrow_1 x \rightarrow_2 z$ implies $\exists u.\ y \rightarrow_2^= u \leftarrow_1^* z$.

3 Insensitivity

Intuitively, $\rightarrow_1$ is insensitive to $\rightarrow_2$, if the inference capabilities of $\rightarrow_1$ are preserved under application of $\rightarrow_2$.

Definition 1. Let $(A, \rightarrow_1)$ and $(A, \rightarrow_2)$ be reduction systems. We say that $\rightarrow_1$ *is insensitive to* $\rightarrow_2$ iff the following requirements are satisfied.

1. If $y \leftarrow_1 x \rightarrow_2 z$, $y \neq z$, and $z \rightarrow_2 y$, then $z \rightarrow_1 y$.

$$\begin{tikzcd}
x \arrow[r, "1"] \arrow[dr, "2"'] & y & & x \arrow[r, "1"] \arrow[dr, "2"'] & y \\
& z \arrow[u, "2"'] & \Rightarrow & & z \arrow[u, "1"']
\end{tikzcd}$$

2. If $y \leftarrow_1 x \rightarrow_2 z$, $y \neq z$, $y \not\rightarrow_2 z$, and $z \not\rightarrow_2 y$, then $u \in A$ exists s.t. $y \rightarrow_2 u \leftarrow_1 z$.

$$\begin{tikzcd}
x \arrow[r, "1"] \arrow[dr, "2"'] & y & & x \arrow[r, "1"] \arrow[dr, "2"'] & y \arrow[r, "2"] & u \\
& z & \Rightarrow & & z \arrow[ur, "1"'] &
\end{tikzcd}$$

Corollary 1. *If $\rightarrow_1$ and $\rightarrow_2$ are insensitive to $\rightarrow_3$, then $\rightarrow_1 \cup \rightarrow_2$ is insensitive to $\rightarrow_3$.*

Corollary 2. *Let $(A, \to)$ be a reduction system. If $\to$ is insensitive to itself, then it is locally confluent.*

The following propositions show that the concepts of insensitivity and strong commutation are related but not equivalent.

Proposition 1. *Let $(A, \to_1)$ and $(A, \to_2)$ be reduction systems. If $\to_1$ is insensitive to $\to_2$, then $\to_1$ and $\to_2$ commute strongly.*

Proof. We have to show that $y \leftarrow_1 x \to_2 z$ implies $\exists u.\, y \to_2^= u \leftarrow_1^* z$. If $y = z$, we are done. Otherwise, there are three cases. If $y \to_2 z$, we are done. If $z \to_2 y$, then $z \to_1 y$ because $\to_1$ is insensitive to $\to_2$. If neither $y \to_2 z$ nor $z \to_2 y$, then $u \in A$ exists s.t. $y \to_2 u \leftarrow_1 z$ because $\to_1$ is insensitive to $\to_2$. ☐

Proposition 2. *Let $(A, \to_2)$ be a reduction system and let $\to_1 \subseteq \to_2$ be a transitive reduction s.t. $\to_1$ and $\to_2$ commute strongly. If $y \leftarrow_1 x \to_2 z$, $y \neq z$, $y \not\to_2 z$, and $z \not\to_2 y$, then $u \in A$ exists s.t. $y \to_2 u \leftarrow_1 z$.*

Proof. By strong commutation, we know that $u \in A$ exists s.t. $y \to_2^= u \leftarrow_1^* z$. Suppose $u = z$. Then $y \to_2^= z$. Because $y \neq z$, $y \to_2 z$. This contradicts the premise and thus $z \to_1^+ u$. Because $\to_1$ is transitive, $z \to_1 u$. Suppose $u = y$. Then $z \to_1 y$ and thus $z \to_2 y$ because $\to_1 \subseteq \to_2$. This contradicts the premise and thus $y \to_2 u$. ☐

4 Confluence Through Insensitivity

Theorem 1. *Let $(A, \to_2)$ be a terminating reduction system. If $\to_1 \subseteq \to_2$ is insensitive to $\to_2$, then $\to_1$ is locally confluent.*

Proof. For each pair $(y,z) \in A \times A$ s.t. $\exists x.\, y \leftarrow_1 x \to_2 z$, $\exists x.\, y \leftarrow_2 x \to_1 z$, $y \neq z$, $y \not\to_2 z$, and $z \not\to_2 y$, choose a pair $(\hat{y}, \hat{z}) \in A \times A$ s.t. $y \to_2 \hat{y} \leftarrow_1 z$ and $z \to_2 \hat{z} \leftarrow_1 y$. This is possible because $\to_1$ is insensitive to $\to_2$.

Let $(x_2, y_2) \prec (x_1, y_1)$ iff $x_1 \to_2 x_2$. $\prec$ is well-founded because $\to_2$ is terminating. By well-founded recursion on $\prec$, we define $s(y,z)$ for all (y,z) that satisfy $\exists x.\, y \leftarrow_1 x \to_2 z$ and $\exists x.\, y \leftarrow_2 x \to_1 z$:

$$s(y,z) = \begin{cases} (y,z), & \text{if } y = z,\ y \to_2 z, \text{ or } z \to_2 y \\ (y,z), s(\hat{y},\hat{z}) & \text{otherwise} \end{cases}$$

In the second case, $s(\hat{y},\hat{z})$ is well-defined because $(\hat{y},\hat{z})$ exists, $\hat{y} \leftarrow_1 z \to_2 \hat{z}$, and $\hat{y} \leftarrow_2 y \to_1 \hat{z}$. Since s is defined by well-founded recursion on $\prec$, $s(y,z)$ is finite for all (y,z) that s is defined for.

Let $y_1 \leftarrow_1 x \to_1 z_1$. $s(y_1, z_1)$ is well-defined because $\to_1 \subseteq \to_2$ and thus $y_1 \leftarrow_1 x \to_2 z_1$ and $y_1 \leftarrow_2 x \to_1 z_1$. Let $n > 0$ s.t. $s(y_1,z_1) = (y_1,z_1), \dots, (y_n,z_n)$. We observe that, for all $1 \leq k < n$, $y_k \to_2 y_{k+1} \leftarrow_1 z_k$ and $z_k \to_2 z_{k+1} \leftarrow_1 y_k$, and that $y_n = z_n$, $y_n \to_2 z_n$, or $y_n \leftarrow_2 z_n$. The following figure shows a situation where $n \geq 5$.

$$\begin{tikzcd}
x \arrow[r, "1"] \arrow[dr, "1"'] & y_1 \arrow[r, "2"] \arrow[dr, "1"] & y_2 \arrow[r, "2"] \arrow[dr, "1"] & y_3 \arrow[r, "2", "+"'] & y_{n-1} \arrow[r, "2"] \arrow[dr, "1"] & y_n \arrow[d, <->, bend left, "2"', "="] \\
 & z_1 \arrow[r, "2"'] \arrow[ur, "1"'] & z_2 \arrow[r, "2"'] \arrow[ur, "1"'] & z_3 \arrow[r, "2"', "+"] & z_{n-1} \arrow[r, "2"'] \arrow[ur, "1"'] & z_n
\end{tikzcd}$$

It remains to show that $y_1 \downarrow_1 z_1$. If n is odd, then $y_1 \rightarrow_1^* y_n$ and $z_1 \rightarrow_1^* z_n$. If n is even, then $y_1 \rightarrow_1^* z_n$ and $z_1 \rightarrow_1^* y_n$. If $y_n = z_n$, we are done. If $y_n \rightarrow_2 z_n$, then $y_n \rightarrow_1 z_n$ because $y_n \leftarrow_2 y_{n-1} \rightarrow_1 z_n$ and $\rightarrow_1$ is insensitive to $\rightarrow_2$. If $z_n \rightarrow_2 y_n$, then $z_n \rightarrow_1 y_n$ because $z_n \leftarrow_2 z_{n-1} \rightarrow_1 y_n$ and $\rightarrow_1$ is insensitive to $\rightarrow_2$. $\square$

The following result is obtained by applying Newman's Lemma. Newman's Lemma states that a terminating reduction is confluent iff it is locally confluent.

Corollary 3. *Let $(A, \rightarrow_2)$ be a terminating reduction system. If $\rightarrow_1 \subseteq \rightarrow_2$ is insensitive to $\rightarrow_2$, then $\rightarrow_1$ is confluent.*

5 A Model of Finite-Domain Constraint Solving

To apply our method to finite-domain (FD) constraint solvers, it is necessary to provide a reduction system that captures the process of solving FD constraints. We use a reduction system where reduction steps transform finite constraint networks by pruning values from domains. Neither the addition nor the removal of variables and constraints is supported.

On the conceptual level, a *finite constraint network* (FCN) is a finite hypergraph with variables as nodes and constraints as hyperarcs. Given a FCN, the corresponding *finite constraint satisfaction problem* (FCSP) consists in finding a variable valuation that satisfies all the constraints. We will not distinguish between a FCN and its FCSP.

Let P be a FCSP with variables X and constraints C. We assume that there is a unary constraint for each variable that specifies its set of admissible values. P will be represented by a triple (X, δ, C) where δ is a total function on X (the *domain function* of P) that associates each variable with its set of admissible values (its *domain*). We say that P is *ground* iff all its variables have singleton domains. We say that P is *failed* iff at least one of its variables has an empty domain. We use $\text{scope}(P)$ and $\text{store}(P)$ to denote the variables and constraints of P, respectively. If c is a constraint, $\text{scope}(c)$ denotes the set of variables constrained by c.

Frequently, we will refer to domain functions that have not been declared explicitly. However, in such a case, there will a FCSP the domain function belongs to according to the following naming scheme: If P, P_i, R, R_i, Γ, and Σ denote FCSPs, then δ, δ_i, ρ, ρ_i, γ, and σ are their respective domain functions.

We will consider FCSPs with integer domains only. If a and b are integers, we write $[a, b]$ to denote the set of integers i with $a \leq i \leq b$.

Definition 2. Let $P_0 = (X_0, \delta_0, C_0)$ and $P_1 = (X_1, \delta_1, C_1)$ be FCSPs.

1. $P_0 \rightarrow_{\text{FD}} P_1$ iff $X_1 = X_0$, $C_1 = C_0$, $\delta_1 \neq \delta_0$, and $\delta_1(x) \subseteq \delta_0(x)$ for all $x \in X_0$.

2. $P_1 \in \text{gs}(P_0)$ (P_1 *is a ground successor of* P_0) iff $P_0 \rightarrow_{\text{FD}} P_1$ and P_1 is ground.

3. $P_1 \in \text{sol}(P_0)$ (P_1 *solves* P_0) iff $P_1 \in \text{gs}(P_0)$ and δ_1 simultaneously satisfies all $c \in C$.

4. $P_0 \equiv P_1$ (P_0 *and* P_1 *are equivalent*) iff $\text{sol}(P_0) = \text{sol}(P_1)$.

Corollary 4. *$\rightarrow_{\text{FD}}$ is strict and convergent.*

Corollary 5. *Let $P_0 = (X, \delta_0, C) \rightarrow_{\text{FD}} P_1$.*

1. *Lower (Upper) bounds of domains grow (shrink) monotonically, i.e.*
   $\min \delta_0(x) \leq \min \delta_1(x) \leq \max \delta_1(x) \leq \max \delta_0(x)$ *for all $x \in X$.*

2. *Sets of ground successors shrink monotonically, i.e.* $\mathrm{gs}(P_0) \supseteq \mathrm{gs}(P_1)$.

3. *Solution sets shrink monotonically, i.e.* $\mathrm{sol}(P_0) \supseteq \mathrm{sol}(P_1)$.

Definition 3. $\rightarrow_r \subseteq \rightarrow_{\mathrm{FD}}$ is called *correct* iff, for all $P_0 \rightarrow_r P_1$, $P_0 \equiv P_1$.

Definition 4. $\rightarrow_{\mathrm{C}} = \bigcup \{\rightarrow_r \subseteq \rightarrow_{\mathrm{FD}} : \rightarrow_r \text{ is correct}\}$

$\rightarrow_{\mathrm{FD}}$ allows for arbitrary domain reductions while $\rightarrow_{\mathrm{C}} \subseteq \rightarrow_{\mathrm{FD}}$ only allows for domain reductions that preserve solutions.

Corollary 6. $\rightarrow_{\mathrm{C}}$ *is terminating and correct.*

Lemma 1. *Let* $\rightarrow_r \subseteq \rightarrow_{\mathrm{FD}}$. *If* $\rightarrow_r$ *is correct and insensitive to* $\rightarrow_{\mathrm{FD}}$, *then it is insensitive to* $\rightarrow_{\mathrm{C}}$.

Proof. Let $P_1 \leftarrow_r P_0 \rightarrow_{\mathrm{C}} P_2$ s.t. $P_2 \rightarrow_{\mathrm{C}} P_1$. $P_2 \rightarrow_r P_1$ because $\rightarrow_{\mathrm{C}} \subseteq \rightarrow_{\mathrm{FD}}$ and $\rightarrow_r$ is insensitive to $\rightarrow_{\mathrm{FD}}$.

Let $P_1 \leftarrow_r P_0 \rightarrow_{\mathrm{C}} P_2$ s.t. $P_1 \neq P_2$, $P_1 \not\rightarrow_{\mathrm{C}} P_2$, and $P_2 \not\rightarrow_{\mathrm{C}} P_1$. We have to show that P_3 exists s.t. $P_1 \rightarrow_{\mathrm{C}} P_3 \leftarrow_r P_2$. By Definitions 3 and 4, $P_1 \not\rightarrow_{\mathrm{C}} P_2$ iff $P_1 \not\rightarrow_{\mathrm{FD}} P_2$ or $\mathrm{sol}(P_1) \neq \mathrm{sol}(P_2)$. However, $\mathrm{sol}(P_1) = \mathrm{sol}(P_0) = \mathrm{sol}(P_2)$ because $P_1 \leftarrow_r P_0 \rightarrow_{\mathrm{C}} P_2$ and both $\rightarrow_r$ and $\rightarrow_{\mathrm{C}}$ are correct. Hence $P_1 \not\rightarrow_{\mathrm{FD}} P_2$. By a symmetric argument, $P_2 \not\rightarrow_{\mathrm{FD}} P_1$. Furthermore, $P_0 \rightarrow_{\mathrm{FD}} P_2$ because $\rightarrow_{\mathrm{C}} \subseteq \rightarrow_{\mathrm{FD}}$. By the insensitivity of $\rightarrow_r$ to $\rightarrow_{\mathrm{FD}}$, P_3 exists s.t. $P_1 \rightarrow_{\mathrm{FD}} P_3 \leftarrow_r P_2$. Finally, $\mathrm{sol}(P_3) = \mathrm{sol}(P_2)$ because $P_2 \rightarrow_r P_3$ and $\rightarrow_r$ is correct. In consequence, $\mathrm{sol}(P_1) = \mathrm{sol}(P_3)$ and thus $P_1 \rightarrow_{\mathrm{C}} P_3$.

We conclude that $\rightarrow_r$ is insensitive to $\rightarrow_{\mathrm{C}}$. □

Lemma 2. *Let* $P_0 = (X, \delta_0, C) \rightarrow_{\mathrm{FD}} P_1$ *and* $Y = \{y_1, \ldots, y_n\} \subseteq X$ *s.t.* $\delta_1(x) = \delta_0(x)$ *for all* $x \in X - Y$.

1. *If* $P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{FD}} P_1$, *then*

$$\delta_1(y_1) \times \ldots \times \delta_1(y_n) \subset \delta_2(y_1) \times \ldots \times \delta_2(y_n) \subset \delta_0(y_1) \times \ldots \times \delta_0(y_n).$$

2. *If* $P_0 \rightarrow_{\mathrm{FD}} P_2$, $P_1 \not\rightarrow_{\mathrm{FD}} P_2$, $P_2 \not\rightarrow_{\mathrm{FD}} P_1$, *and* $P_1 \neq P_2$, *then*

$$\delta_2(y_1) \times \ldots \times \delta_2(y_n) \not\subseteq \delta_1(y_1) \times \ldots \times \delta_1(y_n).$$

3. *If* $P_0 \rightarrow_{\mathrm{C}} P_2$, $P_1 \not\rightarrow_{\mathrm{C}} P_2$, $P_2 \not\rightarrow_{\mathrm{C}} P_1$, *and* $P_1 \neq P_2$, *then*

$$\delta_2(y_1) \times \ldots \times \delta_2(y_n) \not\subseteq \delta_1(y_1) \times \ldots \times \delta_1(y_n).$$

Proof.

1. Let $x \in X - Y$. By Definition 2, $\delta_1(x) \subseteq \delta_2(x) \subseteq \delta_0(x)$. Considering that $\delta_1(x) = \delta_0(x)$, we obtain $\delta_1(x) = \delta_2(x) = \delta_0(x)$. In consequence, the variables in Y are the only variables the domains of which may be reduced in the course of $P_0 \rightarrow_{\mathrm{FD}} P_2$. Taking into account that some reduction has to take place in the course of $P_0 \rightarrow_{\mathrm{FD}} P_2$, we end up with

$$\delta_2(y_1) \times \ldots \times \delta_2(y_n) \subset \delta_0(y_1) \times \ldots \times \delta_0(y_n).$$

By a similar argument,

$$\delta_1(y_1) \times \ldots \times \delta_1(y_n) \subset \delta_2(y_1) \times \ldots \times \delta_2(y_n).$$

2. Let $Z = X - Y$. We start by noting that $\delta_2(x) \subseteq \delta_0(x) = \delta_1(x)$ for all $x \in Z$. Now suppose

$$\delta_2(y_1) \times \ldots \times \delta_2(y_n) \subseteq \delta_1(y_1) \times \ldots \times \delta_1(y_n).$$

Then either $\delta_2(y_i) = \delta_1(y_i)$ for all $1 \le i \le n$ or $1 \le i \le n$ exists s.t. $\delta_2(y_i) \subset \delta_1(y_i)$.

Suppose $\delta_2(y_i) = \delta_1(y_i)$ for all $1 \le i \le n$. If $\delta_2(x) = \delta_1(x)$ for all $x \in Z$, then $P_2 = P_1$. If $x \in Z$ exists s.t. $\delta_2(x) \subset \delta_1(x)$, then $P_1 \rightarrow_{\text{FD}} P_2$.

Suppose $1 \le i \le n$ exists s.t. $\delta_2(y_i) \subset \delta_1(y_i)$. Then $P_1 \rightarrow_{\text{FD}} P_2$ because $\delta_2(x) \subseteq \delta_1(x)$ for all $x \in Z$.

3. By Definitions 3 and 4, $P_1 \not\rightarrow_{\text{C}} P_2$ iff $P_1 \not\rightarrow_{\text{FD}} P_2$ or $\text{sol}(P_1) \neq \text{sol}(P_2)$. However, $\text{sol}(P_1) = \text{sol}(P_0) = \text{sol}(P_2)$ because $P_1 \leftarrow_{\text{C}} P_0 \rightarrow_{\text{C}} P_2$ and $\rightarrow_{\text{C}}$ is correct. Hence $P_1 \not\rightarrow_{\text{FD}} P_2$. By a symmetric argument, $P_2 \not\rightarrow_{\text{FD}} P_1$. Thus, by (2),

$$\delta_2(y_1) \times \ldots \times \delta_2(y_n) \not\subseteq \delta_1(y_1) \times \ldots \times \delta_1(y_n).$$

□

Lemma 3. *If* $P_0 \rightarrow_{\text{C}} P_1$ *and* $P_0 \rightarrow^{=}_{\text{FD}} P_2 \rightarrow^{=}_{\text{FD}} P_1$, *then* $\text{sol}(P_0) = \text{sol}(P_2) = \text{sol}(P_1)$.

Proof. By Corollary 5, $\text{sol}(P_0) \subseteq \text{sol}(P_2) \subseteq \text{sol}(P_1)$ and, by the correctness of $\rightarrow_{\text{C}}$, $\text{sol}(P_0) = \text{sol}(P_1)$. □

6 Insensitivity Through Local Consistency

Suppose a constraint solver is known to maintain a certain level of local consistency. Then we have a performance guarantee: Whatever input state the solver is applied to, the output state will satisfy an invariant specific to the level of local consistency maintained. The question arises whether this specific guarantee entails insensitivity to some suitable superset relation.

In particular, we consider domain and interval consistency [VSD98] as both notions are important in FD constraint solving.

- The language `cc(FD)` [VSD98] allows for arithmetic constraints over integer variables and, for each relation, it can maintain domain or interval consistency, as required by the user.

- AC-4 and GAC-4 [Tsa93] are well-known algorithms to maintain domain consistency. AC-4 applies only to binary constraints while GAC-4 is not restricted wrt. to the arity of constraints.

- An alldiff constraint [vH01] specifies that all its variables must take pairwise distinct values. alldiff constraints are ubiquitous in timetabling and scheduling; they are used to model disjunctive scheduling problems where tasks have unit duration. Mehlhorn & Thiel [MT00] present an efficient algorithm to maintain interval consistency for alldiff constraints. Régin [Rég94] proposes an efficient algorithm to maintain domain consistency.

- For each value, a global cardinality constraint (gcc) imposes bounds on the number of variables that the value may be assigned to. The gcc generalizes the alldiff constraint; it has applications in timetabling and scheduling. Régin [Rég96] presents an efficient algorithm to maintain domain consistency for this type of constraint.

Indeed, as we show in the following, if a solver maintains domain or interval consistency, then it is insensitive to $\rightarrow_{\mathrm{FD}}$. In consequence, any combination of solvers where each solver maintains either level of local consistency is confluent.

6.1 Domain Consistency

Suppose $P = (X, \delta, C)$ is a FCSP and $c = p(x_1, \ldots, x_n) \in C$. According to van Hentenryck et al. [VSD98], c is *domain-consistent*, if, for each variable x_i and for each value $v_i \in \delta(x_i)$, there exist values $v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n$ in $\delta(x_1), \ldots, \delta(x_{i-1}), \delta(x_{i+1}), \ldots, \delta(x_n)$ s.t. $p(v_1, \ldots, v_n)$ holds.

Suppose c is a constraint. For each $x \in \mathrm{scope}(c)$, we define the transition $\rightarrow_{\mathrm{DC}(c,x)}$ (cf. Definition 5). This transition eliminates exactly those values from the domain of x that do not have support according to the definition of c. To be more precise, if $P_0 \rightarrow_{\mathrm{DC}(c,x)} P_1$, then $\delta_1(x) \subset \delta_0(x)$, each value in $\delta_1(x)$ occurs in a solution to the relaxation $P_0|_{\{c\}} = (X, \delta_0, \{c\})$ of P_0, and no predecessor of P_1 wrt. $\rightarrow_{\mathrm{FD}}$ satisfies these conditions. We proceed by showing that $\rightarrow_{\mathrm{DC}(c,x)}$ is correct and insensitive to $\rightarrow_{\mathrm{FD}}$. In consequence, $\rightarrow_{\mathrm{DC}(c)} = \bigcup_{x \in \mathrm{scope}(c)} \rightarrow_{\mathrm{DC}(c,x)}$ is correct and insensitive to $\rightarrow_{\mathrm{FD}}$. Quite obviously, $\rightarrow_{\mathrm{DC}(c)}$ maintains domain consistency for c.

Corollary 7. *If $P_0 \rightarrow_{\mathrm{FD}} P_1$ and $c \in \mathrm{store}(P_0)$, then $P_0|_{\{c\}} \rightarrow_{\mathrm{FD}} P_1|_{\{c\}}$.*

Definition 5. We say that $P_0 \rightarrow_{\mathrm{DC}(c,x)} P_1$ iff $P_0 \rightarrow_{\mathrm{FD}} P_1$, $c \in \mathrm{store}(P_0)$, $x \in \mathrm{scope}(c)$, and $\delta_1 = \delta_0$ except for

$$\delta_1(x) = \left\{a \in \delta_0(x) : \exists \Sigma \in \mathrm{sol}(P_0|_{\{c\}}).\, x\sigma = a\right\}.$$

Proposition 3. *Let c be a constraint and $x \in \mathrm{scope}(c)$. $\rightarrow_{\mathrm{DC}(c,x)}$ is correct.*

Proof. Suppose $P_0 \rightarrow_{\mathrm{DC}(c,x)} P_1$ and let $R = P_0|_{\{c\}}$. By Definition 5,

$$\delta_1(x) = \{a \in \delta_0(x) : \exists \Sigma \in \mathrm{sol}(R).\, x\sigma = a\}.$$

We have to show that $\mathrm{sol}(P_0) = \mathrm{sol}(P_1)$. By Corollary 5, $\mathrm{sol}(P_1) \subseteq \mathrm{sol}(P_0)$ because $P_0 \rightarrow_{\mathrm{FD}} P_1$. To show that $\mathrm{sol}(P_0) \subseteq \mathrm{sol}(P_1)$, let $\Sigma \in \mathrm{sol}(P_0)$ and $a = x\sigma$. Obviously,

$$\mathrm{sol}(P_1) = \{\Sigma \in \mathrm{sol}(P_0) : x\sigma \in \delta_1(x)\}.$$

Suppose $\Sigma \notin \mathrm{sol}(P_1)$, or equivalently, $a \notin \delta_1(x)$. In this case, $x\sigma \neq a$ for all $\Sigma \in \mathrm{sol}(R)$. Since R is a relaxation of P_0, $\mathrm{sol}(P_0) \subseteq \mathrm{sol}(R)$ and thus $x\sigma \neq a$. □

Lemma 4. *If $P_0 \rightarrow_{\mathrm{DC}(c,x)} P_1$, then $P_0|_{\{c\}} \rightarrow_{\mathrm{DC}(c,x)} P_1|_{\{c\}}$.*

Proof. Let $R = P_0|_{\{c\}}$. By Definition 5,

$$\delta_1(x) = \{a \in \delta_0(x) : \exists \Sigma \in \mathrm{sol}(R).\, x\sigma = a\}$$

We have to show that

$$\delta_1(x) = \left\{a \in \delta_0(x) : \exists \Sigma \in \mathrm{sol}(R|_{\{c\}}).\, x\sigma = a\right\}.$$

This follows immediately from the observation that $R = R|_{\{c\}}$. □

Proposition 4. *If $P_0 \rightarrow_{\mathrm{DC}(c,x)} P_1$ and $P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{FD}} P_1$, then $P_2 \rightarrow_{\mathrm{DC}(c,x)} P_1$.*

Proof. Let $R_i = P_i|_{\{c\}}$. By Definition 5,

$$\delta_1(x) = \{a \in \delta_0(x) : \exists \Sigma \in \mathrm{sol}(R_0).\, x\sigma = a\}$$

By Lemma 4, $R_0 \rightarrow_{\mathrm{DC}(c,x)} R_1$ and, by Corollary 7, $R_0 \rightarrow_{\mathrm{FD}} R_2 \rightarrow_{\mathrm{FD}} R_1$. Hence, by Lemma 3, $\mathrm{sol}(R_0) = \mathrm{sol}(R_2)$. Furthermore, by Lemma 2, $\delta_1(x) \subset \delta_2(x) \subset \delta_0(x)$. As a simple consequence from these facts,

$$\delta_1(x) = \{a \in \delta_2(x) : \exists \Sigma \in \mathrm{sol}(R_2).\, x\sigma = a\}.$$

and thus, by Definition 5, $P_2 \rightarrow_{\mathrm{DC}(c,x)} P_1$. $\square$

Proposition 5. *If $P_0 \rightarrow_{\mathrm{DC}(c,x)} P_1$, $P_0 \rightarrow_{\mathrm{FD}} P_2$, $P_1 \not\rightarrow_{\mathrm{FD}} P_2$, $P_2 \not\rightarrow_{\mathrm{FD}} P_1$, and $P_1 \neq P_2$, then a FCSP P_3 exists s.t. $P_1 \rightarrow_{\mathrm{FD}} P_3$ and $P_2 \rightarrow_{\mathrm{DC}(c,x)} P_3$.*

Proof. Let $R_i = P_i|_{\{c\}}$. By Definition 5,

$$\delta_1(x) = \{a \in \delta_0(x) : \exists \Sigma \in \mathrm{sol}(R_0).\, x\sigma = a\}$$

Let $X = \mathrm{scope}(P_0)$, $C = \mathrm{store}(P_0)$, and $P_3 = (X, \delta_3, C)$ with $\delta_3 = \delta_2$ except for

$$\delta_3(x) = \{a \in \delta_2(x) : \exists \Sigma \in \mathrm{sol}(R_2).\, x\sigma = a\}.$$

We note that, by Corollary 7, $R_0 \rightarrow_{\mathrm{FD}} R_2$ and thus, by Corollary 5, $\mathrm{sol}(R_2) \subseteq \mathrm{sol}(R_0)$.

$P_2 \rightarrow_{\mathrm{FD}} P_3$: By Lemma 2, $\delta_2(x) \not\subseteq \delta_1(x)$, or equivalently, $a \in \delta_0(x)$ exists s.t. $a \in \delta_2(x)$ and $a \notin \delta_1(x)$. By $a \notin \delta_1(x)$, $x\sigma \neq a$ for all $\Sigma \in \mathrm{sol}(R_0)$. Since $\mathrm{sol}(R_2) \subseteq \mathrm{sol}(R_0)$, $a \notin \delta_3(x)$ and thus $\delta_3(x) \subset \delta_2(x)$.

$P_1 \rightarrow_{\mathrm{FD}} P_3$: $P_3 \neq P_1$ because otherwise $P_2 \rightarrow_{\mathrm{FD}} P_1$. For all $y \in X - \{x\}$, $\delta_1(y) = \delta_0(y) \supseteq \delta_2(y) = \delta_3(y)$ because $P_1 \leftarrow_{\mathrm{DC}(c,x)} P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{DC}(c,x)} P_3$. Suppose $\delta_3(x) \not\subseteq \delta_1(x)$, or equivalently, $a \in \delta_0(x)$ exists s.t. $a \in \delta_3(x)$ and $a \notin \delta_1(x)$. By $a \notin \delta_1(x)$, $x\sigma \neq a$ for all $\Sigma \in \mathrm{sol}(R_0)$. Since $\mathrm{sol}(R_2) \subseteq \mathrm{sol}(R_0)$, $a \notin \delta_3(x)$. $\square$

Corollary 8. *Let c be a constraint and $x \in \mathrm{scope}(c)$. $\rightarrow_{\mathrm{DC}(c,x)}$ is insensitive to $\rightarrow_{\mathrm{FD}}$.*

6.2 Interval Consistency

Suppose $P = (X, \delta, C)$ is a FCSP and $c = p(x_1, \ldots, x_n) \in C$. Let

$$B(x) = \begin{cases} \emptyset, & \text{if } \delta(x) = \emptyset, \\ \{\min \delta(x), \max \delta(x)\} & \text{otherwise.} \end{cases}$$

Let

$$\rho(x) = \begin{cases} \emptyset, & \text{if } \delta(x) = \emptyset, \\ [\min \delta(x), \max \delta(x)] & \text{otherwise.} \end{cases}$$

We say that c is *interval-consistent*, if, for each variable x_i and for each value v_i of $B(x_i)$, there exist values $v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n$ in $\rho(x_1), \ldots, \rho(x_{i-1}), \rho(x_{i+1}), \ldots, \rho(x_n)$ s.t. $p(v_1, \ldots, v_n)$ holds. This definition coincides with the original definition by van Hentenryck et al. [VSD98] except for that it also applies to constraints with a variable that has an empty domain.

Suppose c is a constraint. For each $x \in \mathrm{scope}(c)$, we define the transition $\rightarrow_{\mathrm{IC}(c,x)}$ (cf. Definitions 6 and 7). This transition tightens the bounds of the domain of x according to the definition of c. To be more precise, if $P_0 \rightarrow_{\mathrm{IC}(c,x)} P_1$, then $\delta_1(x) \subset \delta_0(x)$, both $\min \delta_1(x)$ and $\max \delta_1(x)$ occur in solutions to the relaxation $\mathrm{ir}(P_0|_{\{c\}}, x)$ of P_0 that does not touch the domain of x but relaxes all other domains according to [VSD98], and no predecessor of P_1 wrt. $\rightarrow_{\mathrm{FD}}$ satisfies these conditions. We proceed by showing that $\rightarrow_{\mathrm{IC}(c,x)}$ is correct and insensitive to $\rightarrow_{\mathrm{FD}}$. In consequence, $\rightarrow_{\mathrm{IC}(c)}=\bigcup_{x\in\mathrm{scope}(c)}\rightarrow_{\mathrm{IC}(c,x)}$ is correct and insensitive to $\rightarrow_{\mathrm{FD}}$. Quite obviously, $\rightarrow_{\mathrm{IC}(c)}$ maintains interval consistency for c.

Definition 6. Suppose $P=(X,\delta,C)$ is a FCSP with $x\in X$. Then $\mathrm{ir}(P,x)=(X,\rho,C)$ with

$$\rho(y)=\begin{cases}\delta(y), & \text{if } y=x \text{ or } \delta(y)=\emptyset\\ [\min\delta(y),\max\delta(y)] & \text{otherwise}.\end{cases}$$

Corollary 9. *Suppose $P_0=(X,\delta_0,C)\rightarrow_{\mathrm{FD}}P_1$.*

1. *If $x\in X$, then $\mathrm{ir}(P_0,x)\rightarrow^{=}_{\mathrm{FD}}\mathrm{ir}(P_1,x)$.*
2. *If $x\in X$ and $c\in C$, then $\mathrm{ir}(P_0|_{\{c\}},x)\rightarrow^{=}_{\mathrm{FD}}\mathrm{ir}(P_1|_{\{c\}},x)$.*

Definition 7. We say that $P_0\rightarrow_{\mathrm{IC}(c,x)}P_1$ iff $P_0\rightarrow_{\mathrm{FD}}P_1$, $c\in\mathrm{store}(P_0)$, $x\in\mathrm{scope}(c)$, and $\delta_1=\delta_0$ except for

$$\delta_1(x)=\begin{cases}\emptyset, & \text{if } S=\emptyset\\ \{a\in\delta_0(x):l\le a\le u\} & \text{otherwise}\end{cases}$$

where $S=\mathrm{sol}(\mathrm{ir}(P_0|_{\{c\}},x))$, $l=\min\{x\sigma:\Sigma\in S\}$, and $u=\max\{x\sigma:\Sigma\in S\}$.

Proposition 6. *Let c be a constraint and $x\in\mathrm{scope}(c)$. $\rightarrow_{\mathrm{IC}(c,x)}$ is correct.*

Proof. Suppose $P_0\rightarrow_{\mathrm{IC}(c,x)}P_1$. By Definition 7,

$$\delta_1(x)=\begin{cases}\emptyset, & \text{if } S=\emptyset\\ \{a\in\delta_0(x):l\le a\le u\} & \text{otherwise}\end{cases}$$

where $S=\mathrm{sol}(\mathrm{ir}(P_0|_{\{c\}},x))$, $l=\min\{x\sigma:\Sigma\in S\}$, and $u=\max\{x\sigma:\Sigma\in S\}$. We have to show that $\mathrm{sol}(P_0)=\mathrm{sol}(P_1)$.

Suppose $S=\emptyset$. Then $\delta_1(x)=\emptyset$ and thus $\mathrm{sol}(P_1)=\emptyset$. Since R is a relaxation of P_0, $\mathrm{sol}(P_0)\subseteq\mathrm{sol}(R)=S=\emptyset$ and thus $\mathrm{sol}(P_0)=\emptyset$.

Suppose $S\neq\emptyset$. By Corollary 5, $\mathrm{sol}(P_1)\subseteq\mathrm{sol}(P_0)$ because $P_0\rightarrow_{\mathrm{FD}}P_1$. To show that $\mathrm{sol}(P_0)\subseteq\mathrm{sol}(P_1)$, let $\Sigma\in\mathrm{sol}(P_0)$ and $a=x\sigma$. Obviously,

$$\mathrm{sol}(P_1)=\{\Sigma\in\mathrm{sol}(P_0):x\sigma\in\delta_1(x)\}.$$

Suppose $\Sigma\notin\mathrm{sol}(P_1)$, or equivalently, $a\notin\delta_1(x)$. In this case, $a<l$ or $a>u$ and thus $\Sigma\notin S$. Since R is a relaxation of P_0, $\mathrm{sol}(P_0)\subseteq\mathrm{sol}(R)=S$ and thus $\Sigma\notin\mathrm{sol}(P_0)$. $\square$

Lemma 5. *If $P_0\rightarrow_{\mathrm{IC}(c,x)}P_1$, then $\mathrm{ir}(P_0|_{\{c\}},x)\rightarrow_{\mathrm{IC}(c,x)}\mathrm{ir}(P_1|_{\{c\}},x)$.*

Proof. Let $R_i=\mathrm{ir}(P_i|_{\{c\}},x)$, $S_i=\mathrm{sol}(R_i)$, $l_i=\min\{x\sigma:\Sigma\in S_i\}$, and $u_i=\max\{x\sigma:\Sigma\in S_i\}$. Let $\hat{R}_i=\mathrm{ir}(R_i|_{\{c\}},x)$, $\hat{S}_i=\mathrm{sol}(\hat{R}_i)$, $\hat{l}_i=\min\{x\sigma:\Sigma\in\hat{S}_i\}$, and $\hat{u}_i=\max\{x\sigma:\Sigma\in\hat{S}_i\}$.

$R_0\rightarrow_{\mathrm{FD}}R_1$: By Corollary 9, $R_0\rightarrow^{=}_{\mathrm{FD}}R_1$. Suppose $R_0=R_1$. Then $\rho_0(x)=\rho_1(x)$ and thus, by Definition 6, $\delta_0(x)=\delta_1(x)$. This is inconsistent with $P_0\rightarrow_{\mathrm{IC}(c,x)}P_1$.

By Definition 7, it remains to show that

$$\rho_1(x)=\begin{cases}\emptyset, & \text{if } \hat{S}_0=\emptyset\\ \{a\in\rho_0(x):\hat{l}_0\le a\le\hat{u}_0\} & \text{otherwise}.\end{cases}$$

This follows immediately from three facts:

1. By Definition 7,

$$\delta_1(x) = \begin{cases} \emptyset, & \text{if } S_0 = \emptyset \\ \{a \in \delta_0(x) : l_0 \le a \le u_0\} & \text{otherwise.} \end{cases}$$

2. By Definition 6, $\rho_0(x) = \delta_0(x)$ and $\rho_1(x) = \delta_1(x)$.

3. $R_0 = \hat{R}_0$ and thus $S_0 = \hat{S}_0$, $l_0 = \hat{l}_0$, and $u_0 = \hat{u}_0$.

□

Proposition 7. *If $P_0 \rightarrow_{\mathrm{IC}(c,x)} P_1$ and $P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{FD}} P_1$, then $P_2 \rightarrow_{\mathrm{IC}(c,x)} P_1$.*

Proof. Let $R_i = \mathrm{ir}(P_i|_{\{c\}}, x)$, $S_i = \mathrm{sol}(R_i)$, $l_i = \min\{x\sigma : \Sigma \in S_i\}$, and $u_i = \max\{x\sigma : \Sigma \in S_i\}$. By Definition 7,

$$\delta_1(x) = \begin{cases} \emptyset, & \text{if } S_0 = \emptyset \\ \{a \in \delta_0(x) : l_0 \le a \le u_0\} & \text{otherwise.} \end{cases}$$

By Lemma 5, $R_0 \rightarrow_{\mathrm{IC}(c,x)} R_1$ and, by Corollary 9, $R_0 \overset{=}{\rightarrow}_{\mathrm{FD}} R_2 \overset{=}{\rightarrow}_{\mathrm{FD}} R_1$. Hence, by Lemma 3, $S_0 = S_2$ and thus $l_0 = l_2$ and $u_0 = u_2$. Furthermore, by Lemma 2, $\delta_1(x) \subset \delta_2(x) \subset \delta_0(x)$. As a simple consequence from these facts,

$$\delta_1(x) = \begin{cases} \emptyset, & \text{if } S_2 = \emptyset \\ \{a \in \delta_2(x) : l_2 \le a \le u_2\} & \text{otherwise.} \end{cases}$$

and thus, by Definition 7, $P_2 \rightarrow_{\mathrm{IC}(c,x)} P_1$. □

Proposition 8. *If $P_0 \rightarrow_{\mathrm{IC}(c,x)} P_1$, $P_0 \rightarrow_{\mathrm{FD}} P_2$, $P_1 \not\rightarrow_{\mathrm{FD}} P_2$, $P_2 \not\rightarrow_{\mathrm{FD}} P_1$, and $P_1 \neq P_2$, then a FCSP P_3 exists s.t. $P_1 \rightarrow_{\mathrm{FD}} P_3$ and $P_2 \rightarrow_{\mathrm{IC}(c,x)} P_3$.*

Proof. Let $R_i = \mathrm{ir}(P_i|_{\{c\}}, x)$, $S_i = \mathrm{sol}(R_i)$, $l_i = \min\{x\sigma : \Sigma \in S_i\}$, and $u_i = \max\{x\sigma : \Sigma \in S_i\}$. By Definition 7,

$$\delta_1(x) = \begin{cases} \emptyset, & \text{if } S_0 = \emptyset \\ \{a \in \delta_0(x) : l_0 \le a \le u_0\} & \text{otherwise.} \end{cases}$$

Let $X = \mathrm{scope}(P_0)$, $C = \mathrm{store}(P_0)$, and $P_3 = (X, \delta_3, C)$ with $\delta_3 = \delta_2$ except for

$$\delta_3(x) = \begin{cases} \emptyset, & \text{if } S_2 = \emptyset \\ \{a \in \delta_2(x) : l_2 \le a \le u_2\} & \text{otherwise.} \end{cases}$$

If $S_2 \neq \emptyset$, then $l_0 \le l_2$ and $u_2 \le u_0$: We note that, by Corollary 9, $R_0 \overset{=}{\rightarrow}_{\mathrm{FD}} R_2$ and thus, by Corollary 5, $S_2 \subseteq S_0$. In consequence,

$$\begin{aligned} l_0 &= \min\{x\sigma : \Sigma \in S_0\} \\ &= \min\{\min\{x\sigma : \Sigma \in S_2\}, \min\{x\sigma : \Sigma \in S_0 - S_2\}\} \\ &= \min\{l_2, \min\{x\sigma : \Sigma \in S_0 - S_2\}\} \\ &\le l_2. \end{aligned}$$

By a similar argument, $u_2 \le u_0$.

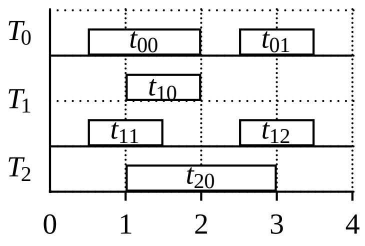
$P_2 \rightarrow_{\mathrm{FD}} P_3$: By Lemma 2, $\delta_2(x) \not\subseteq \delta_1(x)$, or equivalently, $a \in \delta_0(x)$ exists s.t. $a \in \delta_2(x)$ and $a \notin \delta_1(x)$. If $S_2 = \emptyset$, then $\emptyset = \delta_3(x) \subset \delta_2(x) \ni a$. Otherwise, by $a \notin \delta_1(x)$, $a < l_0$ or $a > u_0$ and thus $a < l_2$ or $a > u_2$. Hence $a \notin \delta_3(x)$ and thus $\delta_3(x) \subset \delta_2(x)$.

$P_1 \rightarrow_{\mathrm{FD}} P_3$: $P_3 \neq P_1$ because otherwise $P_2 \rightarrow_{\mathrm{FD}} P_1$. For all $y \in X - \{x\}$, $\delta_1(y) = \delta_0(y) \supseteq \delta_2(y) = \delta_3(y)$ because $P_1 \leftarrow_{\mathrm{IC}(c,x)} P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{IC}(c,x)} P_3$. Suppose $\delta_3(x) \not\subseteq \delta_1(x)$, or equivalently, $a \in \delta_0(x)$ exists s.t. $a \in \delta_3(x)$ and $a \notin \delta_1(x)$. If $S_2 = \emptyset$, then $\delta_3(x) = \emptyset$ contradicts $a \in \delta_3(x)$. Otherwise, by $a \notin \delta_1(x)$, $a < l_0$ or $a > u_0$ and thus $a < l_2$ or $a > u_2$. Hence $a \notin \delta_3(x)$. □

Corollary 10. *Let c be a constraint and $x \in \mathrm{scope}(c)$. $\rightarrow_{\mathrm{IC}(c,x)}$ is insensitive to $\rightarrow_{\mathrm{FD}}$.*

7 Application To A Finite-Domain Constraint Solver

A *track parallelization problem* (TPP) is specified by a set of tracks where each track is a set of tasks. The problem of solving a TPP consists in scheduling the tasks s.t. the tracks are processed in parallel. In its simplest form, a TPP requires to process two tasks in parallel. For a more typical example, consider the following Gantt-like chart:



The chart is based on the tracks T_0, T_1, and T_2 with $T_0 = \{t_{00}, t_{01}\}$, $T_1 = \{t_{10}, t_{11}, t_{12}\}$, and $T_2 = \{t_{20}\}$. T_0 and T_1 are processed in parallel because their schedules cover the same time slots. In contrast, T_2 is not processed in parallel to the other tracks because its schedule covers time slots that the other schedules do not cover and vice versa.

TPPs occur in school timetabling, especially in problem settings where options and official regulations imply the need to parallelize the education of pupils from several classes of the same grade [Mar01].

We proceed as follows. In Section 7.1, we define TPP constraints in terms of syntax and semantics. In Section 7.2, we describe a TPP solver consisting of several reductions. In Section 7.3, we show that the solver and any subset of its reductions are confluent.

7.1 Syntax and Semantics of TPP Constraints

A TPP constraint is written as $\mathtt{tpp}(\mathcal{T})$ where $|\mathcal{T}| > 1$ and, for all $T \in \mathcal{T}$, T is a non-empty set of pairs of FD variables. Each pair $(\mathtt{S}, \mathtt{P})$ of FD variables is intended to model a task in terms of its *start time* $\mathtt{S}$ and its *processing time* $\mathtt{P}$. Fixed start or processing times may be modeled by means of variables with singleton domains. We assume that processing times are greater than 0.

If P is a FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P)$, $T \in \mathcal{T}$, and $t = (\mathtt{S}, \mathtt{P}) \in T$, we write $\delta(t)$ instead of $\delta(\mathtt{S}) \times \delta(\mathtt{P})$.

Definition 8. Let P be a FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P)$. Let $T \in \mathcal{T}$ and $t = (\mathtt{S}, \mathtt{P}) \in T$.

1. *Value covers:*

$$\mathrm{vc}(t,\delta) = \begin{cases} \emptyset, & \text{if } \delta(t) = \emptyset \\ \bigcap_{(s,p)\in\delta(t)} [s, s+p-1] & \text{otherwise} \end{cases}$$

$$\mathrm{vc}(T,\delta) = \bigcup_{t\in T} \mathrm{vc}(t,\delta)$$

$$\mathrm{vc}(\mathcal{T},\delta) = \bigcup_{T\in\mathcal{T}} \mathrm{vc}(T,\delta)$$

2. *Value supplies:*

$$\mathrm{vs}(t,\delta) = \bigcup_{(s,p)\in\delta(t)} [s, s+p-1]$$

$$\mathrm{vs}(T,\delta) = \bigcup_{t\in T} \mathrm{vs}(t,\delta)$$

$$\mathrm{vs}(\mathcal{T},\delta) = \bigcap_{T\in\mathcal{T}} \mathrm{vs}(T,\delta)$$

Definition 9. Let P be an unfailed FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P)$. Let $T \in \mathcal{T}$ and $t = (\mathtt{S},\mathtt{P}) \in T$.

1. *Earliest start times:*

$$\mathrm{est}(t,\delta) = \min \delta(\mathtt{S})$$

$$\mathrm{est}(T,\delta) = \min_{t\in T} \mathrm{est}(t,\delta)$$

$$\mathrm{est}(\mathcal{T},\delta) = \max_{T\in\mathcal{T}} \mathrm{est}(T,\delta)$$

2. *Latest completion times:*

$$\mathrm{lct}(t,\delta) = \max \delta(\mathtt{S}) + \max \delta(\mathtt{P}) - 1$$

$$\mathrm{lct}(T,\delta) = \max_{t\in T} \mathrm{lct}(t,\delta)$$

$$\mathrm{lct}(\mathcal{T},\delta) = \min_{T\in\mathcal{T}} \mathrm{lct}(T,\delta)$$

Definition 10. Let P be a ground FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P)$. δ satisfies $\mathtt{tpp}(\mathcal{T})$ iff

$$|\{\mathrm{vc}(T,\delta) : T \in \mathcal{T}\}| = 1,$$

i.e. iff the track schedules cover the same value set.

Value supplies, value covers, earliest start times, and latest completion times have nice monotonicity properties that are summarized in Lemma 6 and Lemma 7. Lemma 8 shows that value supplies are closely related to earliest start and latest completion times. Lemma 9 summarizes properties of ground FCSPs.

Lemma 6. *Suppose* $P_0 \rightarrow_{\mathrm{FD}} P_1$ *and* $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$. *Let* $T \in \mathcal{T}$ *and* $t \in T$.

1. *Value supplies shrink monotonically, i.e.* $\mathrm{vs}(t,\delta_0) \supseteq \mathrm{vs}(t,\delta_1)$, $\mathrm{vs}(T,\delta_0) \supseteq \mathrm{vs}(T,\delta_1)$, *and* $\mathrm{vs}(\mathcal{T},\delta_0) \supseteq \mathrm{vs}(\mathcal{T},\delta_1)$.

2. *Value covers grow monotonically, i.e.*
   $\mathrm{vc}(t,\delta_0) \subseteq \mathrm{vc}(t,\delta_1)$, $\mathrm{vc}(T,\delta_0) \subseteq \mathrm{vc}(T,\delta_1)$, *and* $\mathrm{vc}(\mathcal{T},\delta_0) \subseteq \mathrm{vc}(\mathcal{T},\delta_1)$.

Proof. All properties follow immediately from Corollary 5. □

Lemma 7. *Suppose $P_0 \rightarrow_{\mathrm{FD}} P_1$, P_0 and P_1 are unfailed, and $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$. Let $T \in \mathcal{T}$ and $t \in T$.*

1. *Earliest start times grow monotonically, i.e.*
   $\mathrm{est}(t,\delta_0) \leq \mathrm{est}(t,\delta_1)$, $\mathrm{est}(T,\delta_0) \leq \mathrm{est}(T,\delta_1)$, *and* $\mathrm{est}(\mathcal{T},\delta_0) \leq \mathrm{est}(\mathcal{T},\delta_1)$.

2. *Latest completion times shrink monotonically, i.e.*
   $\mathrm{lct}(t,\delta_0) \geq \mathrm{lct}(t,\delta_1)$, $\mathrm{lct}(T,\delta_0) \geq \mathrm{lct}(T,\delta_1)$, *and* $\mathrm{lct}(\mathcal{T},\delta_0) \geq \mathrm{lct}(\mathcal{T},\delta_1)$.

Proof. All properties follow immediately from Corollary 5. □

Lemma 8. *Suppose P is an unfailed FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P)$. Let $T \in \mathcal{T}$ and $t = (\mathtt{S},\mathtt{P}) \in T$.*

1. *Earliest start times are equal to the least elements of value supplies, i.e.*
   $\mathrm{est}(t,\delta) = \min \mathrm{vs}(t,\delta)$, $\mathrm{est}(T,\delta) = \min \mathrm{vs}(T,\delta)$, *and* $\mathrm{est}(\mathcal{T},\delta) = \min \mathrm{vs}(\mathcal{T},\delta)$.

2. *Latest completion times are equal to the greatest elements of value supplies, i.e.*
   $\mathrm{lct}(t,\delta) = \max \mathrm{vs}(t,\delta)$, $\mathrm{lct}(T,\delta) = \max \mathrm{vs}(T,\delta)$, *and* $\mathrm{lct}(\mathcal{T},\delta) = \max \mathrm{vs}(\mathcal{T},\delta)$.

Proof. See [Mar01]. □

Lemma 9. *Suppose Γ is a ground FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(\Gamma)$. Let $T \in \mathcal{T}$ and $t \in T$.*

1. *In general,* $\mathrm{vs}(t,\gamma) = \mathrm{vc}(t,\gamma)$ *and* $\mathrm{vs}(T,\gamma) = \mathrm{vc}(T,\gamma)$.

2. *Furthermore, if γ satisfies $\mathtt{tpp}(\mathcal{T})$, then*

   (a) $\mathrm{vc}(T,\gamma) = \mathrm{vs}(\mathcal{T},\gamma) = \mathrm{vc}(\mathcal{T},\gamma)$,

   (b) $\mathrm{est}(T,\gamma) = \mathrm{est}(\mathcal{T},\gamma)$, *and*

   (c) $\mathrm{lct}(T,\gamma) = \mathrm{lct}(\mathcal{T},\gamma)$.

Proof. See [Mar01]. □

Corollary 11. *Suppose P_0 is a FCSP with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$. If $P_0 \rightarrow_{\mathrm{FD}} \ldots \rightarrow_{\mathrm{FD}} \Sigma$, Σ is ground, and σ satisfies $\mathtt{tpp}(\mathcal{T})$, then the relations depicted in Figure 1 hold.*

7.2 Solving TPP Constraints

We propose four reductions for solving TPP constraints. $\rightarrow_{\mathrm{PVS}}$ identifies and prunes all start and processing times that entail the covering of values that are not element of the value supply of the track set. Under certain conditions, $\rightarrow_{\mathrm{FC}}$ forces tasks to cover values. $\rightarrow_{\mathrm{IPT}}$ reveals inconsistencies by comparing bounds on the processing times of tracks. $\rightarrow_{\mathrm{NC}}$ reveals inconsistencies by identifying situations where values that have to be covered cannot be covered.

Definition 11. We say that $P_0 \rightarrow_{\mathrm{PVS}} P_1$ iff $P_0 \rightarrow_{\mathrm{FD}} P_1$ and $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$, $T \in \mathcal{T}$, $t = (\mathtt{S},\mathtt{P}) \in T$, and $a \in \mathrm{vs}(t,\delta_0)$ exist s.t. $a \notin \mathrm{vs}(\mathcal{T},\delta_0)$ and $\delta_1 = \delta_0$ except for

$$\delta_1(t) = \{(s,p) \in \delta_0(t) : a \notin [s, s+p-1]\}.$$

$$\begin{array}{ccccccccccc}
\mathrm{vc}(t,\delta_0) & \subseteq & \dots & \subseteq & \mathrm{vc}(t,\sigma) & = & \mathrm{vs}(t,\sigma) & \subseteq & \dots & \subseteq & \mathrm{vs}(t,\delta_0) \\
\rotatebox{-90}{$\subseteq$} & & & & \rotatebox{-90}{$\subseteq$} & & \rotatebox{-90}{$\subseteq$} & & & & \rotatebox{-90}{$\subseteq$} \\
\mathrm{vc}(T,\delta_0) & \subseteq & \dots & \subseteq & \mathrm{vc}(T,\sigma) & = & \mathrm{vs}(T,\sigma) & \subseteq & \dots & \subseteq & \mathrm{vs}(T,\delta_0) \\
\rotatebox{-90}{$\subseteq$} & & & & \| & & \| & & & & \rotatebox{-90}{$\subseteq$} \\
\mathrm{vc}(\mathcal{T},\delta_0) & \subseteq & \dots & \subseteq & \mathrm{vc}(\mathcal{T},\sigma) & = & \mathrm{vs}(\mathcal{T},\sigma) & \subseteq & \dots & \subseteq & \mathrm{vs}(\mathcal{T},\delta_0)
\end{array}$$

and

$$\begin{array}{ccccccccccc}
\mathrm{est}(t,\delta_0) & \leq & \dots & \leq & \mathrm{est}(t,\sigma) & \leq & \mathrm{lct}(t,\sigma) & \leq & \dots & \leq & \mathrm{lct}(t,\delta_0) \\
\| & & & & \| & & \| & & & & \| \\
\min \mathrm{vs}(t,\delta_0) & \leq & \dots & \leq & \min \mathrm{vs}(t,\sigma) & \leq & \max \mathrm{vs}(t,\sigma) & \leq & \dots & \leq & \max \mathrm{vs}(t,\delta_0) \\
\rotatebox{-90}{$\leq$} & & & & \rotatebox{-90}{$\leq$} & & \rotatebox{-90}{$\geq$} & & & & \rotatebox{-90}{$\geq$} \\
\mathrm{est}(T,\delta_0) & \leq & \dots & \leq & \mathrm{est}(T,\sigma) & \leq & \mathrm{lct}(T,\sigma) & \leq & \dots & \leq & \mathrm{lct}(T,\delta_0) \\
\| & & & & \| & & \| & & & & \| \\
\min \mathrm{vs}(T,\delta_0) & \leq & \dots & \leq & \min \mathrm{vs}(T,\sigma) & \leq & \max \mathrm{vs}(T,\sigma) & \leq & \dots & \leq & \max \mathrm{vs}(T,\delta_0) \\
\rotatebox{-90}{$\geq$} & & & & \| & & \| & & & & \rotatebox{-90}{$\leq$} \\
\mathrm{est}(\mathcal{T},\delta_0) & \leq & \dots & \leq & \mathrm{est}(\mathcal{T},\sigma) & \leq & \mathrm{lct}(\mathcal{T},\sigma) & \leq & \dots & \leq & \mathrm{lct}(\mathcal{T},\delta_0) \\
\| & & & & \| & & \| & & & & \| \\
\min \mathrm{vs}(\mathcal{T},\delta_0) & \leq & \dots & \leq & \min \mathrm{vs}(\mathcal{T},\sigma) & \leq & \max \mathrm{vs}(\mathcal{T},\sigma) & \leq & \dots & \leq & \max \mathrm{vs}(\mathcal{T},\delta_0)
\end{array}$$

Figure 1: The conclusions of Corollary 11.

Example 1. Consider the problem $P_0 = (X, \delta_0, \{\mathtt{tpp}(\mathcal{T})\})$ with $\mathcal{T} = \{T_0, T_1\}$, $T_0 = \{t_{00}\}$, $T_1 = \{t_{10}\}$, $t_{00} = (\mathtt{S}_{00}, \mathtt{P}_{00})$, $t_{10} = (\mathtt{S}_{10}, \mathtt{P}_{10})$, $\delta_0(\mathtt{S}_{00}) = \{1,2,4\}$, $\delta_0(\mathtt{P}_{00}) = \{1\}$, $\delta_0(\mathtt{S}_{10}) = \{1,3,4\}$, and $\delta_0(\mathtt{P}_{10}) = \{1\}$. $\rightarrow_{\mathrm{PVS}}$ applies two times:

1. $\rightarrow_{\mathrm{PVS}}$ applies to t_{00} because $2 \in \mathrm{vs}(t_{00}, \delta_0) = \{1,2,4\}$ and $2 \notin \mathrm{vs}(\mathcal{T}, \delta_0) = \{1,4\}$. We obtain the problem P_1 with $\delta_1 = \delta_0$ except for $\delta_1(\mathtt{S}_{00}) = \{1,4\}$.

2. $\rightarrow_{\mathrm{PVS}}$ applies to t_{10} because $3 \in \mathrm{vs}(t_{00}, \delta_1) = \{1,3,4\}$ and $3 \notin \mathrm{vs}(\mathcal{T}, \delta_1) = \{1,4\}$. We obtain the problem P_2 with $\delta_2 = \delta_1$ except for $\delta_2(\mathtt{S}_{10}) = \{1,4\}$.

Definition 12. We say that $P_0 \rightarrow_{\mathrm{FC}} P_1$ iff $P_0 \rightarrow_{\mathrm{FD}} P_1$ and $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$, $T \in \mathcal{T}$, $t = (\mathtt{S}, \mathtt{P}) \in T$, and $a \in \mathrm{vc}(\mathcal{T}, \delta_0)$ exist s.t. $a \notin \mathrm{vc}(T, \delta_0)$, $a \in \mathrm{vs}(t, \delta_0)$, $a \notin \mathrm{vs}(u, \delta_0)$ for all $u \in T$, $u \neq t$, and $\delta_1 = \delta_0$ except for

$$\delta_1(t) = \{(s,p) \in \delta_0(t) : a \in [s, s+p-1]\}.$$

Example 2. Consider the problem $P_0 = (X, \delta_0, \{\mathtt{tpp}(\mathcal{T})\})$ with $\mathcal{T} = \{T_0, T_1\}$, $T_0 = \{t_{00}\}$, $T_1 = \{t_{10}, t_{11}\}$, $t_{00} = (\mathtt{S}_{00}, \mathtt{P}_{00})$, $t_{10} = (\mathtt{S}_{10}, \mathtt{P}_{10})$, $t_{11} = (\mathtt{S}_{11}, \mathtt{P}_{11})$, $\delta_0(\mathtt{S}_{00}) = [0,2]$, $\delta_0(\mathtt{P}_{00}) = \{5\}$, $\delta_0(\mathtt{S}_{10}) = \{0,3\}$, $\delta_0(\mathtt{P}_{10}) = \{1,2\}$, $\delta_0(\mathtt{S}_{11}) = \{1,3\}$, and $\delta_0(\mathtt{P}_{10}) = \{1,2\}$. $\rightarrow_{\mathrm{FC}}$ applies four times:

1. $\rightarrow_{\text{FC}}$ applies to t_{11} because $2 \in \text{vc}(\mathcal{T},\delta_0) = [2,4]$, $2 \notin \text{vc}(T_1,\delta_0) = \emptyset$, $2 \in \text{vs}(t_{11},\delta_0) = [1,4]$, and $2 \notin \text{vs}(t_{10},\delta_0) = \{0,1,3,4\}$. We obtain the problem P_1 with $\delta_1 = \delta_0$ except for $\delta_1(\texttt{S}_{11}) = \{1\}$ and $\delta_1(\texttt{P}_{11}) = \{2\}$.

2. $\rightarrow_{\text{FC}}$ applies to t_{00} because $1 \in \text{vc}(\mathcal{T},\delta_1) = [1,4]$, $1 \notin \text{vc}(T_0,\delta_1) = [2,4]$, $1 \in \text{vs}(t_{00},\delta_1) = [0,6]$, and t_{00} is the only task in T_0. We obtain the problem P_2 with $\delta_2 = \delta_1$ except for $\delta_2(\texttt{S}_{00}) = \{0,1\}$.

3. $\rightarrow_{\text{FC}}$ applies to t_{10} because $3 \in \text{vc}(\mathcal{T},\delta_2) = [1,4]$, $3 \notin \text{vc}(T_1,\delta_2) = \{1,2\}$, $3 \in \text{vs}(t_{10},\delta_2) = \{0,1,3,4\}$, and $3 \notin \text{vs}(t_{11},\delta_2) = \{1,2\}$. We obtain the problem P_3 with $\delta_3 = \delta_2$ except for $\delta_3(\texttt{S}_{10}) = \{3\}$.

4. $\rightarrow_{\text{FC}}$ applies to t_{10} because $4 \in \text{vc}(\mathcal{T},\delta_3) = [1,4]$, $4 \notin \text{vc}(T_1,\delta_3) = [1,3]$, $4 \in \text{vs}(t_{10},\delta_3) = \{3,4\}$, and $4 \notin \text{vs}(t_{11},\delta_3) = \{1,2\}$. We obtain the problem P_4 with $\delta_4 = \delta_3$ except for $\delta_4(\texttt{P}_{10}) = \{2\}$.

Definition 13. We say that $P_0 \rightarrow_{\text{IPT}} P_1$ iff $P_0 \rightarrow_{\text{FD}} P_1$, P_1 is failed, and $\texttt{tpp}(\mathcal{T}) \in \text{store}(P_0)$, $T_0, T_1 \in \mathcal{T}$, and $l,u \geq 0$ exist s.t., for all $\Gamma \in \text{gs}(P_0)$, l is a lower bound on $|\text{vc}(T_0,\gamma)|$, u is an upper bound on $|\text{vc}(T_1,\gamma)|$, and $u < l$.

Example 3. Consider the problem $(X,\delta,\{\texttt{tpp}(\{T_0,T_1\})\})$ with $T_0 = \{t_{00},t_{01}\}$ and $T_1 = \{t_{10},t_{11}\}$ where $t_{00} = (\{0,5\},\{2\})$, $t_{01} = (\{2,6\},\{1,2,3\})$, $t_{10} = (\{2,3\},\{4,5\})$, and $t_{11} = (\{0,6\},\{2,3\})$. (To simplify matters, the variables have been replaced by their domains.) We note that $\text{vs}(T_0,\delta) = \text{vs}(T_1,\delta) = [0,8]$ and that T_0 cannot cover more than five values. If the tasks of T_1 are allowed to overlap, T_1 has a schedule covering five values and $\rightarrow_{\text{IPT}}$ does not apply. However, if the schedules of T_1 are required to be disjunctive, each of them will cover at least six values. In consequence, the tracks cannot be processed in parallel. $\rightarrow_{\text{IPT}}$ will reveal this inconsistency if the demand for disjunctiveness is considered when computing the lower bound on the number of values covered by the schedules of T_1.

Definition 14. Let $P = (X,\delta,C)$ be a FCSP with $\texttt{tpp}(\mathcal{T}) \in C$. If $T = \{t_1,\ldots t_n\} \in \mathcal{T}$, then $\text{vcg}(\mathcal{T},T,\delta)$ denotes the bipartite graph (U,V,E) with

- $U = \bigcup\limits_{\substack{1 \leq i \leq n \\ \delta(\texttt{P}_i) \neq \emptyset}} \left\{ u_i^j : 0 \leq j < \max \delta(\texttt{P}_i) \right\}$,

- $V = \text{vc}(\mathcal{T},\delta)$, and

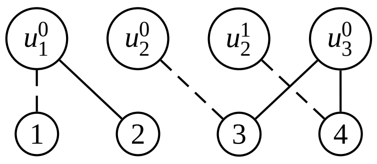
- $E = \left\{ (u_i^j, a) : u_i^j \in U \wedge a \in V \wedge \exists s \in \delta(\texttt{S}_i).\ s + j = a \right\}$.

We call this structure *value-cover graph*.

Definition 15. We say that $P_0 \rightarrow_{\text{NC}} P_1$ iff $P_0 \rightarrow_{\text{FD}} P_1$, P_1 is failed, and $\texttt{tpp}(\mathcal{T}) \in \text{store}(P_0)$ and $T \in \mathcal{T}$ exist s.t. $\text{vcg}(\mathcal{T},T,\delta_0) = (U,V,E)$ does not have a matching[1] M with $|M| = |V|$.

Example 4. Consider the problem $(X,\delta,\{\texttt{tpp}(\{T_0,T_1\})\})$ with $T_1 = \{t_{10},t_{11},t_{12}\}$, $t_{10} = (\{1,2\},\{1\})$, $t_{11} = (\{3\},\{2\})$, and $t_{12} = (\{3,4\},\{1\})$. (To simplify matters, the variables have been replaced by their domains.) Suppose $\text{vc}(T_0,\delta) = \text{vs}(T_0,\delta) = [1,4]$. We note that $\text{vs}(T_1,\delta) = [1,4]$. Now consider the value-cover graph $\text{vcg}(\{T_0,T_1\},T_1,\delta)$:

[1]Given a bipartite graph (U,V,E), a *matching* is a subset of edges $M \subseteq E$ s.t., for all vertices $v \in U \cup V$, at most one edge of M is incident on v.



The dotted edges constitute a matching of cardinality 3 and it is easy to verify that it has maximum cardinality. Hence only three values out of $[1,4]$ can be covered simultaneously. $\rightarrow_{\mathrm{NC}}$ detects this inconsistency and signals a failure.

Proposition 9. *$\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, $\rightarrow_{\mathrm{FC}}$, and $\rightarrow_{\mathrm{NC}}$ are correct.*

Proof. See [Mar01]. □

7.3 Confluence Properties of the TPP Solver

We show that each combination of $\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, $\rightarrow_{\mathrm{FC}}$, and $\rightarrow_{\mathrm{NC}}$ is confluent. We proceed as follows. We show that $\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, and $\rightarrow_{\mathrm{NC}}$ are insensitive to $\rightarrow_{\mathrm{FD}}$ (cf. Corollaries 12, 14, and 15) and that $\rightarrow_{\mathrm{FC}}$ is insensitive to $\rightarrow_{\mathrm{C}}$ (cf. 13) where $\rightarrow_{\mathrm{C}}$ is the union of all correct reductions contained in $\rightarrow_{\mathrm{FD}}$ (cf. Definition 4). It turns out that, if a correct reduction is insensitive to $\rightarrow_{\mathrm{FD}}$, then it is insensitive to $\rightarrow_{\mathrm{C}}$ (cf. Lemma 1). Thus, each of $\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, $\rightarrow_{\mathrm{FC}}$, and $\rightarrow_{\mathrm{NC}}$ is insensitive to $\rightarrow_{\mathrm{C}}$ and, by Corollary 1, each combination of $\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, $\rightarrow_{\mathrm{FC}}$, and $\rightarrow_{\mathrm{NC}}$ is insensitive to $\rightarrow_{\mathrm{C}}$. Then, by Corollary 3, each combination of $\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, $\rightarrow_{\mathrm{FC}}$, and $\rightarrow_{\mathrm{NC}}$ is confluent. The proofs make heavy use of monotonicity and correctness properties.

Proposition 10. *If $P_0 \rightarrow_{\mathrm{PVS}} P_1$ and $P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{FD}} P_1$, then $P_2 \rightarrow_{\mathrm{PVS}} P_1$.*

Proof. Let $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$, $T \in \mathcal{T}$, $t = (\mathtt{S},\mathtt{P}) \in T$, and $a \in \mathrm{vs}(t,\delta_0)$ s.t. $a \notin \mathrm{vs}(\mathcal{T},\delta_0)$ and $\delta_1 = \delta_0$ except for

$$\delta_1(t) = \{(s,p) \in \delta_0(t) : a \notin [s,s+p-1]\}.$$

By Lemma 2, $\delta_1(t) \subset \delta_2(t) \subset \delta_0(t)$. As a consequence,

$$\delta_1(t) = \{(s,p) \in \delta_2(t) : a \notin [s,s+p-1]\}.$$

Let $\Delta = \delta_2(t) - \delta_1(t)$. We observe that

$$\emptyset \neq \Delta \subset \delta_0(t) - \delta_1(t) = \{(s,p) \in \delta_0(t) : a \in [s,s+p-1]\}.$$

It follows that

$$a \in \bigcup_{(s,p)\in\Delta} [s,s+p-1] \subseteq \bigcup_{(s,p)\in\delta_2(t)} [s,s+p-1] = \mathrm{vs}(t,\delta_2).$$

Furthermore, by Lemma 6, $a \notin \mathrm{vs}(\mathcal{T},\delta_2)$. We conclude that $P_2 \rightarrow_{\mathrm{PVS}} P_1$. □

Proposition 11. *If $P_0 \rightarrow_{\mathrm{PVS}} P_1$, $P_0 \rightarrow_{\mathrm{FD}} P_2$, $P_1 \not\rightarrow_{\mathrm{FD}} P_2$, $P_2 \not\rightarrow_{\mathrm{FD}} P_1$, and $P_1 \neq P_2$, then a FCSP P_3 exists s.t. $P_1 \rightarrow_{\mathrm{FD}} P_3$ and $P_2 \rightarrow_{\mathrm{PVS}} P_3$.*

Proof. Let $X = \mathrm{scope}(P_0)$ and $C = \mathrm{store}(P_0)$. Let $\mathtt{tpp}(\mathcal{T}) \in C$, $T \in \mathcal{T}$, $t = (\mathtt{S},\mathtt{P}) \in T$, and $a \in \mathrm{vs}(t,\delta_0)$ s.t. $a \notin \mathrm{vs}(\mathcal{T},\delta_0)$ and $\delta_1 = \delta_0$ except for

$$\delta_1(t) = \{(s,p) \in \delta_0(t) : a \notin [s,s+p-1]\}.$$

Let $P_3 = (X, \delta_3, C)$ with $\delta_3 = \delta_2$ except for

$$\delta_3(t) = \{(s,p) \in \delta_2(t) : a \notin [s, s+p-1]\}.$$

$P_2 \rightarrow_{\mathrm{PVS}} P_3$: By Lemma 2, $\delta_2(t) \not\subseteq \delta_1(t)$, or equivalently, $(s,p) \in \delta_0(t)$ exists s.t. $(s,p) \in \delta_2(t)$ and $(s,p) \notin \delta_1(t)$. We conclude that $a \in [s, s+p-1]$ and thus $(s,p) \notin \delta_3(t)$. Hence $\delta_3(t) \subset \delta_2(t)$ and thus $P_2 \rightarrow_{\mathrm{FD}} P_3$. Furthermore, $a \in \mathrm{vs}(t, \delta_2)$ and, by Lemma 6, $a \notin \mathrm{vs}(\mathcal{T}, \delta_2)$. It follows that $P_2 \rightarrow_{\mathrm{PVS}} P_3$.

$P_1 \rightarrow_{\mathrm{FD}} P_3$: $P_3 \neq P_1$ because otherwise $P_2 \rightarrow_{\mathrm{FD}} P_1$. For all $x \in X - \{\mathtt{S}, \mathtt{P}\}$, $\delta_1(x) = \delta_0(x) \supseteq \delta_2(x) = \delta_3(x)$ because $P_1 \leftarrow_{\mathrm{PVS}} P_0 \rightarrow_{\mathrm{FD}} P_2 \rightarrow_{\mathrm{PVS}} P_3$. Suppose $\delta_3(t) \not\subseteq \delta_1(t)$, or equivalently, $(s,p) \in \delta_0(t)$ exists s.t. $(s,p) \in \delta_3(t)$ and $(s,p) \notin \delta_1(t)$. We conclude that $a \in [s, s+p-1]$ and thus $(s,p) \notin \delta_3(t)$. □

Corollary 12. *$\rightarrow_{\mathrm{PVS}}$ is insensitive to $\rightarrow_{\mathrm{FD}}$.*

Proposition 12. *If $P_0 \rightarrow_{\mathrm{FC}} P_1$ and $P_0 \rightarrow_{\mathrm{C}} P_2 \rightarrow_{\mathrm{C}} P_1$, then $P_2 \rightarrow_{\mathrm{FC}} P_1$.*

Proof. Let $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$, $T \in \mathcal{T}$, $t = (\mathtt{S}, \mathtt{P}) \in T$, and $a \in \mathrm{vc}(\mathcal{T}, \delta_0)$ s.t. $a \notin \mathrm{vc}(T, \delta_0)$, $a \in \mathrm{vs}(t, \delta_0)$, $a \notin \mathrm{vs}(u, \delta_0)$ for all $u \in T$, $u \neq t$, and $\delta_1 = \delta_0$ except for

$$\delta_1(t) = \{(s,p) \in \delta_0(t) : a \in [s, s+p-1]\}.$$

By Lemma 2, $\delta_1(t) \subset \delta_2(t) \subset \delta_0(t)$. As a consequence,

$$\delta_1(t) = \{(s,p) \in \delta_2(t) : a \in [s, s+p-1]\}.$$

Let $\Delta = \delta_2(t) - \delta_1(t)$. We observe that

$$\emptyset \neq \Delta \subset \delta_0(t) - \delta_1(t) = \{(s,p) \in \delta_0(t) : a \notin [s, s+p-1]\}.$$

It follows that

$$a \in \bigcup_{(s,p) \in \delta_1(t)} [s, s+p-1] \subseteq \bigcup_{(s,p) \in \delta_2(t)} [s, s+p-1] = \mathrm{vs}(t, \delta_2)$$

and

$$a \notin \bigcap_{(s,p) \in \Delta} [s, s+p-1] \supseteq \bigcap_{(s,p) \in \delta_2(t)} [s, s+p-1] = \mathrm{vc}(t, \delta_2).$$

Now, because no task in R other than t can cover a, $a \notin \mathrm{vc}(T, \delta_2)$. Furthermore, by Lemma 6, $a \in \mathrm{vc}(\mathcal{T}, \delta_2)$ and $a \notin \mathrm{vs}(u, \delta_2)$ for all $u \in T$, $u \neq t$. We conclude that $P_2 \rightarrow_{\mathrm{FC}} P_1$. □

Proposition 13. *If $P_0 \rightarrow_{\mathrm{FC}} P_1$, $P_0 \rightarrow_{\mathrm{C}} P_2$, $P_1 \not\rightarrow_{\mathrm{C}} P_2$, $P_2 \not\rightarrow_{\mathrm{C}} P_1$, and $P_1 \neq P_2$, then a FCSP P_3 exists s.t. $P_1 \rightarrow_{\mathrm{C}} P_3$ and $P_2 \rightarrow_{\mathrm{FC}} P_3$.*

Proof. Let $X = \mathrm{scope}(P_0)$ and $C = \mathrm{store}(P_0)$. Let $\mathtt{tpp}(\mathcal{T}) \in C$, $T \in \mathcal{T}$, $t = (\mathtt{S}, \mathtt{P}) \in T$, and $a \in \mathrm{vc}(\mathcal{T}, \delta_0)$ s.t. $a \notin \mathrm{vc}(T, \delta_0)$, $a \in \mathrm{vs}(t, \delta_0)$, $a \notin \mathrm{vs}(u, \delta_0)$ for all $u \in T$, $u \neq t$, and $\delta_1 = \delta_0$ except for

$$\delta_1(t) = \{(s,p) \in \delta_0(t) : a \in [s, s+p-1]\}.$$

Let $P_3 = (X, \delta_3, C)$ with $\delta_3 = \delta_2$ except for

$$\delta_3(t) = \{(s,p) \in \delta_2(t) : a \in [s, s+p-1]\}.$$

$P_2 \to_{\text{FC}} P_3$: By Lemma 6, $a \in \text{vc}(\mathcal{T}, \delta_2)$ and $a \notin \text{vs}(u, \delta_2)$ for all $u \in T$, $u \neq t$. By Lemma 2, $\delta_2(t) \not\subseteq \delta_1(t)$, or equivalently, $(s,p) \in \delta_0(t)$ exists s.t. $(s,p) \in \delta_2(t)$ and $(s,p) \notin \delta_1(t)$. We conclude that $a \notin [s, s+p-1]$ and thus $(s,p) \notin \delta_3(t)$. Hence $\delta_3(t) \subset \delta_2(t)$ and thus $P_2 \to_{\text{FD}} P_3$. Furthermore, it follows that $a \notin \text{vc}(t, \delta_2)$. Now, because no task in R other than t can cover a, $a \notin \text{vc}(T, \delta_2)$. To show that $a \in \text{vs}(t, \delta_2)$, let Σ in $\text{sol}(P_0)$. By Lemma 6, $a \in \text{vc}(\mathcal{T}, \sigma)$. By Lemma 9, $\text{vc}(T, \sigma) = \text{vc}(\mathcal{T}, \sigma)$. Suppose $a \notin \text{vs}(t, \sigma)$. By Lemma 9, $a \notin \text{vc}(t, \sigma)$. Now, because no task in R covers a, $a \notin \text{vc}(T, \sigma)$. Putting it all together, we obtain the contradiction $a \notin \text{vc}(T, \sigma) = \text{vc}(\mathcal{T}, \sigma) \ni a$. $\text{sol}(P_0) = \text{sol}(P_2)$ because $P_0 \to_{\text{C}} P_2$ and $\to_{\text{C}}$ is correct. Hence $\Sigma \in \text{sol}(P_2)$ and thus $P_2 \to_{\text{FD}} \Sigma$. By Lemma 6, $a \in \text{vs}(t, \sigma) \subseteq \text{vs}(t, \delta_2)$.

$P_1 \to_{\text{C}} P_3$: $P_3 \neq P_1$ because otherwise $P_2 \to_{\text{FD}} P_1$. For all $x \in X - \{\texttt{S}, \texttt{P}\}$, $\delta_1(x) = \delta_0(x) \supseteq \delta_2(x) = \delta_3(x)$ because $P_1 \leftarrow_{\text{FC}} P_0 \to_{\text{FD}} P_2 \to_{\text{FC}} P_3$. Suppose $\delta_3(t) \not\subseteq \delta_1(t)$, or equivalently, $(s,p) \in \delta_0(t)$ exists s.t. $(s,p) \in \delta_3(t)$ and $(s,p) \notin \delta_1(t)$. We conclude that $a \notin [s, s+p-1]$ and thus $(s,p) \notin \delta_3(t)$. Finally, $\text{sol}(P_1) = \text{sol}(P_0) = \text{sol}(P_2) = \text{sol}(P_3)$ because $P_1 \leftarrow_{\text{FC}} P_0 \to_{\text{C}} P_2 \to_{\text{FC}} P_3$ and both $\to_{\text{FC}}$ and $\to_{\text{C}}$ are correct. □

Corollary 13. *$\to_{\text{FC}}$ is insensitive to $\to_{\text{C}}$.*

Proposition 14. *If $P_0 \to_{\text{IPT}} P_1$ and $P_0 \to_{\text{FD}} P_2 \to_{\text{FD}} P_1$, then $P_2 \to_{\text{IPT}} P_1$.*

Proof. Let $\texttt{tpp}(\mathcal{T}) \in \text{store}(P_0)$, $T_0, T_1 \in \mathcal{T}$, and $l, u \geq 0$ s.t., for all $\Gamma \in \text{gs}(P_0)$, l is a lower bound on $|\text{vc}(T_0, \gamma)|$, u is an upper bound on $|\text{vc}(T_1, \gamma)|$, and $u < l$. By Corollary 5, $\text{gs}(P_2) \subseteq \text{gs}(P_0)$. Let $\Gamma \in \text{gs}(P_2)$. Because $\Gamma \in \text{gs}(P_0)$, we know that l is a lower bound on $|\text{vc}(T_0, \gamma)|$, u is an upper bound on $|\text{vc}(T_1, \gamma)|$, and $u < l$. Furthermore, P_1 is failed because $P_0 \to_{\text{IPT}} P_1$. We conclude that $P_2 \to_{\text{IPT}} P_1$. □

Proposition 15. *If $P_0 \to_{\text{IPT}} P_1$, $P_0 \to_{\text{FD}} P_2$, $P_1 \not\to_{\text{FD}} P_2$, $P_2 \not\to_{\text{FD}} P_1$, and $P_1 \neq P_2$, then a FCSP P_3 exists s.t. $P_1 \to_{\text{FD}} P_3$ and $P_2 \to_{\text{IPT}} P_3$.*

Proof. Let $X = \text{scope}(P_0)$ and $C = \text{store}(P_0)$. Let $\texttt{tpp}(\mathcal{T}) \in C$, $T_0, T_1 \in \mathcal{T}$, and $l, u \geq 0$ s.t., for all $\Gamma \in \text{gs}(P_0)$, l is a lower bound on $|\text{vc}(T_0, \gamma)|$, u is an upper bound on $|\text{vc}(T_1, \gamma)|$, and $u < l$. Let $P_3 = (X, \delta_3, C)$ with $\delta_3(x) = \delta_1(x) \cap \delta_2(x)$ for all $x \in X$. P_3 is failed because P_1 is failed.

Suppose $P_1 \not\to_{\text{FD}} P_3$. Then either $\delta_1 = \delta_3$ or $x \in X$ exists s.t. $\delta_3(x) \not\subseteq \delta_1(x)$. The latter case contradicts the construction of P_3. If $\delta_1 = \delta_3$, then $\delta_1(x) \subseteq \delta_2(x)$ for all $x \in X$ and thus either $P_1 = P_2$ or $P_2 \to_{\text{FD}} P_1$.

Suppose $P_2 \not\to_{\text{FD}} P_3$. Then either $\delta_2 = \delta_3$ and thus $P_2 = P_3$ or $x \in X$ exists s.t. $\delta_3(x) \not\subseteq \delta_2(x)$. The former case contradicts $P_1 \not\to_{\text{FD}} P_2$, the latter case contradicts the construction of P_3.

Furthermore, by Corollary 5, $\text{gs}(P_2) \subseteq \text{gs}(P_0)$. Let $\Gamma \in \text{gs}(P_2)$. Because $\Gamma \in \text{gs}(P_0)$, we know that l is a lower bound on $|\text{vc}(T_0, \gamma)|$, u is an upper bound on $|\text{vc}(T_1, \gamma)|$, and $u < l$.

We conclude that $P_2 \to_{\text{IPT}} P_3$. □

Corollary 14. *$\to_{\text{IPT}}$ is insensitive to $\to_{\text{FD}}$.*

Proposition 16. *If $P_0 \to_{\text{NC}} P_1$ and $P_0 \to_{\text{FD}} P_2 \to_{\text{FD}} P_1$, then $P_2 \to_{\text{NC}} P_1$.*

Proof. Let $\texttt{tpp}(\mathcal{T}) \in \text{store}(P_0)$ and $T \in \mathcal{T}$ s.t. $G_0 = (U_0, V_0, E_0) = \text{vcg}(\mathcal{T}, T, \delta_0)$ does not have a matching M_0 with $|M_0| = |V_0|$. Suppose $G_2 = (U_2, V_2, E_2) = \text{vcg}(\mathcal{T}, T, \delta_2)$ has a matching M_2 with $|M_2| = |V_2|$. Let $M_0 = \left\{(u_i^j, a) \in M_2 : a \in V_0\right\}$. We will show that M_0 is a matching in G_0 and that $|M_0| = |V_0|$.

1. M_0 is a matching because $M_0 \subseteq M_2$.

2. $M_0 \subseteq E_0$: Let $(u_i^j, a) \in M_0$. We know that $u_i^j \in U_2$, $a \in V_0$, and that $s \in \delta_2(\mathtt{S}_i)$ exists s.t. $s + j = a$. Let $s \in \delta_2(\mathtt{S}_i)$ s.t. $s + j = a$. $u_i^j \in U_0$ and $s \in \delta_0(\mathtt{S}_i)$ because $P_0 \rightarrow_{\mathrm{FD}} P_2$.

3. $|M_0| = |V_0|$ because, for all $a \in V_2$, a is matched by M_2 and, by Lemma 6, $V_0 = \mathrm{vc}(\mathcal{T}, \delta_0) \subseteq \mathrm{vc}(\mathcal{T}, \delta_2) = V_2$.

Furthermore, P_1 is failed because $P_0 \rightarrow_{\mathrm{NC}} P_1$. We conclude that $P_2 \rightarrow_{\mathrm{NC}} P_1$. $\square$

Proposition 17. *If $P_0 \rightarrow_{\mathrm{NC}} P_1$, $P_0 \rightarrow_{\mathrm{FD}} P_2$, $P_1 \not\rightarrow_{\mathrm{FD}} P_2$, $P_2 \not\rightarrow_{\mathrm{FD}} P_1$, and $P_1 \neq P_2$, then a FCSP P_3 exists s.t. $P_1 \rightarrow_{\mathrm{FD}} P_3$ and $P_2 \rightarrow_{\mathrm{NC}} P_3$.*

Proof. Similar to the proof of Proposition 15 by exploiting the fact that $(U_2, V_2, E_2) = \mathrm{vcg}(\mathcal{T}, T, \delta_2)$ with $\mathtt{tpp}(\mathcal{T}) \in \mathrm{store}(P_0)$ and $T \in \mathcal{T}$ does not have a matching M_2 with $|M_2| = |V_2|$ as shown in the proof of Proposition 16. $\square$

Corollary 15. *$\rightarrow_{\mathrm{NC}}$ is insensitive to $\rightarrow_{\mathrm{FD}}$.*

Corollary 16. *$\rightarrow_{\mathrm{PVS}}$, $\rightarrow_{\mathrm{IPT}}$, and $\rightarrow_{\mathrm{NC}}$ are insensitive to $\rightarrow_{\mathrm{C}}$.*

Corollary 17. *If $R \subseteq \{\rightarrow_{\mathrm{PVS}}, \rightarrow_{\mathrm{FC}}, \rightarrow_{\mathrm{IPT}}, \rightarrow_{\mathrm{NC}}\}$ and $\rightarrow_r = \bigcup R$, then $\rightarrow_r$ is terminating, insensitive to $\rightarrow_{\mathrm{C}}$, locally confluent, and confluent.*

8 Related Work

The Commutative Union Lemma [BN98] (CUL) is a well-known tool to prove confluence. It states that the union of two reductions is confluent if both reductions are confluent and commute. In the following, we compare our method to the method suggested by the CUL. We are interested in how to proceed in different situations: when proving the confluence of a reduction from scratch and when proving the confluence of a reduction that has been obtained by extending or reducing a confluent reduction. Note that our statements wrt. to the number of proof obligations arising on top-level are not meant to say anything about the complexity and the difficulties of the proofs that are actually required.

Let $(A, \bigcup_{1 \leq i \leq n} \rightarrow_i)$ be a reduction system. To show that $\bigcup_{1 \leq i \leq n} \rightarrow_i$ is confluent, the CUL suggests to show that each $\rightarrow_i$, $1 \leq i \leq n$, is confluent and that, for each $1 < i \leq n$, $\rightarrow_i$ and $\bigcup_{1 \leq k < i} \rightarrow_k$ commute. Hence n confluence proofs and $n-1$ proofs of commutation are required. Suppose we are showing that $\rightarrow_i$, $1 < i \leq n$, and $\bigcup_{1 \leq k < i} \rightarrow_k$ commute. In general, this proof requires to consider $i-1$ cases, one for each $\rightarrow_k$, $1 \leq k < i$, except for it is possible to characterize $\bigcup_{1 \leq k < i} \rightarrow_k$ in a way that facilitates to deal with all cases in one sweep. In summary, at least $2n+1 \in O(n)$ and at most $n + \frac{(n-1)(n-2)}{2} \in O(n^2)$ proof obligations arise. Our approach requires n proofs of insensitivity to $\bigcup_{1 \leq i \leq n} \rightarrow_i$, one for each $\rightarrow_i$, $1 \leq i \leq n$. In general, each proof of insensitivity has to consider n cases, one for each $\rightarrow_i$, $1 \leq i \leq n$, except for it is possible to characterize $\bigcup_{1 \leq i \leq n} \rightarrow_i$ or some superset relation in a way that facilitates to deal with all cases in one sweep. In summary, at least n and at most n^2 proof obligations arise. In our application to FD constraint solving, we proved insensitivity to either $\rightarrow_{\mathrm{FD}}$ or $\rightarrow_{\mathrm{C}}$. $\rightarrow_{\mathrm{FD}}$ allows for arbitrary domain reductions while $\rightarrow_{\mathrm{C}} \subseteq \rightarrow_{\mathrm{FD}}$ only allows for domain reductions that preserve solutions. This way it was possible to prove the confluence of the TPP solver with a number of proofs linear in the number of reductions.

Let $\rightarrow_{n+1} \in A \times A$. Suppose $\bigcup_{1 \leq i \leq n} \rightarrow_i$ is known to be confluent. To show that $\bigcup_{1 \leq i \leq n+1} \rightarrow_i$ is confluent, the CUL obliges to show that $\rightarrow_{n+1}$ is confluent and that $\rightarrow_{n+1}$ and $\bigcup_{1 \leq i \leq n} \rightarrow_i$ commute. As explained above, the proof of commutation may require to distinguish up to

n cases. In the worst case, our approach obliges to show that, for all $1 \leq i \leq n+1$, $\rightarrow_i$ is insensitive to $\bigcup_{1 \leq i \leq n+1} \rightarrow_i$. This amounts to proving confluence from scratch. However, if we can characterize a superset relation that contains all reductions we might wish to investigate, it is possible to reuse all prior proofs of insensitivity: Suppose we know that $\bigcup_{1 \leq i \leq n} \rightarrow_i$ is insensitive to the superset relation, then it is sufficient to show that $\rightarrow_{n+1}$ is insensitive to the superset relation. In our application to FD constraint solving, we used $\rightarrow_{\mathrm{FD}}$ and $\rightarrow_{\mathrm{C}}$ as superset relations. If we wished to extend the TPP solver, it was sufficient to show that the extension is insensitive to either $\rightarrow_{\mathrm{FD}}$ or $\rightarrow_{\mathrm{C}}$.

Suppose $\bigcup_{1 \leq i \leq n} \rightarrow_i$ is known to be confluent. Let $1 \leq k \leq n$. Is $\bigcup_{1 \leq i \leq n, i \neq k} \rightarrow_i$ confluent? If the CUL has been applied, then, for each $k < l \leq n$, it is necessary to prove that $\rightarrow_l$ and $\bigcup_{1 \leq i < l, i \neq k}$ commute. If our approach has been used, no proof obligations arise, independent of whether insensitivity has been proved to $\bigcup_{1 \leq i \leq n} \rightarrow_i$ or to some superset relation.

Suppose $\bigcup_{1 \leq i \leq n} \rightarrow_i$ is known to be confluent. Let $I \subset [1, \ldots, n]$. Is $\bigcup_{i \in I} \rightarrow_i$ confluent? The answers to this question are similar to those to the previous question. In particular, if our approach has been used to prove the confluence of $\bigcup_{1 \leq i \leq n} \rightarrow_i$, no proof obligations arise.

Critical pairs are employed in the static analysis of term-rewriting systems [BN98] and rule-based constraint solvers [AFM99]. Critical pairs are defined in a domain-specific way; basically, a critical pair is a pair of states obtained from the parallel application of two interfering rewrite rules to a common and minimal ancestor state. The importance of critical pairs is due to their relationship to local confluence. For example, a term-rewriting system is locally confluent iff all its critical pairs are joinable. However, the critical-pair approach is not modular. Suppose we are given a term-rewriting system the critical pairs of which are all joinable. After adding a rule, the approach requires to check all the critical pairs the new rule has with itself and with the other rules. After removing a rule, the approach requires to check all proofs of joinability; any proof that is based on the removed rule has be redone.

9 Conclusion

We developed a sufficient condition for local confluence and demonstrated that it is very well suited to study the behaviour of cooperating constraint solvers. First, we showed that any combination of solvers where each solver maintains either domain or interval consistency is confluent. Second, we studied a solver for track parallelization and showed that any combination of its reductions is confluent.

We call our approach modular because removing reductions does not affect confluence and because proving confluence after adding a reduction requires only a single proof that, if a superset relation is used, does not require to reconsider any other reduction. Our approach is suitable for the study of constraint solvers because superset relations are available that comprise all conceivable solvers.

References

[AFM99] Slim Abdennadher, Thom Frühwirth, and Holger Meuss. Confluence and semantics of constraint simplification rules. *Constraints*, 4(2):133–165, 1999.

[BN98] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[Mar01] Michael Marte. A global constraint for parallelizing the execution of task sets. Technical Report PMS-FB-2001-13, Institut für Informatik der Universität München, 2001. Available from http://www.pms.informatik.uni-muenchen.de/publikationen.

[MT00] Kurt Mehlhorn and Sven Thiel. Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint. In Rina Dechter, editor, *Sixth International Conference on Principles and Practice of Constraint Programming*, LNCS 1894, pages 306–319. Springer, 2000.

[Rég94] Jean-Charles Régin. A filtering algorithm for constraints of difference in CSPs. In *Proceedings of the 12th National Conference on Artificial Intelligence*, pages 362–367. AAAI Press, 1994.

[Rég96] Jean-Charles Régin. Generalized arc consistency for global cardinality constraint. In *Proceedings of the 13th National Conference on Artificial Intelligence and the 8th Innovative Applications of Artificial Intelligence Conference*, pages 209–215. AAAI Press, 1996.

[Tsa93] Edward Tsang. *Foundations of Constraint Satisfaction*. Academic Press, 1993.

[vH01] W. J. van Hoeve. The alldifferent constraint: A survey. In *6th Annual Workshop of the ERCIM Working Group on Constraints*, 2001.

[VSD98] Pascal Van Hentenryck, Vijay Saraswat, and Yves Deville. Design, implementation, and evaluation of the constraint language cc(FD). *Journal of Logic Programming*, 37(2):139–164, 1998.