Constraint Logic Programming

Sylvain Soliman

Sylvain.Soliman@inria.fr

Project-Team LIFEWARE

MPRI 2.35.1 Course – September–November 2017

Part I: CLP - Introduction and Logical Background

1. The Constraint Programming paradigm
2. Examples and Applications
3. First Order Logic
4. Models
5. Logical Theories

Part II: Constraint Logic Programs

6. Constraint Languages
7. CLP($\mathcal{X}$)
8. CLP($\mathcal{H}$)
9. CLP($\mathcal{R}, \mathcal{FD}, \mathcal{B}$)

Part III: CLP - Operational and Fixpoint Semantics

Full abstraction

Theorem 1 ([JL87popl])

$T_P^{\mathcal{X}} \uparrow \omega = O_{gs}(P)$

$T_P^{\mathcal{X}} \uparrow \omega \subset O_{gs}(P)$ is proved by induction on the powers n of $T_P^{\mathcal{X}}$.
$n = 0$, i.e., $\emptyset$, is trivial. Let $A\rho \in T_P^{\mathcal{X}} \uparrow n$, there exists a rule $(A \leftarrow c|A_1, \ldots, A_n) \in P$, s.t. $\{A_1\rho, \ldots, A_n\rho\} \subset T_P^{\mathcal{X}} \uparrow n - 1$ and $\mathcal{X} \models c\rho$. By induction $\{A_1\rho, \ldots, A_n\rho\} \subset O_{gs}(P)$. By definition of O_{gs} and $\wedge$-compositionality. we get $A\rho \in O_{gs}(P)$.
$O_{gs}(P) \subset T_P^{\mathcal{X}} \uparrow \omega$ is proved by induction on the length of derivations. Successes with derivation of length 0 are ground facts in $T_P^{\mathcal{X}} \uparrow 1$.
Let $A\rho \in O_{gs}(P)$ with a derivation of length n. By definition of O_{gs} there exists $(A \leftarrow c|A_1, \ldots, A_n) \in P$ s.t. $\{A_1\rho, \ldots, A_n\rho\} \subset O_{gs}(P)$ and $\mathcal{X} \models c\rho$. By induction $\{A_1\rho, \ldots, A_n\rho\} \subset T_P^{\mathcal{X}} \uparrow \omega$. Hence by definition of $T_P^{\mathcal{X}}$ we get $A\rho \in T_P^{\mathcal{X}} \uparrow \omega$.

Part IV: Logical Semantics

Soundness of CSLD Resolution

Theorem 2 ([JL87popl])

If c is a computed answer for the goal G then $M_P^{\mathcal{X}} \models c \supset G$, $P \models_{\mathcal{X}} c \supset G$ and $P, \mathcal{T} \models c \supset G$.

If $G = (d|A_1, \ldots, A_n)$, we deduce from the $\wedge$-compositionality lemma, that there exist computed answers $c_1, \ldots, c_n$ for the goals $A_1, \ldots, A_n$ such that $c = d \wedge \bigwedge_{i=1}^{n} c_i$ is satisfiable. For every $1 \leq i \leq n$
$c_i|A_i \in S_P^{\mathcal{X}} \uparrow \omega$,
$[c_i|A_i]_{\mathcal{X}} \subset M_P^{\mathcal{X}}$, hence $M_P^{\mathcal{X}} \models \forall(c_i \supset A_i)$,
$P \models_{\mathcal{X}} \forall(c_i \supset A_i)$ as $M_P^{\mathcal{X}}$ is the least $\mathcal{X}$-model of P,
$P \models_{\mathcal{X}} \forall(c \supset A_i)$ as $\mathcal{X} \models \forall(c \supset c_i)$ for all i, $1 \leq i \leq n$.
Therefore we have $P \models_{\mathcal{X}} \forall(c \supset (d \wedge A_1 \wedge \cdots \wedge A_n))$,
and as the same reasoning applies to any model $\mathcal{X}$ of $\mathcal{T}$,
$P, \mathcal{T} \models \forall(c \supset (d \wedge A_1 \wedge \cdots \wedge A_n))$

Completeness of CSLD resolution

Theorem 3 ([Maher87iclp])

If $M_P^{\mathcal{X}} \models c \supset G$ then there exists a set $\{c_i\}_{i \geq 0}$ of computed answers for G, such that: $\mathcal{X} \models \forall (c \supset \bigvee_{i \geq 0} \exists Y_i c_i)$.

Proof.

For every solution ρ of c, for every atom A_j in G,
$M_P^{\mathcal{X}} \models A_j\rho$ iff $A_j\rho \in T_P^{\mathcal{X}} \uparrow \omega$, iff $A_j\rho \in [S_P^{\mathcal{X}} \uparrow \omega]_{\mathcal{X}}$
iff $c_{j,\rho}|A_j \in S_P^{\mathcal{X}} \uparrow \omega$, for some constraint $c_{j,\rho}$ s.t. ρ is solution of $\exists Y_{j,\rho} c_{j,\rho}$,
where $Y_{j,\rho} = V(c_{j,\rho}) \setminus V(A_j)$,
iff $c_{j,\rho}$ is a computed answer for A_j and $\mathcal{X} \models \exists Y_{j,\rho} c_{j,\rho} \rho$.
Let c_ρ be the conjunction of $c_{j,\rho}$ for all j. c_ρ is a computed answer for G.
By taking the collection of c_ρ for all ρ we get $\mathcal{X} \models \forall (c \supset \bigvee_{c_\rho} \exists Y_\rho c_\rho)$ □

Completeness w.r.t. the theory of the structure

Theorem 4 ([Maher87iclp])

If $P, \mathcal{T} \models c \supset G$ then there exists a finite set $\{c_1, \ldots, c_n\}$ of computed answers to G, such that:
$\mathcal{T} \models \forall(c \supset \exists Y_1 c_1 \vee \cdots \vee \exists Y_n c_n)$.

Proof.

If $P, \mathcal{T} \models c \supset G$ then for every model $\mathcal{X}$ of $\mathcal{T}$, for every $\mathcal{X}$-solution ρ of c, there exists a computed constraint $c_{\mathcal{X},\rho}$ for G s.t. $\mathcal{X} \models c_{\mathcal{X},\rho}\rho$.
Let $\{c_i\}_{i\geq 1}$ be the set of these computed answers. Then for every model $\mathcal{X}$ and for every $\mathcal{X}$-valuation ρ, $\mathcal{X} \models c \supset \bigvee_{i\geq 1} \exists Y_i c_i$,
therefore $\mathcal{T} \models c \supset \bigvee_{i\geq 1} \exists Y_i c_i$,
As $\mathcal{T} \cup \{\exists(c \wedge \neg\exists Y_i c_i)\}_i$ is unsatisfiable, by applying the compactness theorem of first-order logic there exists a finite part $\{c_i\}_{1\leq i\leq n}$, s.t. $\mathcal{T} \models c \supset \bigvee_{i=1}^{n} \exists Y_i c_i$. □

Part V: Constraint Solving

Reified constraints in CLP($\mathcal{B}, \mathcal{FD}$)

The reified constraint $B \Leftrightarrow (X < Y)$
associates a boolean variable B to the satisfaction of the constraint $X < Y$

Arc consistency:
B is set to 1 when

Reified constraints in CLP($\mathcal{B}, \mathcal{FD}$)

The reified constraint $B \Leftrightarrow (X < Y)$
associates a boolean variable B to the satisfaction of the constraint $X < Y$

Arc consistency:
B is set to 1 when $domain(X) < domain(Y)$,
B is set to 0 when $domain(Y) \leq domain(X)$
$domain(X)$ is set to

Reified constraints in CLP($\mathcal{B}, \mathcal{FD}$)

The reified constraint $B \Leftrightarrow (X < Y)$
associates a boolean variable B to the satisfaction of the constraint $X < Y$

Arc consistency:
B is set to 1 when $domain(X) < domain(Y)$,
B is set to 0 when $domain(Y) \leq domain(X)$
$domain(X)$ is set to $\{v \in domain(X) \mid v < max(Y)\}$ when $B = 1$,
$domain(Y)$ is set to $\{v \in domain(Y) \mid v > min(X))\}$ when $B = 1$,
$domain(X)$ is set to $\{v \in domain(X) \mid v \geq min(Y)\}$ when $B = 0$,
$domain(Y)$ is set to $\{v \in domain(Y) \mid v \leq max(X))\}$ when $B = 0$

Cardinality constraint

Cardinality constraint $card(N, [C_1, \ldots, C_m])$ is true iff there are exactly N constraints true in $[C_1, \ldots, C_m]$.

```
card(0, []).
card(N, [C | L]) :-
   B in 0..1,
   B #<=> C,
   N #= B + M,
   card(M, L).
```

Time Tabling

The organizers of a congress have 3 rooms and 2 days for eleven half-day sessions. Sessions AJ, JI, IE, CF, BHK, ABCH, DFJ can't be simultaneous, moreover $E < J$, $D < K$, $F < K$

Time Tabling

The organizers of a congress have 3 rooms and 2 days for eleven half-day sessions. Sessions AJ, JI, IE, CF, BHK, ABCH, DFJ can't be simultaneous, moreover $E < J$, $D < K$, $F < K$

```
| ?- [A,B,C,D,E,F,G,H,I,J,K] ins 1..4,
   all_different([A,J]),all_different([J,I]),
   all_different([I,E]),all_different([B,H,K]),
   all_different([A,B,C,H]),all_different([D,F,J]),
   J#>E, K#>D, K#>F,
   atmost(3,[A=1,B=1,C=1,D=1,E=1,F=1,G=1,H=1,I=1,J=1,K=1]),
   atmost(3,[A=2,B=2,C=2,D=2,E=2,F=2,G=2,H=2,I=2,J=2,K=2]),
   atmost(3,[A=3,B=3,C=3,D=3,E=3,F=3,G=3,H=3,I=3,J=3,K=3]),
   atmost(3,[A=4,B=4,C=4,D=4,E=4,F=4,G=4,H=4,I=4,J=4,K=4]),
   labeling([A,B,C,D,E,F,G,H,I,J,K]).

A=1, B=2, C=4, D=1, E=2, F=2, G=4, H=3, I=1, J=3, K=4 ?
```

Magic Series

Find a sequence of integers $(i_0, \ldots, i_{n-1})$ such that i_j is the number of occurrences of the integer j in the sequence

Magic Series

Find a sequence of integers $(i_0, \ldots, i_{n-1})$ such that i_j is the number of occurrences of the integer j in the sequence

$$\bigwedge_{j=0}^{n-1} card(i_j, [i_0 = j, \ldots, i_{n-1} = j])$$

- Constraint propagation with reified constraints $b_k \Leftrightarrow i_k = j$,

Magic Series

Find a sequence of integers $(i_0, \ldots, i_{n-1})$ such that
i_j is the number of occurrences of the integer j in the sequence

$$\bigwedge_{j=0}^{n-1} card(i_j, [i_0 = j, \ldots, i_{n-1} = j])$$

- Constraint propagation with reified constraints $b_k \Leftrightarrow i_k = j$,
- Redundant constraints

Magic Series

Find a sequence of integers $(i_0, \ldots, i_{n-1})$ such that
i_j is the number of occurrences of the integer j in the sequence

$$\bigwedge_{j=0}^{n-1} card(i_j, [i_0 = j, \ldots, i_{n-1} = j])$$

- Constraint propagation with reified constraints $b_k \Leftrightarrow i_k = j$,
- Redundant constraints $n = \sum_{j=0}^{n-1} i_j$,
- Enumeration with first fail heuristics,
- Less than one second CPU for $n = 50$...

Multiple Modeling in CLP($\mathcal{FD}$)

N-queens with two concurrent models: by lines and by columns

```
queens2(N, L) :-
   length(Column, N), Column ins 1..N, safe(Column),
   length(Line, N),   Line ins 1..N,   safe(Line),
   linking(Line, 1, Column),
   append(Line, Column,L), labeling([ff], L).
linking([], _, _).
linking([X | L], I, C) :-
   equivalence(X, I, C, 1),
   I1 is I + 1,
   linking(L, I1, C).
equivalence(_, _, [], _).
equivalence(X, I, [Y | L], J) :-
   B #<=> (X#=J), B #<=> (Y#=I),
   J1 is J + 1,
equivalence(X, I, L, J1).
```

Lexicographic order constraint

`lex([X1,...,Xn])`
iff $X_1 < X_2$ or ($X_1 = X_2$ and ($X_2 < X_3$ ... or $X_{n-1} \leq X_n$))

Lexicographic order constraint

`lex([X1,...,Xn])`

iff $X_1 < X_2$ or ($X_1 = X_2$ and ($X_2 < X_3$ ... or $X_{n-1} \leq X_n$))

```
lex(L):-
    lex(L, B),
    B = 1.

lex([], 1).

lex([_], 1).

lex([X, Y | L], R):-
    B #<=> (X #< Y),
    C #<=> (X #= Y),
    lex([Y | L], D),
    R #<=> B #\/ (C #/\ D).
```

Programming in CLP($\mathcal{H}, \mathcal{B}, \mathcal{FD}, \mathcal{R}$)

- Basic constraints on domains of terms $\mathcal{H}$, bounded integers $\mathcal{FD}$, reals $\mathcal{R}$, booleans $\mathcal{B}$, ontologies $\mathcal{H}_{\leq}$, etc.
- Relations defined *extensionally* by constrained facts:

```
precedence(X, D, Y) :- X + D #< Y.
disjonctives(X, D, Y, E) :- X + D #< Y.
disjonctives(X, D, Y, E) :- Y + E #< X.
```

  and *intentionally* by rules:

```
labeling([]).
labeling([X | L]) :-
   fd_dom(X, D), member(X, D), labeling(L).
```

- Programming of search procedures and heuristics:
  And-parallelism (variable choice): "first-fail" heuristics min domain
  Or-parallelism (value choice): "best-first" heuristics min value

Part VI

Practical CLP Programming

Part VI: Practical CLP Programming

The Warren Abstract Machine

First Prolog implementation in the early 70’s (by Colmerauer et al.).

In 1983, David H. Warren creates the Warren Abstract Machine.

Remains the state of the art (for term representation, basic instructions, …)

Slightly extended for CLP

(C)SLD resolution seen as a call stack (with marks for choice points)

The Warren Abstract Machine

First Prolog implementation in the early 70's (by Colmerauer et al.).

In 1983, David H. Warren creates the Warren Abstract Machine.

Remains the state of the art (for term representation, basic instructions, …)

Slightly extended for CLP (constraints instead of substitutions)

(C)SLD resolution seen as a call stack (with marks for choice points)

Optimizations from the WAM

Search for predicates should be almost in constant time

Use a hash table - indexing - for the predicate name/arity,

Each call normally adds a frame to the call stack (removed on backtracking)

As for other programming paradigms, not always necessary

Optimizations from the WAM

Search for predicates should be almost in constant time

Use a hash table - indexing - for the predicate name/arity,
and the functor of the first argument

Each call normally adds a frame to the call stack (removed on backtracking)

As for other programming paradigms, not always necessary

Optimizations from the WAM

Search for predicates should be almost in constant time

Use a hash table - indexing - for the predicate name/arity, and the functor of the first argument

Each call normally adds a frame to the call stack (removed on backtracking)

As for other programming paradigms, not always necessary

Tail recursion can be optimized,

Optimizations from the WAM

Search for predicates should be almost in constant time

Use a hash table - indexing - for the predicate name/arity, and the functor of the first argument

Each call normally adds a frame to the call stack (removed on backtracking)

As for other programming paradigms, not always necessary

Tail recursion can be optimized, when calling and called contexts are deterministic.

Putting it all together

Naive sum

```
sum(0, []).
sum(S, [H | T]) :-
    sum(S1, T),
    S is S1 + H.
```

Putting it all together

Naive sum

```
sum(0, []).
sum(S, [H | T]) :-
    sum(S1, T),
    S is S1 + H.
```

Much better

```
sum(S, L) :-
    sum_aux(L, 0, S).

sum_aux([], S, S).
sum_aux([H | T], S0, S) :-
    S1 is S0 + H,
    sum_aux(T, S1, S).
```

Putting it all together

If numbers are coded as the fact `number(X)`?

```
sum(S) :- findall(X, number(X), L), sum(S, L).
```

Putting it all together

If numbers are coded as the fact `number(X)`?

```
sum(S) :- findall(X, number(X), L), sum(S, L).
```

```
sum(S) :-
   g_assign(sum, 0),          % nb_setval/assert
   (
      number(N),
      g_read(sum, S1),
      S2 is S1 + N,
      g_assign(sum, S2),
      fail
   ;
      g_read(sum, S)          % nb_getval/retract
   ).
```

Cutting choice-points

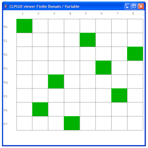
```
try(S) :-
   stream_property(S,
                   input),
   (
      repeat,
      read_term(S, G),
      call(G),
      ground(G),
      !,
      write(G)
   ).
try(S) :-
   ...
```

Cutting choice-points

```prolog
try(S) :-
   stream_property(S,
                   input),
   (
      repeat,
      read_term(S, G),
      call(G),
      ground(G),
      !,
      write(G)
   ).
try(S) :-
   ...
```

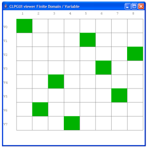
```prolog
try(S) :-
   stream_property(S,
                   input),
   (
      repeat,
      read_term(S, G),
      call(G),
      ground(G)
   ->
      write(G)
   ).
try(S) :-
   ...
```

Symmetries in the N-queens problem



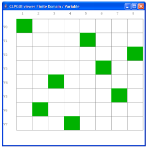
queens(N,[X1,...XN])
iff

Symmetries in the N-queens problem



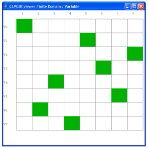
queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*

Symmetries in the N-queens problem



queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry

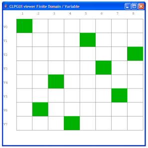
Symmetries in the N-queens problem



queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry

iff queens(N,[N+1-X1,...,N+1-XN]) *horizontal axis symmetry*

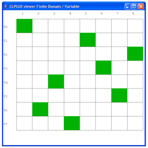
Symmetries in the N-queens problem



queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry

iff queens(N,[N+1-X1,...,N+1-XN]) *horizontal axis symmetry*
value symmetry

Symmetries in the N-queens problem

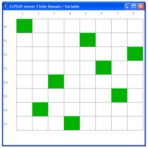


queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry

iff queens(N,[N+1-X1,...,N+1-XN]) *horizontal axis symmetry*
value symmetry

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=N+1-i *rotation symmetry*
variable-value symmetry

Symmetries in the N-queens problem



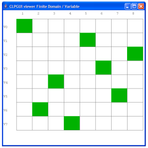
queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry

iff queens(N,[N+1-X1,...,N+1-XN]) *horizontal axis symmetry*
value symmetry

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=N+1-i *rotation symmetry*
variable-value symmetry

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=i *rotation symmetry*
variable-value symmetry

Symmetries in the N-queens problem



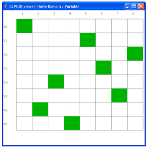
queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry broken by X1<XN

iff queens(N,[N+1-X1,...,N+1-XN]) *horizontal axis symmetry*
value symmetry

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=N+1-i *rotation symmetry*
variable-value symmetry

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=i *rotation symmetry*
variable-value symmetry

Symmetries in the N-queens problem



queens(N,[X1,...XN])
iff queens(N,[XN,...,X1]) *vertical axis symmetry*
variable symmetry broken by X1<XN

iff queens(N,[N+1-X1,...,N+1-XN]) *horizontal axis symmetry*
value symmetry broken by X1<5

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=N+1-i *rotation symmetry*
variable-value symmetry

iff queens(N,[Y1,...,YN]) where Xi=j iff Yj=i *rotation symmetry*
variable-value symmetry

Variable Symmetries

Given a Constraint Satisfaction Problem $c(x_1, ..., x_n)$ over $\mathcal{X}$ a variable symmetry σ is a bijection on variables that preserves solutions:

$$\mathcal{X} \models c(x_1, ..., x_n) \text{ iff } \mathcal{X} \models c(x_{\sigma(1)}, ..., x_{\sigma(n)})$$

Proposition 5 ([Crawford96kr])

If $(\mathcal{X}, \leq)$ is an order, all variable symmetries can be broken by the global constraint

$$\bigwedge_{\sigma \in \Sigma} [x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}]$$

Variable Symmetries

Given a Constraint Satisfaction Problem $c(x_1, ..., x_n)$ over $\mathcal{X}$ a variable symmetry σ is a bijection on variables that preserves solutions:

$$\mathcal{X} \models c(x_1, ..., x_n) \text{ iff } \mathcal{X} \models c(x_{\sigma(1)}, ..., x_{\sigma(n)})$$

Proposition 5 ([Crawford96kr])

If $(\mathcal{X}, \leq)$ is an order, all variable symmetries can be broken by the global constraint

$$\bigwedge_{\sigma \in \Sigma} [x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}]$$

Proof.

This is one way to choose a unique member in each equivalence class of symmetric assignments. □

Variable Symmetry Breaking

Global constraint $[x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}]$
arc consistent (AC) if for every variable, every value in its domain belongs to a solution

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1,...,x_n] \leq_{lex} [x_{\sigma(1)},...,x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma\in\Sigma} AC([x_1,...,x_n] \leq_{lex} [x_{\sigma(1)},...,x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0,1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_2,x_4,x_1,x_3])$ and $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_4,x_3,x_1,x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$	$[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$	$[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$																		

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma}[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma \in \Sigma} AC([x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$ $[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$ $[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1										

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma\in\Sigma} AC([x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$	$[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$	$[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0				0	1			0	0				0	1		
$x_1 = 1$																		

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1,\dots,x_n] \leq_{lex} [x_{\sigma(1)},\dots,x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma\in\Sigma} AC([x_1,\dots,x_n] \leq_{lex} [x_{\sigma(1)},\dots,x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0,1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_2,x_4,x_1,x_3])$ and $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_4,x_3,x_1,x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1								

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1,\dots,x_n] \leq_{lex} [x_{\sigma(1)},\dots,x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma\in\Sigma} AC([x_1,\dots,x_n] \leq_{lex} [x_{\sigma(1)},\dots,x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0,1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_2,x_4,x_1,x_3])$ and $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_4,x_3,x_1,x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$																

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma\in\Sigma} AC([x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1										

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1,\ldots,x_n] \leq_{lex} [x_{\sigma(1)},\ldots,x_{\sigma(n)}])$ *is strictly stronger than* $\bigwedge_{\sigma\in\Sigma} AC([x_1,\ldots,x_n] \leq_{lex} [x_{\sigma(1)},\ldots,x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0,1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_2,x_4,x_1,x_3])$ and $AC([x_1,x_2,x_3,x_4] \leq_{lex} [x_4,x_3,x_1,x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$ $[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$ $[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$																

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma}[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma \in \Sigma} AC([x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$	0	1			1											

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma\in\Sigma}[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma\in\Sigma} AC([x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	[x_1	x_2	x_3	x_4]	$\leq_{lex}$ [x_2	x_4	x_1	x_3]	[x_1	x_2	x_3	x_4]	$\leq_{lex}$ [x_4	x_3	x_1	x_2]
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$	0	1			1				0	1			1			
$x_4 = 0$																

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma} [x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma \in \Sigma} AC([x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$	0	1			1				0	1			1			
$x_4 = 0$	0				1											

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma} [x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma \in \Sigma} AC([x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex} [x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$	0	1			1				0	1			1			
$x_4 = 0$	0				1				0	0			0	1		
$x_4 = 1$																

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma} [x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma \in \Sigma} AC([x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$ $[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$ $[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$	0	1			1				0	1			1			
$x_4 = 0$	0				1				0	0			0	1		
$x_4 = 1$	0				1											

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma} [x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$ *is strictly stronger than* $\bigwedge_{\sigma \in \Sigma} AC([x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \leq_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$	$[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\leq_{lex}$	$[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0				0	1			0	0				0	1		
$x_1 = 1$	1	1	1	1		1	1	1	1	1	0				1	1		
$x_2 = 0$	0	0				0	1			0	0				1			
$x_2 = 1$	0	1				1				0	1				1			
$x_4 = 0$	0					1				0	0				0	1		
$x_4 = 1$	0					1				0					1			

□

However, their conjunction is not AC. Indeed, suppose that $x_4 = 0$,

Breaking Several Variable Symmetries

Proposition 6 ([Puget05cp,Walsh06cp])

$AC(\bigwedge_{\sigma \in \Sigma} [x_1, \ldots, x_n] \le_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$ is strictly stronger than $\bigwedge_{\sigma \in \Sigma} AC([x_1, \ldots, x_n] \le_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}])$.

Proof.

Let $x_1, x_2, x_4 \in \{0, 1\}$ and $x_3 = 1$. Consider two symmetries (1243) and (1423), we have $AC([x_1, x_2, x_3, x_4] \le_{lex} [x_2, x_4, x_1, x_3])$ and $AC([x_1, x_2, x_3, x_4] \le_{lex} [x_4, x_3, x_1, x_2])$.

cases	$[x_1$	x_2	x_3	$x_4]$	$\le_{lex}$ $[x_2$	x_4	x_1	$x_3]$	$[x_1$	x_2	x_3	$x_4]$	$\le_{lex}$ $[x_4$	x_3	x_1	$x_2]$
$x_1 = 0$	0	0			0	1			0	0			0	1		
$x_1 = 1$	1	1	1	1	1	1	1	1	1	0			1	1		
$x_2 = 0$	0	0			0	1			0	0			1			
$x_2 = 1$	0	1			1				0	1			1			
$x_4 = 0$	0				1				0	0			0	1		
$x_4 = 1$	0				1				0				1			

□

However, their conjunction is not AC. Indeed, suppose that $x_4 = 0$, we have $x_1 = x_2 = 0$ and $x_3 = 0$, which is not possible.

Value Symmetry Breaking

A value symmetry is a bijection σ on values that preserves solutions.
$\{x_i = v_i | 1 \leq i \leq n\}$ is a solution iff $\{x_i = \sigma(v_i) | 1 \leq i \leq n\}$ is a solution

All value symmetries can be broken by posting for each value symmetry σ
$[x_1, ..., x_n] \leq_{lex} [\sigma(x_1), ..., \sigma(x_n)]$ [PS03cp]

Example 7 ($\sigma(i) = n + 1 - i$)

The symmetry breaking constraint implies $x_1 \leq n + 1 - x_1$
If n is even,

Value Symmetry Breaking

A value symmetry is a bijection σ on values that preserves solutions.
$\{x_i = v_i | 1 \leq i \leq n\}$ is a solution iff $\{x_i = \sigma(v_i) | 1 \leq i \leq n\}$ is a solution

All value symmetries can be broken by posting for each value symmetry σ
$[x_1, ..., x_n] \leq_{lex} [\sigma(x_1), ..., \sigma(x_n)]$ [PS03cp]

Example 7 ($\sigma(i) = n + 1 - i$)

The symmetry breaking constraint implies $x_1 \leq n + 1 - x_1$
If n is even, the constraint is thus equivalent to $x_1 \leq \frac{n}{2}$
If n is odd, it is equivalent to $x_1 \leq \frac{n+1}{2} \wedge x_1 = \frac{n+1}{2} \Rightarrow x_2 \leq \frac{n+1}{2} \wedge ...$

Breaking Variable and Value Symmetries

Theorem 8 ([Puget05cp,Walsh06cp])

The constraints $[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}]$ *for each variable symmetry* $\sigma \in \Sigma$
and $[x_1, ..., x_m] \leq_{lex} [\sigma'(x_1), ..., \sigma'(x_n)]$ *for each value symmetry* $\sigma' \in \Sigma'$
leave at least one assignment in each equivalence class of solutions.

Proof.

For any assignment ν,

Breaking Variable and Value Symmetries

Theorem 8 ([Puget05cp,Walsh06cp])

The constraints $[x_1, \ldots, x_n] \leq_{lex} [x_{\sigma(1)}, \ldots, x_{\sigma(n)}]$ *for each variable symmetry* $\sigma \in \Sigma$
and $[x_1, \ldots, x_m] \leq_{lex} [\sigma'(x_1), \ldots, \sigma'(x_n)]$ *for each value symmetry* $\sigma' \in \Sigma'$
leave at least one assignment in each equivalence class of solutions.

Proof.

For any assignment ν, one can pick the lex leader ν_1 of ν under Σ

Breaking Variable and Value Symmetries

Theorem 8 ([Puget05cp,Walsh06cp])

The constraints $[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}]$ *for each variable symmetry* $\sigma \in \Sigma$
and $[x_1, ..., x_m] \leq_{lex} [\sigma'(x_1), ..., \sigma'(x_n)]$ *for each value symmetry* $\sigma' \in \Sigma'$
leave at least one assignment in each equivalence class of solutions.

Proof.

For any assignment ν, one can pick the lex leader ν_1 of ν under Σ and then the lex leader ν_2 of ν_1 under Σ'

Breaking Variable and Value Symmetries

Theorem 8 ([Puget05cp,Walsh06cp])

The constraints $[x_1, ..., x_n] \leq_{lex} [x_{\sigma(1)}, ..., x_{\sigma(n)}]$ *for each variable symmetry* $\sigma \in \Sigma$
and $[x_1, ..., x_m] \leq_{lex} [\sigma'(x_1), ..., \sigma'(x_n)]$ *for each value symmetry* $\sigma' \in \Sigma'$
leave at least one assignment in each equivalence class of solutions.

Proof.

For any assignment ν, one can pick the lex leader ν_1 of ν under Σ and then the lex leader ν_2 of ν_1 under Σ'
If ν_2 does not satisfy the lex leader constraint under Σ, iterate.
As the lexicographic orders are well-founded, the process terminates, with an assignment that satisfies all lex leader constraints. □

Breaking Several Variable and Value Symmetries

The iterated lex leader may leave several symmetric assignments.

Example 9

Consider the composition of the reflection symmetries on both variables and boolean values.
The solutions $[0,1,1]$ and $[0,0,1]$ are symmetric but satisfy the lex constraints

$$[x_1, x_2, x_3] \leq [x_3, x_2, x_1]$$

$$[x_1, x_2, x_3] \leq [\neg x_1, \neg x_2, \neg x_3]$$

Indeed $[0,1,1] \leq [1,1,0]$ and $[0,1,1] \leq [1,0,0]$
$[0,0,1] \leq [1,0,0]$ and $[0,0,1] \leq [1,1,0]$
hence both symmetric solutions $[0,1,1]$ and $[0,0,1]$ are lex leaders.

Variable-Value Symmetries

Definition *A variable-value symmetry (or general symmetry) is a bijection σ on pairs $(variable, value)$ that preserves solutions.*

Definition *A valuation $[x_1, ..., x_n]$ is admissible for σ iff $|\{k \mid x_i = j,\ \sigma(i,j) = (k,l)\}| = n$.*

E.g. In the 4-queens, the assignment $[2, 3, 1, 4]$ is admissible for r90 but not $[2, 3, 3, 4]$.

If $[x_1, ..., x_n]$ is **admissible** for σ, let $\sigma[x_1, ..., x_n]$ be its image under σ, $\sigma[x_1, ..., x_n] = [y_1, ..., y_n]$ where $y_k = l$ whenever $x_i = j$ and $\sigma(i,j) = (k,l)$

Variable-Value Symmetry Breaking

Proposition 10

All variable-value symmetries can be broken by posting the constraints

$$\bigwedge_{\sigma \in \Sigma} admissible(\sigma, [x_1, ..., x_n]) \wedge [x_1, ..., x_n] \leq_{lex} \sigma[x_1, ..., x_n]$$

Example 11

In the 4-queens, let $x_1 = 2,\ x_2 \in \{1, 3, 4\},\ x_3$ and $x_4 \in \{1, 2, 3, 4\}$ r90$[x_1, ..., x_4]$ prunes $X_3 \neq 2$ and $X_4 \neq 2$ for admissibility, and $x_4 \neq 1$ for lex.

SBDS

In 99, [BW99cp] proposed a completely different symmetry breaking technique, **Symmetry Breaking During Search** (SBDS).

It overcomes the main drawback of static symmetry breaking:

SBDS

In 99, [BW99cp] proposed a completely different symmetry breaking technique, **Symmetry Breaking During Search** (SBDS).

It overcomes the main drawback of static symmetry breaking: the choice of the representative element in each class of solutions

SBDS

In 99, [BW99cp] proposed a completely different symmetry breaking technique, **Symmetry Breaking During Search** (SBDS).

It overcomes the main drawback of static symmetry breaking: the choice of the representative element in each class of solutions is forced

breaking all trials at improving performance by

SBDS

In 99, [BW99cp] proposed a completely different symmetry breaking technique, **Symmetry Breaking During Search** (SBDS).

It overcomes the main drawback of static symmetry breaking: the choice of the representative element in each class of solutions is forced

breaking all trials at improving performance by clever search heuristics

Symmetric Constraints

Consider a set Σ of symmetries, such that for any constraint c and all $\sigma \in \Sigma$ one can find a constraint $\sigma(c)$ corresponding to the symmetric of c
$\mathcal{X} \models \sigma(c)\rho \Leftrightarrow c\sigma(\rho)$

For example, if σ is the value symmetry that turns v into $N - v$ we have $\sigma(\mathtt{X\ =\ v})$ is

Symmetric Constraints

Consider a set Σ of symmetries, such that for any constraint *c* and all $\sigma \in \Sigma$ one can find a constraint $\sigma(c)$ corresponding to the symmetric of *c*
$\mathcal{X} \models \sigma(c)\rho \Leftrightarrow c\sigma(\rho)$

For example, if σ is the value symmetry that turns *v* into $N - v$ we have $\sigma($`X = v`$)$ is `X = (N - v)`

We can now define a technique for removing symmetries adding constraints when choice-points are explored, *à la* branch and bound.

Enumerating Solutions

The general method of enumeration of solutions is, at each choice-point, to add

- on one branch the constraint c assigning a value to a variable;
- on the other branch the negation of this constraint $\neg c$

SBDS adds supplementary constraints on the second branch:

supposing a partial assignment $\mathcal{A}$ at the choice-point,
for all $\sigma \in \Sigma$ such that $\sigma(\mathcal{A}) = \mathcal{A}$ one adds $\sigma(\neg c)$.

Example

Consider the 4-queens problem over $X_1, X_2, X_3, X_4 \in \{1, 2, 3, 4\}$

with a single (value-)symmetry: $v \mapsto 5 - v$

suppose that at the top of the search tree the leftmost branch corresponds to $X_1 = 1$

when backtracking at the top, the next branch to explore will correspond to the constraint:

$$X_1 \neq 1$$

Example

Consider the 4-queens problem over $X_1, X_2, X_3, X_4 \in \{1, 2, 3, 4\}$

with a single (value-)symmetry: $v \mapsto 5 - v$

suppose that at the top of the search tree the leftmost branch corresponds to $X_1 = 1$

when backtracking at the top, the next branch to explore will correspond to the constraint:

$$X_1 \neq 1 \wedge X_1 \neq 4$$

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 and ρ_2 are two solutions obtained by SBDS, then

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 and ρ_2 are two solutions obtained by SBDS, then

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Proof.

Suppose that $\sigma_0(\rho_1) = \rho_2$ for some σ_0

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 *and* ρ_2 *are two solutions obtained by SBDS, then*

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Proof.

Suppose that $\sigma_0(\rho_1) = \rho_2$ for some σ_0
let $\mathcal{A}$ be the partial assignment at the choice-point that differentiates the ρ_1 and ρ_2 branches, and *c* the constraint added on the ρ_1 branch there.

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 and ρ_2 are two solutions obtained by SBDS, then

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Proof.

Suppose that $\sigma_0(\rho_1) = \rho_2$ for some σ_0
let $\mathcal{A}$ be the partial assignment at the choice-point that differentiates the ρ_1 and ρ_2 branches, and *c* the constraint added on the ρ_1 branch there.
We have $\sigma_0(\mathcal{A}) = \mathcal{A}$

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 and ρ_2 are two solutions obtained by SBDS, then

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Proof.

Suppose that $\sigma_0(\rho_1) = \rho_2$ for some σ_0
let $\mathcal{A}$ be the partial assignment at the choice-point that differentiates the ρ_1 and ρ_2 branches, and *c* the constraint added on the ρ_1 branch there.
We have $\sigma_0(\mathcal{A}) = \mathcal{A}$
since both are solutions, we get that *c* is true in ρ_1

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 *and* ρ_2 *are two solutions obtained by SBDS, then*

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Proof.

Suppose that $\sigma_0(\rho_1) = \rho_2$ for some σ_0
let $\mathcal{A}$ be the partial assignment at the choice-point that differentiates the ρ_1 and ρ_2 branches, and *c* the constraint added on the ρ_1 branch there.
We have $\sigma_0(\mathcal{A}) = \mathcal{A}$
since both are solutions, we get that *c* is true in ρ_1
and that $\sigma_0(\neg c)$ is true in ρ_2

Unicity

Theorem 12 (Non-symmetric Solutions)

If ρ_1 and ρ_2 are two solutions obtained by SBDS, then

$$\forall \sigma \in \Sigma \qquad \sigma(\rho_1) \neq \rho_2$$

Proof.

Suppose that $\sigma_0(\rho_1) = \rho_2$ for some σ_0
let $\mathcal{A}$ be the partial assignment at the choice-point that differentiates the ρ_1 and ρ_2 branches, and *c* the constraint added on the ρ_1 branch there.
We have $\sigma_0(\mathcal{A}) = \mathcal{A}$
since both are solutions, we get that *c* is true in ρ_1
and that $\sigma_0(\neg c)$ is true in ρ_2 i.e., $\neg c$ is true in ρ_1
$\Rightarrow$ contradiction $\square$