When functional and bijective constraints make a CSP polynomial

Philippe David
LIRMM
(UMR C 09928 CNRS/universite Montpellier II)
161 rue Ada, 34392 Montpellier Cedex 5 - France
david@lirmm.fr

Abstract

Many works have been carried out to improve search efficiency in CSPs, but few of them treated the *semantics* of the constraints. In this paper, we expose some properties of two classes of constraints, functional and bijective constraints: we first present conditions under which arc and path consistencies are sufficient to guarantee the existence of a bactrack free solution; we then exhibit classes of polynomial problems, and finally we propose a new method of decomposition for problems containing functional or bijective constraints. An interesting point in this method is that the resolution complexity is known prior to the search.

1 Introduction

Improving search efficiency of Constraint Satisfaction Problems (CSPs) is one of the main goals of researchers in this domain. Their works can be classified into different approaches, like improving backtrack procedure [Haralick and Elliot, 1980; Dechter and Pearl, 1988], or filtering methods (arc and path consistencies in particular) [Montanari, 1974; Mack worth, 1977; Mohr and Henderson, 1986]. Another approach is to characterize polynomial classes; they rely on structure of problems [Freuder, 1978] (leading to the presentation of new methods of decomposition [Dechter and Pearl, 1987; Dechter and Pearl, 1989; Jegou, 1990]) or on the size of their domains [Dechter, 1990]. Few works have been done on the *semantics* of the relations; recently, Deville and Van Hentenryck studied monotonic and functional constraints [l99l], Perlin [1992] factorable relations, and van Beek [1992] identified a class of problems for which the tasks of finding a solution or the corresponding minimal network can both be solved efficiently.

Yet, in real applications, constraints don't take on any form. In particular a kind of constraints is often used: functional ones (VLSI tests [Stallman and Sussman, 1977], peptide synthesis [Janssen *et* a/., 1990]).

In this paper, we will study functional and bijective constraints (the relations defined by these constraints are respectively a function and a bijection). We will focus on the properties of arc and path consistencies when applied to these constraints, presenting a class of polynomial problems and a method of decomposition.

Section 2 presents definitions and preliminaries, section 3 and 4 expose results on functional and bijective constraints.

2 Preliminaries

We first present some definitions and the conventions we take in this paper:

A binary CSP $\mathcal{P}$ is defined by (X, D, C, R), where:

- X is the set of the n variables $X_1, \ldots, X_i, \ldots, X_n$;
- D is the set of the n domains $D_1, \ldots, D_i, \ldots, D_n$, where D_i is the set of the possible values for X_i, d_i, d_j, d_k are values of D_i, D_j, D_k, respectively, and d is the size of the largest domain;
- C is the set of the m constraints, where a constraint C_{ij} between the variables X_i and X_j is defined by its relation R_{ij};
- R is the set of the relations R_{ij}, where R_{ij} is a subset of the cartesian product $D_i \times D_j$ specifying which values of the variables are compatible with each other.

A *consistent instantiation* of a subset Y of the variables $X_{k_1}, \ldots, X_{k_l}$ is a tuple of values $(d_{k_1}, \ldots, d_{k_l})$ such that any constraint included in Y is satisfied. A *solution* is a consistent instantiation of X.

A graph $\mathcal{G} = (X, C)$ called *constraint graph* can be associated to a binary CSP $\mathcal{P}$, where the vertices of $\mathcal{G}$ represent the variables of $\mathcal{P}$, and the arcs (or edges), the constraints.

For any constraint C_{ij}, we assign to $R_{ij}(d_i, d_j)$ the boolean value *true* if and only if (d_i, d_j) belongs to the relation R_{ij}.

If X_i and X_j are not linked, C_{ij} is called a ***universal constraint***; we then have $R_{ij}(d_i, d_j)$ for any (d_i, d_j) of $D_i \times D_j$ $(R_{ij} = D_i \times D_j)$. Universal constraints do not appear in C.

In the following, we will suppose that a CSP is connected (its constraint graph is connected); if not, it can be decomposed into independant problems.

A value $d_k \in D_k$ is called a ***support*** in D_k of a value d_i for the constraint C_{ik} iff $R_{ik}(d_i, d_k)$; it is called a support in D_k of a tuple (d_i, d_j) iff $R_{ik}(d_i, d_k)$ and

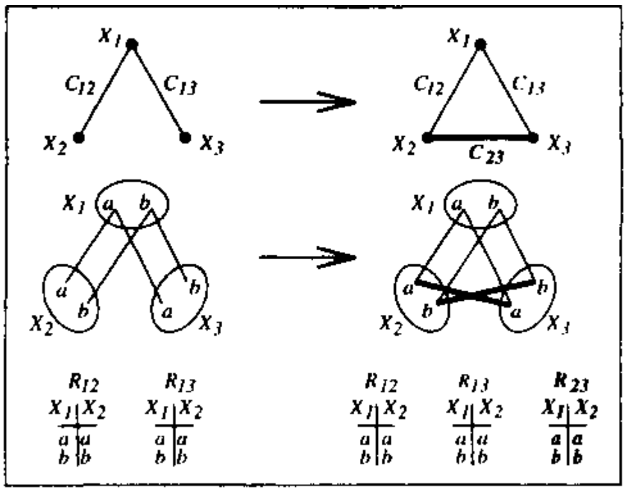


Figure 1: path consistency - representation of relations

$R_{jk}(d_j, d_k)$. Representations of relations can be either tables or graphs (see Figure 1).

We now recall the standard definitions of arc and path consistencies [Mackworth, 1977]:

Definition 2.1 *A domain D_i of D is* arc consistent *iff $\forall d_i \in D_i$, $\forall X_j$ such that $C_{ij} \in C$, $\exists d_j \in D_j$ such that $R_{ij}(d_i, d_j)$.*

Definition 2.2 *A CSP $\mathcal{P}$ is arc consistent iff $\forall D_i \in D$, D_i is arc consistent and $D_i \neq \emptyset$.*

Definition 2.3 *A pair of variables $\{X_i, X_j\}$ is* path consistent *iff $\forall (d_i, d_j) \in R_{ij}$, $\forall X_k \in X$, $\exists d_k \in D_k$ such that $R_{ik}(d_i, d_k)$ and $R_{jk}(d_j, d_k)$.*

Definition 2.4 *A CSP $\mathcal{P}$ is path consistent iff $\forall X_i, X_j \in X$, $\{X_i, X_j\}$ is path consistent.*

In other words, a CSP is arc consistent iff any value of any domain has at least one support for any constraint (and no domain is empty), and it is path consistent iff any tuple of any relation has at least one support in any domain.

More generally, a CSP is *k-consisttut* if and only if any consistent, instantiation of *k* — 1 variables may be extended into any other variable to produce a consistent instantiation of Ar variables. It is *globally consistent* if and only if it is k-consistent for any *k* from 1 to *77*. For binary problems, arc and path consistencies are respectively 2 and 3 consistencies.

A CSP *V* is not necessarily arc or path consistent; it can in this case be transformed into an equivalent problem P^c (i.e. it has the same solutions) by suppressing inconsistent values from the domains (arc consistency) or inconsistent tuples from the relations (path consistency); P^c is then called arc (path) consistent closure of this CSP *V* [Montanari, 1974; Mackworth, 1977; Mohr and Henderson, 1986]. Notice that achieving path consistency may modify the constraint graph structure: some constraints may be induced instead of universal constraints (see Figure 1).

Given a CSP, the question can be to find all solutions, or one solution, or to know if there exists a solution; this problem is known to be NP-complete. However, some polynomial classes have been exhibited, based upon the size of the domains [Dechter, 1990], or upon structural properties of the CSPs (width lower than the consistency degree of the CSP iFreuder, 1978]). Studying these classes provides bounds to the search complexity, and decomposition methods: cycle-cutset method [Dechter and Pearl, 1987], tree clustering [Dechter and Pearl, 1989], cyclic clustering [jegou, 1990].

Our purpose in this paper is to characterize polynomial classes, based upon the semantics of two kinds of constraints: functional and bijective constraints; we also present a decomposition method improving search efficiency in problems containing functional or bijective constraints.

3 Functional constraints

Definition 3.1
Given two variables X_i and X_j, we denote $X_i \rightarrow X_j$ iff for all $d_i \in D_i$ there exists at most one $d_j \in D_j$ such that $R_{ij}(d_i, d_j)$.
If it exists, the value is then noted $d_j = f_{ij}(d_i)$.
A constraint C_{ij} is functional *iff $X_i \rightarrow X_j$ or $X_j \rightarrow X_i$.*

The meaning of functional constraints is the same as this of functional dependencies used in the field of relational databases [Ullman, 1982].

For instance, the constraint $Z = Y^2$ is functional, with $Y \rightarrow Z$.

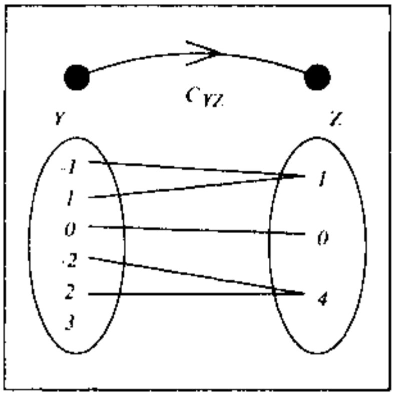


Figure 2: an exemple of a functional constraint

We can notice that for any functional constraint C_{ij}, $|R_{ij}| \leq d$.

When all the constraints are functional, achieving arc consistency is easy: AC4, the algorithm proposed by Mohr and Henderson [1986], is linear in the size of the problem; as the size of any relation is here bounded by d, the complexity is then $O(md)$.

In the following, we study a binary CSP $\mathcal{P}$ composed with the subset C_f of all functional constraints, and the subset of the other ones (non functional), C_o. If it only contains functional constraints ($C_o = \emptyset$), $\mathcal{P}$ is then called functional.

Let $\mathcal{G} = (X, C)$ be the constraint graph of $\mathcal{P}$. C contains both directed arcs representing functional constraints, where an arc is directed from X_i to X_j iff

$X_i \to X_j$, and edges (undirected arcs) for non functional constraints.

We can notice that arc and path consistency keeps the functionality property to any constraint (achieving it only suppresses values from domains and tuples from relations): when achieving arc consistency and path consistency on a CSP $\mathcal{P} = (X, D, C_f \cup C_o, R)$, we obtain the CSP $\mathcal{P}^c = (X, D^c, C_f^c \cup C_o^c, R^c)$ where $C_f \subseteq C_f^c$. However, even if $\mathcal{P}$ is functional ($C_o = \emptyset$), $\mathcal{P}^c$ isn't necessarily functional, because non functional constraints may be induced while achieving path consistency. Though, we can exhibit a class of induced constraints that are functional: transitivity constraints; property 3.1 characterizes them more formally:

Property 3.1 *Let $\mathcal{P} = (X, D, C, R)$ be a CSP, $\mathcal{P}^c = (X, D^c, C^c, R^c)$ its arc and path consistent closure, and $\mathcal{G} = (X, C)$ and $\mathcal{G}^c = (X, C^c)$ their constraint graphs. For all X_i, X_k in X, if there exists in C a sequence of functional constraints such that $X_i \to X_{k_1}, \ldots, X_{k_j} \to X_{k_{j+1}}, \ldots, X_{k_p} \to X_k$, then the constraint C_{ik}^c of C^c is functional ($X_i \to X_k$).*

Proof:

Each constraint of the sequence is functional; consequently, it allows at most one $d_{k_{j+1}}$ for each d_{k_j}.

The sequence of constraints $C_i, \ldots, C_k$ thus permits at most one d_k for each d_i.

The induced constraint C_{ik}^c is therefore functional: $X_i \to X_k$. □

This property is an immediate consequence of Armstrong's transitivity axiom for functional dependencies [Ullman, 1982].

We now recall some definitions issued from graph theory and define notions used in the following of the section.

Definitions 3.2 [Berge, 1970]

A variable X_j is a descendant *from X_i in a directed graph (X, C_f) iff there exists in C_f a path $(X_i, X_{k_1}, \ldots, X_{k_p}, X_j)$ such that $X_i \to X_{k_1}, \ldots, X_{k_p} \to X_j$.*

A variable X_r is a root *of a directed graph (X, C_f) iff any other variable of X is a descendant from X_r.*

Definition 3.3

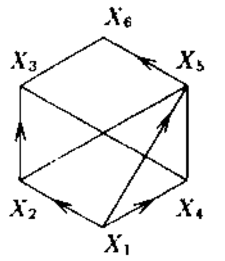
A subset $\mathcal{R}$ of variables of X is a root set *of a directed graph (X, C_f) iff any variable of $X - \mathcal{R}$ is a descendant from an element of $\mathcal{R}$.*

Definition 3.4 *Let $\mathcal{G} = (X, C_f)$.*

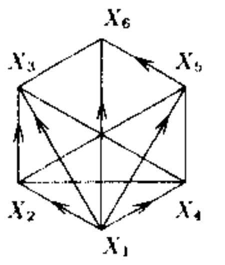
An order on X is called r-compatible *(with G or C_f) iff any variable X_j (except the first one) has at least one ancestor X_i in this order such that $X_i \to X_j$.*

Notice that the existence of such an order is equivalent to this of a root; any r-compatible order first element is a root.

Observe also a property of $\mathcal{G}^c$ ($\mathcal{P}^c$ constraint graph): any order which is r-compatible with $\mathcal{G}$ is r-compatible with $\mathcal{G}^c$; moreover, any order which first element is a root (if it exists) is r-compatible with $\mathcal{G}^c$ (see Figure 3).



$X = \{X_1, X_2, X_3, X_4, X_5, X_6\}$
$C_f = \{X_1 \to X_2, X_1 \to X_4,$
$X_1 \to X_5, X_2 \to X_3, X_5 \to X_6\}$
$C_o = \{C_{25}, C_{34}, C_{36}\}$
The order $\{X_1, X_3, X_2, X_4, X_6, X_5\}$ is not r-compatible
The order $\{X_1, X_2, X_3, X_4, X_5, X_6\}$ is r-compatible with Cf



$X = \{X_1, X_2, X_3, X_4, X_5, X_6\}$
$C_f^c = \{X_1 \to X_2, X_1 \to X_3,$
$X_1 \to X_4, X_1 \to X_5$
$X_1 \to X_6, X_2 \to X_3, X_5 \to X_6\}$
$C_o^c = \{C_{24}, C_{25}, C_{34}, C_{36}\}$
The orders $\{X_1, X_3, X_2, X_4, X_6, X_5\}$ and $\{X_1, X_2, X_3, X_4, X_5, X_6\}$ are both r-compatible with C_f

Figure 3: r-compatible orders of a CSP and of its arc and path consistent, closure[1]

The results we present now rely on the following property:

Property 3.2 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent, I_{k-1} be a consistent instantiation of $\{X_1, \ldots, X_{k-1}\}$, and X_k a new variable to instantiate. If there exists X_j ($1 \le j \le k-1$) such that $X_j \to X_k$, then I_{k-1} may be extended into a consistent instantiation of $\{X_1, \ldots, X_{k-1}, X_k\}$.*

Proof: Let $I_{k-1} = (d_1, \ldots, d_{k-1})$ be a consistent instantiation of $\{X_1, \ldots, X_{k-1}\}$, X_k a new variable to be instantiated, and X_j an instantiated variable ($1 \le j \le k-1$) such that $X_j \to X_k$.

As $\mathcal{P}$ is arc consistent, there exists $d_k \in D_k$ such that $R_{jk}(d_j, d_k)$ ($d_k = f_{jk}(d_j)$).

For all $i < k$

I_{k-1} is consistent, so $R_{ij}(d_i, d_j)$;

$\mathcal{P}$ is path consistent, (d_i, d_j) thus has a support in D_k: there exists $d'_k \in D_k$ such that $R_{ik}(d_i, d'_k)$ and $R_{jk}(d_j, d'_k)$;

$X_j \to X_k$, so there exists one only $d_k \in D_k$ such that $R_{jk}(d_j, d_k)$ ($d_k = f_{jk}(d_j)$);

Consequently, $d'_k = d_k$, so $R_{ik}(d_i, d_k)$

For all $i < k$ we therefore have $R_{ik}(d_i, d_k)$, which implies that $I_k = (d_1, \ldots, d_{k-1}, d_k)$ is consistent. The instantiation may thus be extended to X_k. □

An interesting point in this property is that the condition only concerns the functional constraints; the number, the kind and the structure of the other ones don't matter.

[1]This is just an exemple of how achieving arc and path consistency may modify the constraint graph: the constraint C_{35} might as well have been induced.

The existence of an r-compatible order makes it more easy to solve the problem, as corollary 3.3 shows:

Corollary 3.3 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent. If there exists on X an r-compatible order, then a solution exists and it can be found in linear time.*

Proof: The instantiation order just has to be r-compatible.

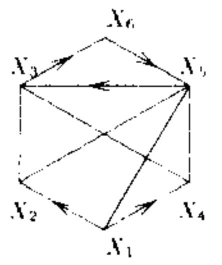
As the instantiation order is r-compatible, any variable to be instantiated X_j owns at least one predecessor X_i (thus X_i is already instantiated with the value d_i) such that $X_i \rightarrow X_j$. Due to property 3.2, $d_j = f_{ij}(d_i)$ may therefore instantiate X_j consistently. □

But such an order doesn't always exist; the problem can though be simplified due to the following corollary:

Corollary 3.4 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent, and I_{k-1} be a consistent instantiation of $X_1, \ldots, X_{k-1}$.*
If any non instantiated variable X_i ($k \leq i \leq n$) is such that there exists X_j verifying $X_j \rightarrow X_i$ and $1 \leq j < k$ (there exists an instantiated variable X_j such that $X_j \rightarrow X_i$), then I_{k-1} can be extended to a solution without backtrack.

Proof: Obviously deduced from property 3.2. □

Notice that the properties of the set $\{X_1, \ldots, X_{k-1}\}$ are these of a root set of G (see Figure 4).



$X = \{X_1, X_2, X_3, X_4, X_5, X_6\}$
$C_f = \{X_1 \rightarrow X_2, X_1 \rightarrow X_4,$
$X_3 \rightarrow X_6, X_6 \rightarrow X_5, X_5 \rightarrow X_3\}$
$C_n = \{C_{15}, C_{23}, C_{25}, C_{34}, C_{45}\}$

There is no r-compatible order
The set $\mathcal{R} = \{X_1, X_3\}$ is a minimum root set of $\mathcal{G}$

Figure 4: minimum root set.

We can therefore deduce theorem 1:

Theorem 1 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent, and $\mathcal{R} \subseteq X$ a root set of $\mathcal{G} = (X, C_f)$.*
If there exists a consistent instantiation of $\mathcal{R}$, then it can be extended to a solution without backtrack.

Proof: it's a direct consequency of corollary 3.4. □

An arc and path consistent problem is thus reduced into the resolution of one of its root sets; after this root set is consistently instantiated, this instantiation can be extended to a backtrack free solution. We can therefore deduce the number of solutions of the problem, as property 3.5 shows:

Property 3.5 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent, and $\mathcal{R} \subseteq X$ a root set of $\mathcal{G} = (X, C_f)$.*

The number of solutions of $\mathcal{P}$ equals the number of consistent instantiations of $\mathcal{R}$: each consistent instantiation of $\mathcal{R}$ can be extended to exactly one solution of $\mathcal{P}$.

Proof: Let $I = (d_{k_1}, \ldots, d_{k_r})$ be a consistent instantiation of $\mathcal{R} = X_{k_1}, \ldots, X_{k_r}$.

Due to Theorem 1, I can be extended to at least one solution of $\mathcal{P}$.

$\mathcal{R}$ is a root set of $\mathcal{G}$; for all $X_j \in X - \mathcal{R}$ there exists thus in $\mathcal{R}$ a variable X_{k_i} such that $X_{k_i} \rightarrow X_j$. There is therefore an only d_j in D_j such that $R_{k_i j}(d_{k_i}, d_j)$.

For all $X_j \in X - \mathcal{R}$, there exists consequently at most one consistent instantiation of $\mathcal{R} \cup X_j$ containing I, and obviously at most one solution (containing I).

I can thus be extended to exactly one solution of $\mathcal{P}$. □

As a solution can't obviously be an extension of two different instantiations of $\mathcal{R}$, there is a bijection between the set of all consistent instantiations of $\mathcal{R}$ and the set $\mathcal{S}$ of the solutions of $\mathcal{P}$.

A resolution method appears, that can be decomposed in four phases:

Phase 1: Determination of a minimum root set[2] $\mathcal{R}$ of $\mathcal{G} = (X, C_f)$.

Phase 2: Achievement of the arc and path consistent closure of $\mathcal{P}$.

Phase 3: Instantiation of $\mathcal{R}$.

Phase 4: Instantiation of $X - \mathcal{R}$.

The phase 1 complexity is $O(n + m)$ (determination of the strongly connected components [Tarjan, 1972]); phase 2 is $O(n^3d^3)$ [Mohr and Henderson, 1986]; phase 3 needs $O(d^r)$ to instantiate $\mathcal{R}$, where r is the size of $\mathcal{R}$. At last, the instantiation of $X - \mathcal{R}$ is backtrack free; phase 4 can thus be achieved in $O(n - r)$.

The complexity is so reduced from $O(d^n)$ for usual case, to $O(n^3d^3 + d^r)$.

The size r of the root set is known from phase 1; the resolution complexity of a problem may consequently be bounded prior to the search.

As the problem can be reduced into its root set resolution, a polynomial class appears, characterized by the following property:

Property 3.6 *Let $\mathcal{P} = (X, D, C, R)$. If it owns a root set $\mathcal{R}$ which size is less or equal to 3, then it can be solved in $O(n^3d^3)$.*

Proof: The complexity is due to the computation of $\mathcal{P}^c$, arc and path consistent closure of $\mathcal{P}$.

If it fails, $\mathcal{P}$ admits no solution. Otherwise, as P^c is arc and path consistent, up to 3 variables may be consistently instantiated. As there are at most 3 variables in $\mathcal{R}$, they just have to be instantiated first; after they are instantiated, corollary 3.4 guarantees we will find a solution without backtrack. □

Notice that a complete functional CSP $\mathcal{P} = (X, D, C, R)$ ((X, C_f) is a complete graph) owns at least one root: it can therefore be solved polynomialy. So, unlike non functional CSPs, a complete graph of functional constraints makes the problem more easy.

[2] $\mathcal{R} \subseteq X$ is composed with one element from each source of the quotient graph obtained by reduction of the strongly connected components of $\mathcal{G}$ [David, 1992]

We will now study a subclass of functional constraints: bijective constraints.

4 Bijective constraints

They are defined as follow:

Definition 4.1
A constraint C_{ij} is bijective *iff $X_i \rightarrow X_j$ and $X_j \rightarrow X_i$.*

If all its constraints are bijective, a CSP $\mathcal{P}$ is then called bijective. Otherwise, the set C of the constraints is divided into C_b and C_o, respectively composed with the bijective constraints and the non bijective ones.

We now present definitions of an r-compatible order and a root set for bijective constraints:

Definition 4.2 *Let $\mathcal{G} = (X, Cb)$.*
An order on X is r-compatible iff any variable X_j except the first one owns at least one ancestor X_i in this order such that $C_{ij} \in C_b$.
$\mathcal{R}$ is a root set *of $\mathcal{G}$ iff for all X_i of X, there exists $X_r \in \mathcal{R}$ such that there exists a path from X_r to X_i in C_b.*

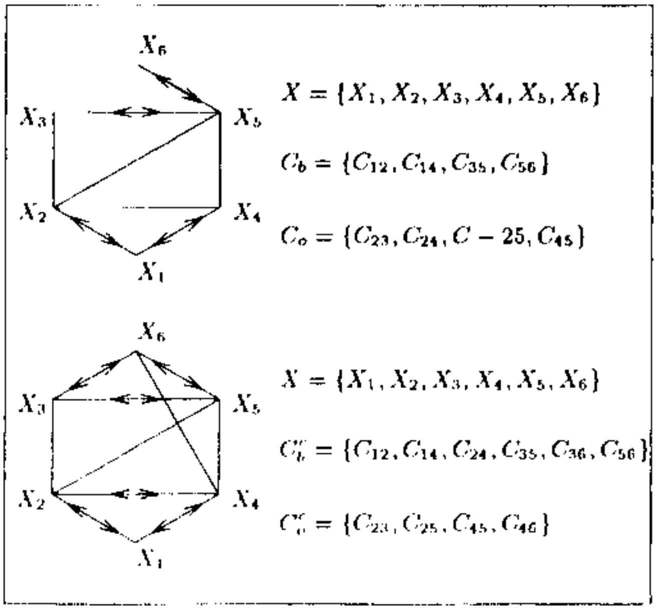
Deville and Van Hentenryck [1991] presented some properties of arc consistency for bijective constraints[3] (in particular AC5, achieving arc consistency in $O(md)$); our purpose is to extend these results to properties of path consistency.

First notice that bijective constraints are particular functional constraints. As such, they own all of their properties (in particular, the size of each relation is bounded by the size of the domain). But bijectivity is a stronger property than functionality, and we can expect this new class to own stronger properties, as property 4.7 shows:

Property 4.7 *Let $\mathcal{P} = (X, D, C_b \cup C_o, R)$ be a CSP, and $P^c = (X, D^c, C^c, R^c)$ its arc and path consistent closure.*
If $\mathcal{G} = (X, C_b)$ is connected, then P^c is complete and bijective, and all domains D_i have the same size.

Proof:
We first prove that P^c is complete and bijective:
For any pair of variables $\{X_i, X_j\}$ of $\mathcal{P}$:
If C_{ij} is a non bijective constraint of C_o or an universal constraint:
C_b is connected; there exists consequently a path μ from X_i to X_j in C_b, $\mu = (X_i = X_{i_0}, X_{i_1}, \ldots, X_{i_{k-1}}, X_{i_k} = X_j)$, such that $\forall p \in [0..k-1]$, $C_{i_p i_{p+1}}$ is bijective.
As every $C_{i_p i_{p+1}}$ is bijective, there exists at most one d_j allowed by this path for each d_i (and inversely, at most one d_i for each d_j).
The constraint C^c_{ij} computed by path consistency is therefore bijective.
If C_{ij} is a bijective constraint, C^c_{ij} will also be bijective (achieving arc and path consistency only suppresses values from domains and tuples from relations).
$\mathcal{P}^c$ is thus complete and bijective.

[3] they called them functional constraints


X_6
X_3
X_5
X_2
X_4
X_1
$X = \{X_1, X_2, X_3, X_4, X_5, X_6\}$
$C_b = \{C_{12}, C_{14}, C_{35}, C_{56}\}$
$C_o = \{C_{23}, C_{24}, C-25, C_{45}\}$
X_6
X_3
X_5
X_2
X_4
X_1
$X = \{X_1, X_2, X_3, X_4, X_5, X_6\}$
$C^c_b = \{C_{12}, C_{14}, C_{24}, C_{35}, C_{36}, C_{56}\}$
$C^c_o = \{C_{23}, C_{25}, C_{45}, C_{46}\}$
Figure 5: are and path consistent closure of a CSP containing bijective constraints

Let us now prove that all domains of P^c have the same size:
For any X_i, X_j in X:
P^c is complete and bijective, so C^c_{ij} is bijective: any d_i of D^c_i owns at most one support in D^c_j (and inversely, any d_j of D^c_j owns at most one support in D^c_i).
Moreover, as $\mathcal{P}^c$ is the arc and path consistent closure of $\mathcal{P}$, C^c_{ij} is arc consistent: any d_i of D^c_i therefore owns exactly one support in D^c_j (and inversely): there is a one-to-one mapping between D^c_j and D^c_i; their sizes are consequently equal.
All domains of $\mathcal{P}^c$ thus have the same size. □

In particular, the arc and path consistent closure of a bijective CSP remains bijective, unlike functional constraints where only the functionality of each constraint was kept. Moreover, as achieving arc and path consistency on a CSP of which graph (X, C_b) is connected induces a complete graph of bijective constraints, any order on X is r-compatible.

We can now present the following property:

Property 4.8 *Any bijective CSP that is arc and path consistent is globally consistent.*

Proof: by induction on the degree of consistency.
Let $\mathcal{P} = (X, D, C, R)$ be bijective, arc and path consistent.
Suppose $\mathcal{P}$ is $(k-1)$-consistent ; is it k-consistent?
Let $I_{k-1} = (d_1, \ldots, d_{k-1})$ be a consistent instantiation of $\{X_1, \ldots, X_{k-1}\}$.
The graph (X, C) is complete and bijective (property 4.7), so any j verifies $X_j \rightarrow X_k$).
Due to property 3.2, I_{k-1} can thus be consistently extended to X_k.
Consequently, $\mathcal{P}$ is k-consistent. □

But the arc and path consistent closure of any $\mathcal{P} = (X, D, C_b \cup C_v, R)$ isn't necessarily complete and bijective (if C_b isn't connected). The problem may though be decomposed, like functional constraints, with a root set of $\mathcal{P}$:

Theorem 2 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent, and $\mathcal{R} \subset X$ a root set of $\mathcal{G} = (X, C_b)$. If there exists a consistent instantiation of $\mathcal{R}$, then it can be extended into a backtrack free solution.*

Proof: like Theorem 1, with bijective constraints instead of functional ones. □

To build a minimum root set, we now just have to choose one (any) element in every connected component of (X, C_b).

Moreover, as well as functional constraints, the number of solutions depends on the root set of $\mathcal{G}$:

Property 4.9 *Let $\mathcal{P} = (X, D, C, R)$ be arc and path consistent, and $\mathcal{R} \subset X$ a root set of $\mathcal{G} = (X, C_b)$. The number of solutions of $\mathcal{P}$ equals the number of consistent instantiations of $\mathcal{R}$.*

Furthermore, if $\mathcal{P}$ is also bijective, there exist exactly d^c solutions, (where d^c is the size of each domain of $\mathcal{P}$).

Proof:

The first part of the property is similar to property 3.5.]f P is bijective, properties 4 8 and 4 7 respectively tell us that P is globally consistent (so any variable is a root set), and that all domains have the same size d' The number of consistent instantiations of R is thus the size of any domain

5 Conclusion

We Presented in this paper some properties of arc and path consistencies for two classes of constraints, functional and bijective constraints, and characterized polynomial problems. We also proposed a decomposition method for these classes, based upon the semantics of the constraints. Yet, many works remain to be done about this subject: in particular, extending this work to non binary constraints, as well as studying other classes of constraints, like monofonic constraints lor instance.

Acknowledgments

I wish to thank Philippe .Janssen and Marie-Catherine Vilarern for their helpful comments

References

|Berge, 1970] C. Berge. Graphes et Hypergraphes. Dunod, 1970.

[David, 1992] Ph. David. Quelques proprietes des consisfances d' are et de chemin pour deux classes de CSP. Research Report 92-007, LIRMM, May 1992.

[Dechter and Pearl, 1987] R. Dechter and J. Pearl. The cycle-cutset method for improving search performance in AI applications. In Proceedings of the Third IEEE Conference on AI applications, pages 224 230, Orlando, Florida, 1987.

[Dechter and Pearl, 1988] R. Dechter and J. Pearl. Network-based heuristics for constraint satisfaction problems. Artificial Intelligence, 34(1): 1-38, 1 1988.

[Dechter and Pearl, 1989] R.Dechter and .J. Pearl. Tree clustering for Constraint Networks. Artificial Intelligence, 38:353 366, 1989.

[Dechter, 1990] R. Dechter. From local to global consistency. In Proceedings of the Eighth Beennal Conference of the Canadian Society for Computational studies of Intelligence, pages 231 237, Ottawa, Canada, 1990.

[Deville and Van Hentenryck, 1991] Y Deville and P. Van Hentenryck. An efficient arc consistency algorithm for a class of CSP problems. In Proceedings of the Twelfth International Joint Conference on Arti - ficial Intelligence, pages 325 330, Sydney, Australia, 1991.

[Freuder, 1978] F. C. Freuder. Synthesizing Constraint Expressions. Communications of the ACM, 21(11):958 966, November 1978.

[Haralick and Elliot, 1980] R. M. llaralick and G. L. Elliot Increasing tree search efficiency for constraint satisfaction problems. Artificial Intelligence, 14:263-313, 1980.

[Janssen et a(, 1990] Ph. Janssen, Ph. Jegou, B. Nouguier. M.-C. Vilarern, and B. Castro. SYNTHIA: Assisted design of peptide synthesis plans. New Journal of Chemistry, I4(I2»:969 976. 1990.

[Jogou, 1990] Ph. Jegou. Cyclic clustering: a compromise between tree clustering and cycle cutset method for improving search efficiency. In Proceedings oj the European Conference on Artificial Intelligence, pages 369 371, Stockholm, Sweden, 1990.

[Mackworth, 1977] A. K. Mackworth Consistency in networks of relations. Artificial Intelligence,8(I):99 Us. 1977

[Mohr and Henderson, 1986] R Mohr and T. C Hen derson Arc and Pat h Consistency Revisited Artificial Intelligence. 28:225 233, 1986.

[Montanari, 1974] U Montanari. Networks of Constraints: fundamental properties and applications to picture processing. Information Sciences, 7:95-132, 1974.

[Perlm, 1992] M. Perlin. Arc consistency for factorable relations. Artificial Intelligence, 53:329 342, 1992.

[Stallman and Sussman, 1977] R. M. Stallman and C. J. Sussman. Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis. Artificial intelligence, 9(2): 135-196, 1977.

[Tarjan, 1972] R Tarjan. Depth-first search and linear Graph algorithms. SIAM Journal on Computing, 1(2). 146 160, June 1972.

[I'llman, 1982] J. D. Ullman. Principles of database systems. Computer Science Press, 1982.

[van Beek, 1992] P. van Beek. On the Minimality and Decomposability of Constraint Networks. In Proceedings of the Penth National Conference on Artificial Intelligence, pages 447-452, San Jose, California, 1992.