# Constraint Satisfaction Problems

## Constraint Networks

Malte Helmert and Stefan Wölfl

Albert-Ludwigs-Universität Freiburg

April 24, 2007

# Constraint Networks

Recall:

**Definition**

A constraint network is a triple

$$\mathcal{C} = \langle V, \mathrm{dom}, C \rangle$$

where:

- $V$ is a non-empty and finite set of variables.
- $\mathrm{dom}$ is a function that assigns a non-empty (value) set (domain) to each variable $v \in V$.
- $C$ is a set of relations over variables of $V$ (constraints), i. e., each constraint is a relation $R_{v_1,\dots,v_n}$ over some variables $v_1, \dots, v_n$ in $V$.

Without loss of generality, we can assume that, for each subset $S$ of variables, $C$ contains only one constraint with scope $S$.

# Example: $4$-Queens Problem



Consider variables $v_1, \dots, v_4$ (associated to the columns of a $4 \times 4$-chess board).

Each of these variables $v_i$ has as its domain $\{1, \dots, 4\}$ (conceived of as the row positions of a queen in column $i$).

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |

Define then binary constraints (thus encoding possible queen movements):

$$R_{v_1,v_2} := \{(1,3),(1,4),(2,4),(3,1),(4,1),(4,2)\}$$

$$R_{v_1,v_3} := \{(1,2),(1,4),(2,1),(2,3),(3,2),(3,4),(4,1),(4,3)\}$$

...

# Instantiation, Solution

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

## Definition

(a) An instantiation of a subset $V'$ of $V$ is an assignment $a : V' \to \bigcup_{v \in V'} \mathrm{dom}(v_i)$ with $a(v_i) \in \mathrm{dom}(v_i)$.

(b) An instantiation $a$ is a partial solution if $a$ satisfies each constraint with scope $S \subseteq V'$ (we also say: $a$ is consistent relative to $\mathcal{C}$).

(c) For an instantiation $a$ of a subset $V' = \{v_1, \ldots, v_k\}$ and a constraint $R$ with scope $S \subseteq V'$, let

$$\overline{a}[S] := (a(v_1), \ldots, a(v_k)).$$

(d) A solution is an instantiation of all variables in $V$ that is consistent relative to $\mathcal{C}$.

# Instantiation, Solution

Note:

(a) An instantiation of variables in $V' \subseteq V$, $a$, is consistent relative to $\mathcal{C}$ iff

$$\overline{a}[S] \in R, \quad \text{for each constraint } R \text{ with } S \subseteq V'.$$

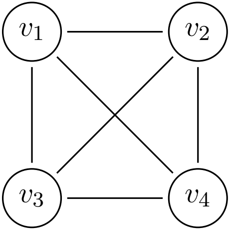
(b) Not each partial solution is part of a (full) solution, i. e., there may be partial solutions of a constraint network that cannot be extended to a solution. For the 4-queens problem, for example,

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ |
|---|---|---|---|---|
| 1 | q | | | |
| 2 | | | q | |
| 3 | | | | |
| 4 | | q | | |

# Example: Primal Constraint Graph



The primal constraint graph for the $4$-queens problem is a complete graph:




Its dual constraint graph? Its hypergraph?

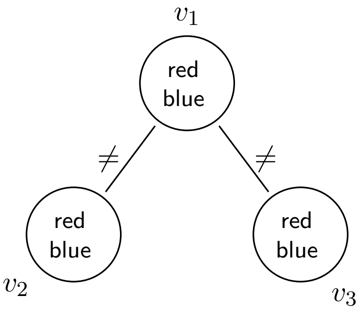
# Equivalence

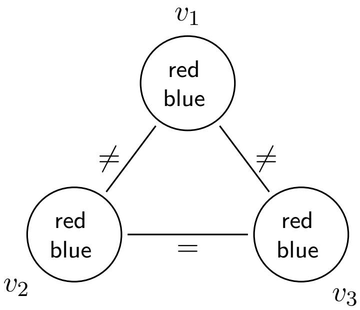
Let $\mathcal{C}$ and $\mathcal{C}'$ be constraint networks on the same set of variables and on the same domains for each variable.

## Definition

$\mathcal{C}$ and $\mathcal{C}'$ are equivalent if each solution of $\mathcal{C}$ is a solution of $\mathcal{C}'$, and vice versa.

Example:



# Tighter Networks and Equivalence



Let $\mathcal{C}$ and $\mathcal{C}'$ be constraint networks on the same set of variables and on the same domains for each variable. We assume that for each set of variables $S$, both networks contain at most one constraint with scope $S$.

**Definition**

$\mathcal{C}'$ is at least as tight as $\mathcal{C}$ if for each constraint $R'$ of $\mathcal{C}'$ with scope $S$, it holds:

(a) $\mathcal{C}$ has no constraint with scope $S$, or

(b) $R' \subseteq R$, where $R$ is the constraint of $\mathcal{C}$ with scope $S$.

Note: $\mathcal{C}'$ may be at least as tight as $\mathcal{C}$ although $|C'| > |C|$.

# Intersection of Networks

Let $\mathcal{C}$ and $\mathcal{C}'$ be constraint networks as above.

## Definition

The intersection of $\mathcal{C}$ and $\mathcal{C}'$, $\mathcal{C} \cap \mathcal{C}'$, is the network defined by intersecting for each scope $S$ of constraints $R_S \in C$ and $R'_S \in C'$ the respective relations, i. e.,

$$R''_S := R_S \cap R'_S.$$

If for a scope $S$ only one of the networks contains a constraint, then we set:

$$R''_S := R_S \quad (\text{or } := R'_S, \text{ resp.})$$

## Lemma

*If $\mathcal{C}$ and $\mathcal{C}'$ are equivalent networks, then $\mathcal{C} \cap \mathcal{C}'$ is equivalent to both networks and at least as tight as both networks.*

# Minimal Network



**Definition**

Let $\mathcal{C}_0$ be a constraint network and let $\mathcal{C}_1 \dots, \mathcal{C}_k$ be the set of *all* constraint network (defined on the same set of variables and the same domains) that are equivalent to $\mathcal{C}_0$.

$$\bigcap_{1 \leq i \leq k} \mathcal{C}_i$$

is the minimal network of $\mathcal{C}_0$.

The minimal network is equivalent to and as least as tight as all the constraint networks $\mathcal{C}_i$. There is no network equivalent to $\mathcal{C}_0$ that is tighter than the minimal network.

# Projecting Relations

Let $R_S$ be a relation with scope $S = \{v_1, \ldots, v_k\}$ (we can conceive of $R_S$ as a constraint network ...).

## Definition

The projection network of $R_S$, $\mathrm{Proj}(R_S)$, is the constraint network defined by

$$\begin{aligned} V &:= S \\ \mathrm{dom}(v_i) &:= \pi_{v_i}(R_S) \\ R_{v_i,v_j} &:= \pi_{v_i,v_j}(R_S) \end{aligned}$$

Note: The projection network is an upper approximation by binary networks in the following sense:

## Lemma

*Any solution of $R_S$ (as a network) defines a solution of $\mathrm{Proj}(R_S)$.*

# Binary Representation



## Definition

A relation $R_S$ with scope $S$ has a binary representation if the relation (conceived of as a network) is equivalent to $\mathrm{Proj}(R_S)$.

From the fact that a relation has a binary representation, it does not follow that all its projections have binary representations as well (Exercise!).

## Definition

A relation $R_S$ with scope $S$ is binary decomposable if the relation itself and all its projections to subsets of $S$ (with at least 3 elements) have a binary representation.

# Literature


Rina Dechter.
Constraint Processing,
Chapter 2, Morgan Kaufmann, 2003