

DM841  
DISCRETE OPTIMIZATION

Part I  
**Further Notions of Local Consistency**

Marco Chiarandini

Department of Mathematics & Computer Science  
University of Southern Denmark

1. Higher Order Consistencies
2. Weaker arc consistencies

1. Higher Order Consistencies

2. Weaker arc consistencies

- ▶ arc consistency does not remove all inconsistencies: even if a CSP is arc consistent there might be no solution
- ▶ arc consistency deals with each constraint separately
- ▶ stronger consistencies techniques are studied:
  - ▶ path consistency (generalizes arc consistency to arbitrary binary constraints)
  - ▶ restricted path consistency
  - ▶  $k$ -consistency
  - ▶  $(i, j)$ -consistent

# Path consistency

Given  $\mathcal{P} = \langle X, \mathcal{D}, \mathcal{C} \rangle$  normalized:

- ▶ Given two variables  $x_i, x_j$ , the pair  $(v_i, v_j) \in D(x_i) \times D(x_j)$  is  $p$ -path consistent iff for all  $Y = (x_i = x_{k_1}, x_{k_2}, \dots, x_{k_p} = x_j)$  with  $C_{k_q, k_{q+1}} \in \mathcal{C}$   
 $\exists \tau : \tau[Y] = (v_i = v_{k_1}, \dots, v_{k_p} = v_j) \in \pi_Y(\mathcal{D})$  and  
 $(v_{k_q}, v_{k_{q+1}}) \in C_{k_q, k_{q+1}}, q = 1, \dots, p - 1$
- ▶ the CSP  $\mathcal{P}$  is  $p$ -path consistent iff for any  $(x_i, x_j), i \neq j$  any locally consistent pair of values (ie, satisfying all binary constraints between  $x_i, x_j$ ) is  $p$ -path consistent.

## Example

$$\mathcal{P} = \langle X = (x_1, x_2, x_3), D(x_i) = \{1, 2\}, \mathcal{C} \equiv \{x_1 \neq x_2, x_2 \neq x_3\} \rangle$$

Not path consistent: e.g., for  $(x_1, 1), (x_3, 2)$  there is no  $x_2$

$\mathcal{P} = \langle X, \mathcal{D}, \mathcal{C} \cup \{x_1 = x_3\} \rangle$  is path consistent (local consistency of  $x_1, x_3$  removes values  $x_1 \neq x_3$ )

Alternative definition:

- ▶ constraint composition:

$$C_{x_1, x_3} = C_{x_1, x_2} \cdot C_{x_2, x_3} = \{(a, b) \mid \exists c, (a, c) \in C_{x_1, x_2}, (c, b) \in C_{x_2, x_3}\}$$

- ▶ A normalized CSP  $\mathcal{P}$  is **2-path consistent** if for each subset  $\{x_1, x_2, x_3\}$  of its variables we have  $C_{x_1, x_3} \subseteq C_{x_1 x_2} \cdot C_{x_2 x_3}$
- ▶ Note: the sequence is arbitrary and the order irrelevant hence 6 conditions needs to be considered
- ▶ A CSP without binary constraints is trivially path consistent

Path Consistency rule 1 (propagator):

$$\frac{\langle C_{xy}, C_{xz}, C_{yz}; x \in D(x), y \in D(y), z \in D(z) \rangle}{\langle C'_{xy}, C_{xz}, C_{yz}; x \in D(x), y \in D(y), z \in D(z) \rangle}$$

where  $C'_{xy} := C_{xy} \cap C_{xz} \cdot C_{zy}$

Path Consistency rule 2 (propagator):

$$\frac{\langle C_{xy}, C_{xz}, C_{yz}; x \in D(x), y \in D(y), z \in D(z) \rangle}{\langle C_{xy}, C'_{xz}, C_{yz}; x \in D(x), y \in D(y), z \in D(z) \rangle}$$

where  $C'_{xz} := C_{xz} \cap C_{xy} \cdot C_{yz}$

Path Consistency rule 3 (propagator):

$$\frac{\langle C_{xy}, C_{xz}, C_{yz}; x \in D(x), y \in D(y), z \in D(z) \rangle}{\langle C_{xy}, C_{xz}, C'_{yz}; x \in D(x), y \in D(y), z \in D(z) \rangle}$$

where  $C'_{yz} := C_{yz} \cap C_{yx} \cdot C_{xz}$

## Example

$$\langle x < y, y < z, x < z; x \in [0..4], y \in [1..5], z \in [6..10] \rangle$$

is path consistent. Indeed:

$$C_{x,z} = \{(a, c) \mid a < c, a \in [0..4], c \in [6..10]\}$$

$$C_{x,y} = \{(a, b) \mid a < b, a \in [0..4], b \in [1..5]\}$$

$$C_{y,z} = \{(b, c) \mid b < c, b \in [1..5], c \in [6..10]\}$$

## Example

$$\langle x < y, y < z, x < z; x \in [0..4], y \in [1..5], z \in [5..10] \rangle$$

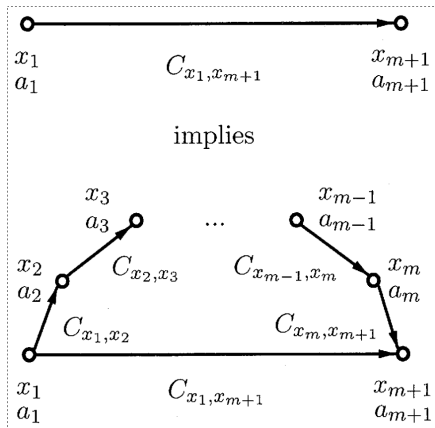
is not path consistent. Indeed:

$C_{x,z} = \{(a, c) \mid a < c, a \in [0..4], c \in [5..10]\}$  and for  $4 \in [0..4]$  and  $5 \in [5..10]$  no  $b \in [1..5]$  such that  $4 < b$  and  $b < 5$ .



# p-path consistency

The p-path consistency defined earlier generalizes 2-path consistency:



2-path consistency if the path has length 2

- ▶ CSP is  $p$ -path consistent  $\iff$  2-path consistent (Montanari, 1974).  
Proof by induction.
- ▶ Hence, sufficient to enforce consistency on paths of length 2.
- ▶ path consistency algorithms work with path of length two only and, like AC algorithms, make these paths consistent with revisions.
- ▶ Even if PC eliminates more inconsistencies than AC, seldom used in practice because of efficiency issues
- ▶ PC requires extensional representation of constraints and hence huge amount memory.
- ▶ Restricted PC does AC and PC only when a variable is left with one value.

# $k$ -consistency

Given  $\mathcal{P} = \langle X, \mathcal{D}, \mathcal{C} \rangle$ , and set of variables  $Y \subseteq X$  with  $|Y| = k - 1$ :

- ▶ a locally consistent instantiation  $I$  on  $Y$  is  $k$ -consistent iff for any  $k$ th variable  $x_{i_k} \in X \setminus Y \exists$  a value  $v_{i_k} \in D(x_{i_k}) : I \cup \{x_{i_k}, v_{i_k}\}$  is locally consistent
- ▶ the CSP  $\mathcal{P}$  is  $k$ -consistent iff for all  $Y$  of  $k - 1$  variables any locally consistent  $I$  on  $Y$  is  $k$ -consistent.

## Example

In general CSP, arc-consistent  $\neq$  2-consistent

$$D(x_1) = D(x_2) = \{1, 2, 3\}, \quad x_1 \leq x_2, x_1 \neq x_2$$

arc consistent, every value has a support on one constraint

not 2-consistent,  $x_1 = 3$  cannot be extended to  $x_2$  and  $x_2 = 1$  not to  $x_1$  with both constraints

arc consistency: each binary constraint separately taken is not violated

2-consistency: any constraint is not violated

## Example

$$D(x_i) = \{1, 2\}, i = 1, 2, 3; \mathcal{C} = \{(1, 1, 1), (2, 2, 2)\}$$

is  $\mathcal{P}$  path consistent?

Yes because no binary constraint such that  $X(C) \subseteq Y$

is  $\mathcal{P}$  3-consistent? No, because  $(x_1, 1), (x_2, 2)$  is locally consistent but cannot be extended consistently to  $x_3$ .

## Example

$$\langle D(x) = [1..2], D(y) = [1..2], D(z) = [2..4]; \mathcal{C} = \{x \neq y, x + y = z\} \rangle$$

- ▶ 1-consistent?
- ▶ 2-consistent?
- ▶ 3-consistent?

- ▶ A node consistent normalized CSP is arc consistent iff it is 2-consistent
- ▶ A node consistent normalized binary CSP is path consistent iff it is 3-consistent

That is, if the CSP is **normalized**:

- ▶ node consistency corresponds to 1-consistency
- ▶ arc consistency corresponds to 2-consistency
- ▶ path consistency corresponds to 3-consistency

However, in **general** CSP, no relationship between  $k$ -consistency and  $l$ -consistency for  $k \neq l$  exists:

- ▶ for any  $k > 1$ , there exists an inconsistent CSP on  $k$  variables that is  $(k - 1)$ -consistent but not  $k$ -consistent
- ▶ for any  $k > 2$ , there exists a consistent CSP on  $k$  variables that is not  $(k - 1)$ -consistent but is  $k$ -consistent
- ▶ for any  $k > 2$ , there exists an inconsistent CSP on  $k$  variables that is  $k$ -consistent
- ▶ for any  $k > 2$ , there exists a consistent CSP on  $k$  variables that is not  $k$ -consistent

## Example

- ▶  $\langle x_1 \neq x_2, x_2 \neq x_3, x_1 \neq x_3; x_1 \in \{0, 1\}, x_2 \in \{0, 1\}, x_3 \in \{0, 1\} \rangle$
- ▶  $\langle x_1 \neq x_2, x_1 \neq x_3; x_1 \in \{a, b\}, x_2 \in \{a\}, \dots, x_k \in \{a\} \rangle$  every  $(k - 1)$ -consistent instantiation is a restriction of the consistent instantiation  $(b, a, a, \dots, a)$
- ▶  $\langle x_1 \neq x_2, x_2 \neq x_3, x_1 \neq x_3; x_1 \in \{1\}, x_2 \in \{1\}, x_3 \in \{1\} \rangle$
- ▶  $\langle x_1 \neq x_2, x_2 \neq x_3, x_1 \neq x_3; x_1 \in \{1\}, x_2 \in \{1, 2, 3\}, x_3 \in \{1, 2, 3\} \rangle$

- ▶  $\mathcal{P}$  is **strongly  $k$ -consistent** iff it is  $j$ -consistent  $\forall j \leq k$
- ▶ constructing one requires  $O(n^k d^k)$  time and  $O(n^{k-1} d^{k-1})$  space.
- ▶ if  $\mathcal{P}$  is strongly  $n$ -consistent then it is globally consistent
- ▶  $(i, j)$ -consistent: any consistent instantiation of  $i$  different variables can be extended to a consistent instantiation including any  $j$  additional variables  
 $k$  consistency  $\equiv (k - 1, 1)$  consistent
- ▶ strongly  $(i, j)$ -consistent



1. Higher Order Consistencies

2. Weaker arc consistencies

# Weaker arc consistencies

- ▶ reduce calls to Revise in coarse-grained algorithms (Forward Checking)
- ▶ reduce amount of work of Revise (Bound consistency)

# Directional Arc Consistency

- ▶ Uses some linear ordering on the considered variables.
- ▶ Requires existence of supports only 'in one direction'
- ▶ A binary CSP  $\mathcal{P}$  is directionally arc consistent (DAC) according to ordering  $o = (x_1, \dots, x_{k_n})$  on  $X$ , where  $(k_1, \dots, k_n)$  is a permutation of  $(1, \dots, n)$  iff for all  $C_{x_i, x_j} \in \mathcal{C}$ , if  $x_i <_o x_j$  then  $x_i$  is arc consistent on  $C_{x_i, x_j}$ .
- ▶ CSP is dir. arc consistent if it is closed under application of arc consistency rule 1.

## Example

$$\langle x < y; x \in [2..10], y \in [3..7] \rangle$$

not arc consistent but directionally arc consistent for the order  $(y, x)$

# Forward checking

Given  $\mathcal{P}$  binary and  $Y \subseteq X : |D(x_i)| = 1 \forall x_i \in Y$ :

- ▶  $\mathcal{P}$  is forward checking consistent according to instantiation  $I$  on  $Y$  iff it is locally consistent and for all  $x_i \in Y$ , for all  $x_j \in X \setminus Y$  and for all  $C(x_i, x_j) \in \mathcal{C}$  is arc consistent on  $C(x_i, x_j)$ .

(all constraints between assigned and not assigned variables are consistent.)

- ▶ Example:

$$\langle D(x) = [1..3], D(y) = [2, 3], D(z) = [1..3]; \mathcal{C} = \{x < y, y < z\} \rangle$$

after  $x = 1$

- ▶  $O(ed)$  time (Revise called only once per arc)
- ▶ Extension to non-binary constraints

# Other Lookahead Filtering

Defined only by procedure, not by fixed point definition

Algorithm **partial lookahead** and **full lookahead** (aka Maintaining arc consistency)

```

procedure  $PL(N, Y, x_i)$ ;
1  $FC(N, Y, x_i)$ ;
2 foreach  $j \leftarrow i + 1$  to  $n$  do
3   foreach  $k \leftarrow j + 1$  to  $n \mid c_{jk} \in C_N$  do
4     if not  $Revise(x_j, c_{jk})$  then return false

procedure  $FL(N, Y, x_i)$ ;
5  $FC(N, Y, x_i)$ ;
6 foreach  $j \leftarrow i + 1$  to  $n$  do
7   foreach  $k \leftarrow i + 1$  to  $n, k \neq j \mid c_{jk} \in C_N$  do
8     if not  $Revise(x_j, c_{jk})$  then return false

```

Example:

$$\langle D(x) = [1..3], D(y) = [2, 3], D(z) = [1..3]; C = \{x < y, y < z\} \rangle$$

after  $x = 1$ .PL:  $D(x) = \{1\}, D(y) = \{2\}, D(z) = \{1, 2, 3\}$ .

FL:  $D(x) = \{1\}, D(y) = \{2\}, D(z) = \{3\}$

# Bound consistency

- ▶ domains inherit total ordering on  $\mathbf{Z}$ ,  
 $\min_D(x)$  and  $\max_D(x)$  called **bounds** of  $D(x)$
- ▶ Given  $\mathcal{P}$  and  $C$ ,  
a **bounded support**  $\tau$  on  $C$  is a tuple that satisfies  $C$  and such that for all  $x_i \in X(C)$ ,  $\min_D(x_i) \leq \tau[x_i] \leq \max_D(x_i)$ ,  
that is,  $\tau \in C \cap \pi_{X(C)}(D')$  (instead of  $D$ )

$$D'(x_i) = \{v \in \mathbf{Z} \mid \min_D(x_i) \leq v \leq \max_D(x_i)\}$$

- ▶  $C$  is **bound( $\mathbf{Z}$ ) consistent** iff  $\forall x_i \in X$  its bounds belong to a **bounded support** on  $C$
- ▶  $C$  is **range consistent** iff  $\forall x_i \in X$  all its values belong to a **bounded support** on  $C$
- ▶  $C$  is **bound( $\mathbf{D}$ ) consistent** iff  $\forall x_i \in X$  its bounds belong to a **support** on  $C$

- ▶  $GAC < (\text{bound}(\mathbf{D}), \text{range}) < \text{bound}(\mathbf{Z})$  (strictly stronger)  
bound( $\mathbf{D}$ ) and range are incomparable
- ▶ most of the time gain in efficiency

### Example

$$\text{sum}(x_1, \dots, x_k, k)$$

GAC is NP-complete (reduction from SubSet problem).

But bound( $\mathbf{Z}$ ) is polynomial: test  $\forall 1 \leq i \leq n$ :

$$\min_D(x_i) \geq k - \sum_{j \neq i} \max_D(x_j)$$

$$\max_D(x_i) \leq k - \sum_{j \neq i} \min_D(x_j)$$

To which type of consistency levels saw in class do the consistency levels defined in Gecode (ie, domain, bound and value) correspond?

Generalized arc consistency, bound(**Z**) (check on each constraint) and Forward checking, respectively.



- Apt K.R. (2003). **Principles of Constraint Programming**. Cambridge University Press.
- Barták R. (2001). **Theory and practice of constraint propagation**. In *Proceedings of CPDC2001 Workshop*, pp. 7–14. Gliwice.
- Bessiere C. (2006). **Constraint propagation**. In *Handbook of Constraint Programming*, edited by F. Rossi, P. van Beek, and T. Walsh, chap. 3. Elsevier. Also as Technical Report LIRMM 06020, March 2006.