Foundations of Artificial Intelligence

26. Constraint Satisfaction Problems: Path Consistency

Malte Helmert

University of Basel

April 13, 2022

Foundations of Artificial Intelligence

April 13, 2022 — 26. Constraint Satisfaction Problems: Path Consistency

Constraint Satisfaction Problems: Overview

Chapter overview: constraint satisfaction problems:

- 22.–23. Introduction
- 24.–26. Basic Algorithms
  - 24. Backtracking
  - 25. Arc Consistency
  - 26. Path Consistency
- 27.–28. Problem Structure

26.1 Beyond Arc Consistency

Beyond Arc Consistency: Path Consistency

idea of arc consistency:

- For every assignment to a variable u there must be a suitable assignment to every other variable v.
- If not: remove values of u for which no suitable "partner" assignment to v exists.

$\rightsquigarrow$ tighter unary constraint on u

This idea can be extended to three variables (path consistency):

- For every joint assignment to variables u, v there must be a suitable assignment to every third variable w.
- If not: remove pairs of values of u and v for which no suitable "partner" assignment to w exists.

$\rightsquigarrow$ tighter binary constraint on u and v

German: Pfadkonsistenz

Beyond Arc Consistency: i-Consistency

general concept of i-consistency for $i \geq 2$:

- For every joint assignment to variables $v_1, \dots, v_{i-1}$ there must be a suitable assignment to every i-th variable v_i.
- If not: remove value tuples of $v_1, \dots, v_{i-1}$ for which no suitable "partner" assignment for v_i exists.

$\rightsquigarrow$ tighter $(i-1)$-ary constraint on $v_1, \dots, v_{i-1}$

- 2-consistency = arc consistency
- 3-consistency = path consistency (*)

We do not consider general i-consistency further as larger values than $i = 3$ are rarely used and we restrict ourselves to binary constraints in this course.

(*) usual definitions of 3-consistency vs. path consistency differ when ternary constraints are allowed

26.2 Path Consistency

Path Consistency: Definition

Definition (path consistent)

Let $\mathcal{C} = \langle V, \mathrm{dom}, (R_{uv}) \rangle$ be a constraint network.

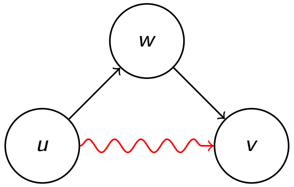
(a) Two different variables $u, v \in V$ are path consistent with respect to a third variable $w \in V$ if for all values $d_u \in \mathrm{dom}(u), d_v \in \mathrm{dom}(v)$ with $\langle d_u, d_v \rangle \in R_{uv}$ there is a value $d_w \in \mathrm{dom}(w)$ with $\langle d_u, d_w \rangle \in R_{uw}$ and $\langle d_v, d_w \rangle \in R_{vw}$.

(b) The constraint network $\mathcal{C}$ is path consistent if for any three variables u, v, w, the variables u and v are path consistent with respect to w.

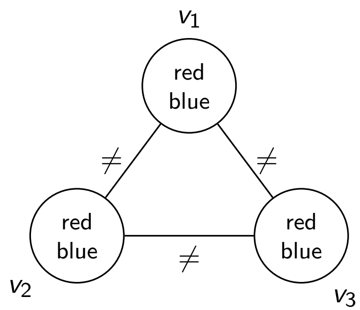
Path Consistency: Remarks

remarks:

- Even if the constraint R_{uv} is trivial, path consistency can infer nontrivial constraints between u and v.
- name "path consistency": path $u \rightarrow w \rightarrow v$ leads to new information on $u \rightarrow v$



Path Consistency: Example



arc consistent, but not path consistent

Processing Variable Triples: revise-3

analogous to revise for arc consistency:

function revise-3($\mathcal{C}, u, v, w$):
$\langle V, \text{dom}, (R_{uv}) \rangle := \mathcal{C}$
for each $\langle d_u, d_v \rangle \in R_{uv}$:
    if there is no $d_w \in \text{dom}(w)$ with
        $\langle d_u, d_w \rangle \in R_{uw}$ **and** $\langle d_v, d_w \rangle \in R_{vw}$:
            remove $\langle d_u, d_v \rangle$ from R_{uv}

input: constraint network $\mathcal{C}$ and three variables u, v, w of $\mathcal{C}$

effect: u, v path consistent with respect to w.
All violating pairs are removed from R_{uv}.

time complexity: $O(k^3)$ where k is maximal domain size

Enforcing Path Consistency: PC-2

analogous to AC-3 for arc consistency:

function PC-2($\mathcal{C}$):
$\langle V, \text{dom}, (R_{uv}) \rangle := \mathcal{C}$
queue $:= \emptyset$
for each set of two variables $\{u, v\}$:
    for each $w \in V \setminus \{u, v\}$:
        insert $\langle u, v, w \rangle$ into *queue*
while *queue* $\neq \emptyset$:
    remove any element $\langle u, v, w \rangle$ from *queue*
    revise-3($\mathcal{C}, u, v, w$)
    if R_{uv} changed in the call to revise-3:
        for each $w' \in V \setminus \{u, v\}$:
            insert $\langle w', u, v \rangle$ into *queue*
            insert $\langle w', v, u \rangle$ into *queue*

PC-2: Discussion

The comments for AC-3 hold analogously.

- PC-2 enforces path consistency
- proof idea: invariant of the **while** loop:
  if $\langle u, v, w \rangle \notin queue$, then u, v path consistent
  with respect to w
- time complexity $O(n^3 k^5)$ for n variables and maximal domain size k (Why?)

26.3 Summary

Summary

- generalization of
  arc consistency (considers pairs of variables)
  to path consistency (considers triples of variables)
  and i-consistency (considers i-tuples of variables)
- arc consistency tightens unary constraints
- path consistency tightens binary constraints
- i-consistency tightens $(i-1)$-ary constraints
- higher levels of consistency more powerful
  but more expensive than arc consistency