Constraint Relationships

Language Features

Universität
Augsburg
University

Preferences in Constraint Solving

Constraint problem (X, D, C)

- Variables X, Domains $D = (D_x)_{x \in X}$, Constraints C

How to deal with over-constrained problems?

$((\{\mathrm{x}, \mathrm{y}, \mathrm{z}\}, \mathrm{D_x} = \mathrm{D_y} = \mathrm{D_z} = \{1, 2, 3\}), \{\mathrm{c_1}, \mathrm{c_2}, \mathrm{c_3}\})$ mit

$$\mathrm{c_1} : \mathrm{x} + 1 = \mathrm{y}$$
$$\mathrm{c_2} : \mathrm{z} = \mathrm{y} + 2$$
$$\mathrm{c_3} : \mathrm{x} + \mathrm{y} \leq 3$$

- Not all constraints can be satisfied simultaneously

Preferences in Constraint Solving

Institute for Software & Systems Engineering

Constraint problem (X, D, C)

- Variables X, Domains $D = (D_x)_{x \in X}$, Constraints C

How to deal with over-constrained problems?

$((\{\mathrm{x}, \mathrm{y}, \mathrm{z}\}, \mathrm{D_x} = \mathrm{D_y} = \mathrm{D_z} = \{1, 2, 3\}), \{\mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3\})$ mit

$$\mathrm{c}_1 : \mathrm{x} + 1 = \mathrm{y}$$
$$\mathrm{c}_2 : \mathrm{z} = \mathrm{y} + 2$$
$$\mathrm{c}_3 : \mathrm{x} + \mathrm{y} \leq 3$$

- Not all constraints can be satisfied simultaneously
    - e. g., c_2 forces $\mathrm{z} = 3$ and $\mathrm{y} = 1$, conflicting c_1
- We can choose between assignments satisfying $\{\mathrm{c}_1, \mathrm{c}_3\}$ or $\{\mathrm{c}_2, \mathrm{c}_3\}$.

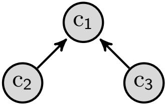
Which assignments $v \in [X \to D]$ should be preferred by an agent/several agents?

Constraint Relationships

Institute for Software & Systems Engineering

Approach (**?**)

- Define relation R over constraints C to denote which constraints are more important than others, e. g.
  - c_1 is more important than c_2
  - c_1 is more important than c_3



Benefits

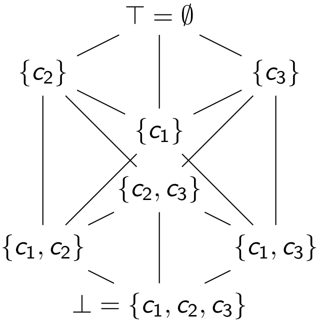
- Qualitative formalism — easy to specify
- Graphical interpretation
  - Semantics (how much more important is a constraint) regulated by
  - dominance properties that are either “hierarchical” or “egalitarian”
  - Single-Predecessors-Dominance (SPD) vs. Transitive-Predecessors-Dominance (TPD)

SoftConstraints in MiniZinc

```
% X: {x,y,z} D_i = {1,2,3}, i in X
%    * c1: x + 1 = y * c2: z = y + 2 * c3: x + y <= 3
% (c) ISSE
% isse.uni-augsburg.de/en/software/constraint-relationships/
include "soft_constraints/minizinc_bundle.mzn";

var 1..3: x; var 1..3: y; var 1..3: z;

% read as "soft constraint c1 is satisfied iff x + 1 = y"
constraint x + 1 = y <-> satisfied[1];
constraint z = y + 2 <-> satisfied[2];
constraint x + y <= 3 <-> satisfied[3];

% soft constraint specific for this model
nScs = 3; nCrEdges = 2;
crEdges = [| 2, 1 | 3, 1 |]; % read c2 is less important than c1

solve minimize penSum; % minimize the sum of penalties
```

Search types

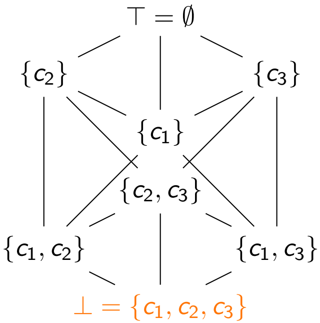
The whole valuation space (partially ordered)



```
%
% Typical Optimization Routine (Branch and Bound):
%
%  1. Look for the first feasible solution
%  2. Impose restrictions on the next feasible solution
%  3. Repeat
```

Search types: Strictly better

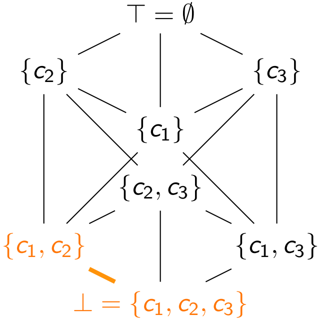
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
      = repeat(
           if next() then
             commit() /\
             post(isWorse(sol(vScs), vScs))
           else break endif     );
```

Search types: Strictly better

The whole valuation space (partially ordered)



$\top = \emptyset$

$\{c_2\}$ $\{c_3\}$

$\{c_1\}$

$\{c_2, c_3\}$

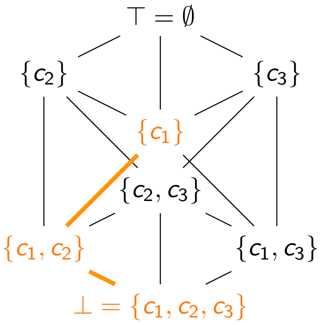
$\{c_1, c_2\}$ $\{c_1, c_3\}$

$\bot = \{c_1, c_2, c_3\}$

```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
      = repeat(
           if next() then
             commit() /\
             post(isWorse(sol(vScs), vScs))
           else break endif     );
```

Search types: Strictly better

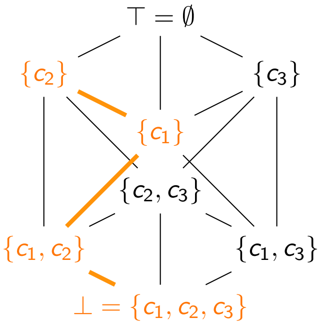
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
     = repeat(
          if next() then
            commit() /\
            post(isWorse(sol(vScs), vScs))
          else break endif     );
```

Search types: Strictly better

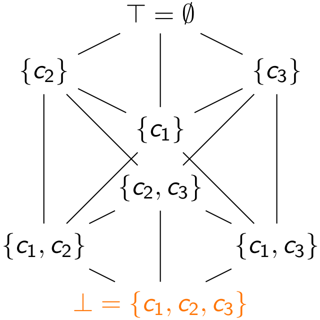
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
      = repeat(
           if next() then
             commit() /\
             post(isWorse(sol(vScs), vScs))
           else break endif     );
```

Search types: Only not dominated

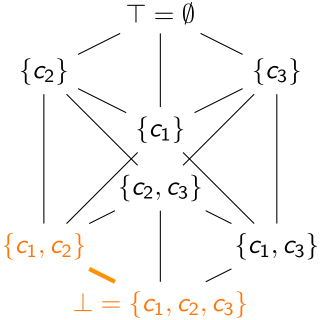
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
      = repeat(
           if next() then
             commit() /\
             post((isWorse(vScs, sol(vScs)) \/ vScs = sol(vScs)))
           else break endif      );
```

Search types: Only not dominated

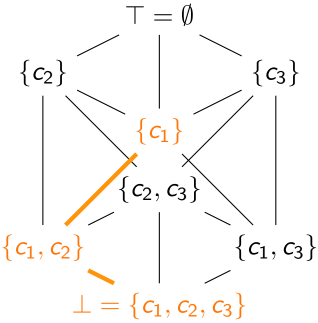
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
     = repeat(
          if next() then
            commit() /\
            post((isWorse(vScs, sol(vScs)) \/ vScs = sol(vScs)))
          else break endif    );
```

Search types: Only not dominated

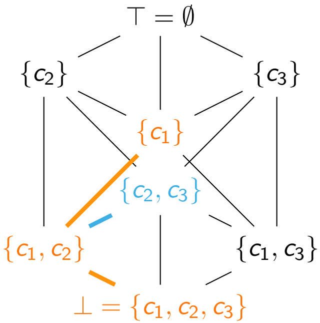
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
     = repeat(
          if next() then
            commit() /\
            post((isWorse(vScs, sol(vScs)) \/ vScs = sol(vScs)))
          else break endif     );
```

Search types: Only not dominated

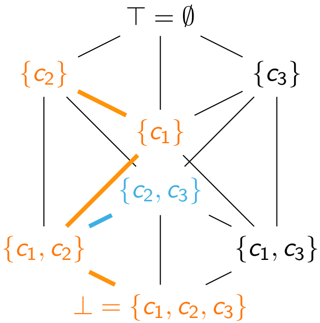
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
      = repeat(
           if next() then
             commit() /\
             post((isWorse(vScs, sol(vScs)) \/ vScs = sol(vScs)))
           else break endif     );
```

Search types: Only not dominated

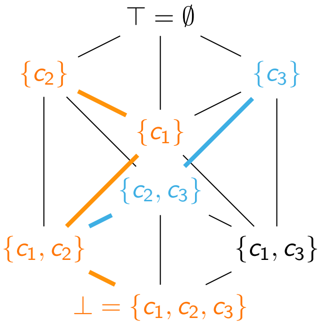
The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
      = repeat(
          if next() then
            commit() /\
            post((isWorse(vScs, sol(vScs)) \/ vScs = sol(vScs)))
          else break endif     );
```

Search types: Only not dominated

The whole valuation space (partially ordered)



```
function ann: strictlyBetterBAB(var set of SOFTCONSTRAINTS: vScs)
     = repeat(
          if next() then
            commit() /\
            post((isWorse(vScs, sol(vScs)) \/ vScs = sol(vScs)))
          else break endif     );
```

Language Features

a

Language Features: Suitable Weighting

a

Language Features: Consistency Checks

a

Language Features: Variable Ordering

a

Language Features: Redundant Constraints

a

Language Features: Custom Search

a

Quellen I