Discrete Optimization

Constraint Programming: Part IV

Goals of the Lecture

- Illustrating the rich modeling language of constraint programming
- Key aspect of constraint programming
  - ability to state complex, idiosyncratic constraints

Global Constraints

- Critical features of constraint programming
  - capture combinatorial substructures arising in many applications

Global Constraints

- Critical features of constraint programming
  - capture combinatorial substructures arising in many applications
- Modeling
  - make modeling easier and more natural

Global Constraints

- Critical features of constraint programming
  - capture combinatorial substructures arising in many applications
- Modeling
  - make modeling easier and more natural
- Problem solving
  - convey the problem structure to the solver that does not have to rediscover it
  - give the ability to exploit dedicated algorithms

Global Constraints - AllDifferent

- alldifferent(x_1,...,x_n)
  - specifies that x_1,...,x_n take values that are different

```
range R = 1..8;
var{int} row[R] in R;
solve {
   forall(i in R,j in R: i < j) {
      row[i] ≠  row[j];
      row[i] + i ≠  row[j] + j;
      row[i] - i ≠  row[j] - j;
   }
}
```

Global Constraints - AllDifferent

- alldifferent($x_1,\ldots,x_n$)
  - specifies that $x_1,\ldots,x_n$ take values that are different

```
range R = 1..8;
var{int} row[R] in R;
solve {
   forall(i in R,j in R: i < j) {
      row[i] ≠  row[j];
      row[i] + i ≠  row[j] + j;
      row[i] - i ≠  row[j] - j;
   }
}
```

⇒

```
range R = 1..8;
var{int} row[R] in R;
solve {
   alldifferent(row);
   alldifferent(all(i in R) row[i]+i);
   alldifferent(all(i in R) row[i]-i);
}
```

Global Constraints - AllDifferent

- alldifferent($x_1,\ldots,x_n$)
  - specifies that $x_1,\ldots,x_n$ take values that are different

```
range R = 1..8;
var{int} row[R] in R;
solve {
   forall(i in R,j in R: i < j) {
      row[i] ≠  row[j];
      row[i] + i ≠  row[j] + j;
      row[i] - i ≠  row[j] - j;
   }
}
```

```
range R = 1..8;
var{int} row[R] in R;
solve {
   alldifferent(row);
   alldifferent(all(i in R) row[i]+i);
   alldifferent(all(i in R) row[i]-i);
}
```

array comprehension: collect all the elements in an array

Why global constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n

- Feasibility testing
  - Can I find values in the variable domains such that the constraint holds?

Why global constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n

$$\mathrm{c}(x_1, \ldots, x_n)$$

$$D_1 = D(x_1), \ldots, D_n = D(x_n)$$

- Feasibility testing
  - Can I find values in the variable domains such that the constraint holds?

Why global constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n

$$\mathrm{c}(x_1, \ldots, x_n)$$

$$D_1 = D(x_1), \ldots, D_n = D(x_n)$$

- Feasibility testing
  - Can I find values in the variable domains such that the constraint holds?

Why global constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n

$$\mathrm{c}(x_1, \ldots, x_n)$$

$$D_1 = D(x_1), \ldots, D_n = D(x_n)$$

- Feasibility testing
  - Can I find values in the variable domains such that the constraint holds?

$$\exists\, v_1 \in D_1, \ldots, v_n \in D_n :$$
$$\mathrm{c}(x_1 = v_1, \ldots, x_n = v_n) = \mathrm{true}$$

Why Global Constraints?

- Given
  - a constraint alldifferent($x_1,\ldots,x_3$)
  - x_1 in [1..2], ..., x_3 in [1..2]
- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], ..., x_3 in [1..2]

$$\mathrm{alldifferent}(x_1, x_2, x_3)$$
$$D_1 = \{1,2\}, D_2 = \{1,2\}, D_3 = \{1,2\}$$

- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], ..., x_3 in [1..2]

$$\text{alldifferent}(x_1, x_2, x_3)$$
$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2\}$$

- Is this feasible?

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], ..., x_3 in [1..2]

$$\text{alldifferent}(x_1, x_2, x_3)$$
$$D_1 = \{1,2\}, D_2 = \{1,2\}, D_3 = \{1,2\}$$

- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)

Why Global Constraints?

$$\text{alldifferent}(x_1, x_2, x_3)$$

$$D_1 = \{1,2\}, D_2 = \{1,2\}, D_3 = \{1,2\}$$

- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)

Why Global Constraints?

$$\text{alldifferent}(x_1, x_2, x_3)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2\}$$

- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)
- However $x_1 \neq x_2, x_2 \neq x_3, x_3 \neq x_1$

Why Global Constraints?

$$\text{alldifferent}(x_1, x_2, x_3)$$

$$D_1 = \{1,2\}, D_2 = \{1,2\}, D_3 = \{1,2\}$$

- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)
- However $x_1 \neq x_2, x_2 \neq x_3, x_3 \neq x_1$

$$\exists\, v_1 \in D_1, \ldots, v_n \in D_n :$$
$$\text{c}(x_1 = v_1, \ldots, x_n = v_n) = \text{true}$$

Why Global Constraints?

$$\text{alldifferent}(x_1, x_2, x_3)$$
$$D_1 = \{1,2\}, D_2 = \{1,2\}, D_3 = \{1,2\}$$

- Is this feasible?
  - No, only two values for 3 variables (pigeon hole principle)
- However $x_1 \neq x_2, x_2 \neq x_3, x_3 \neq x_1$

$$\exists\, v_1 \in D_1, \ldots, v_n \in D_n :$$
$$\text{c}(x_1 = v_1, \ldots, x_n = v_n) = \text{true}$$

$$\neq (x_1 = 1, x_2 = 2) = \text{true}$$
$$\neq (x_2 = 1, x_3 = 2) = \text{true}$$
$$\neq (x_3 = 1, x_1 = 2) = \text{true}$$

Why Global Constraints?

$$\text{alldifferent}(x_1, x_2, x_3)$$

$$D_1 = \{1,2\}, D_2 = \{1,2\}, D_3 = \{1,2\}$$

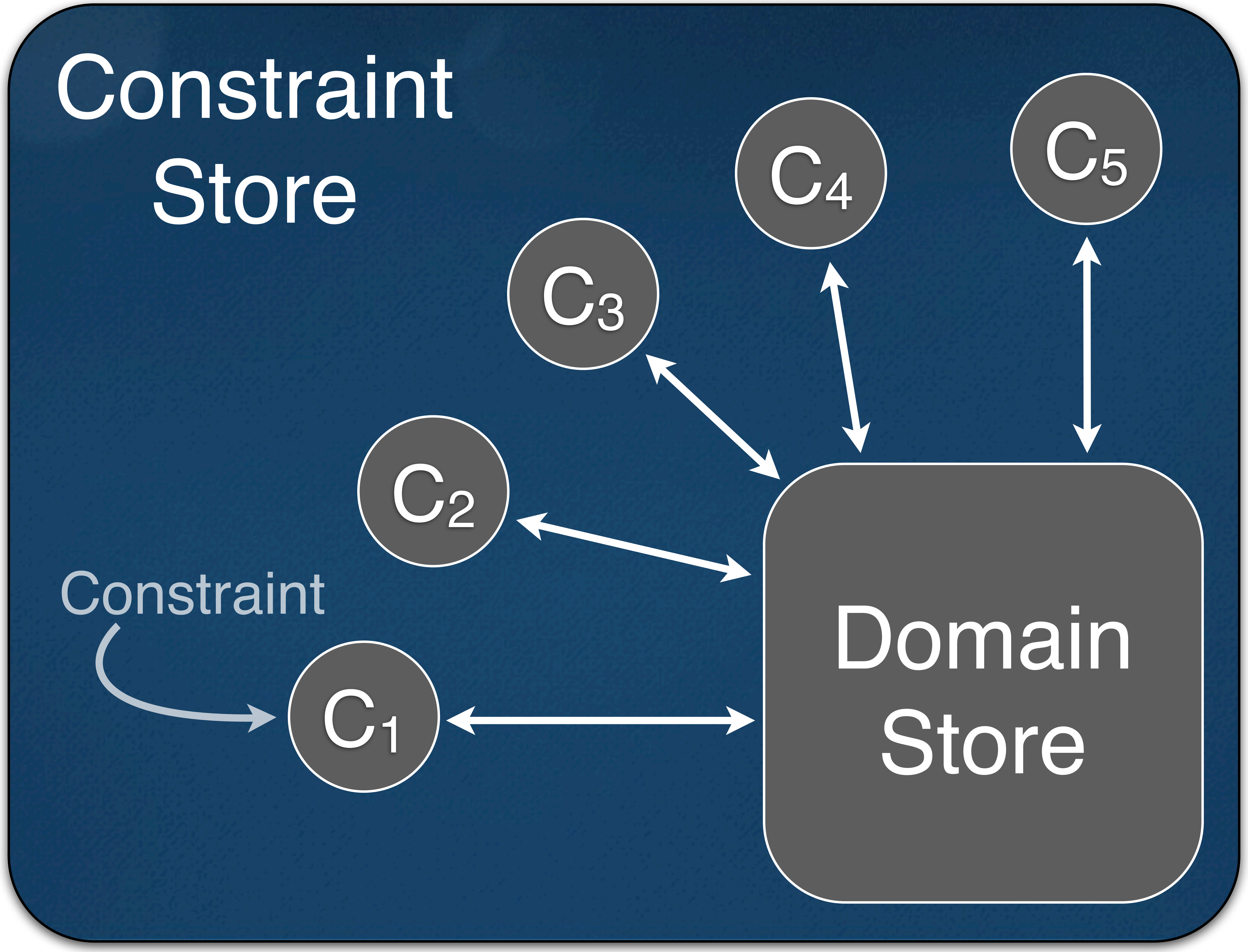
Global constraints make it possible to discover infeasibilities earlier

- However $x_1 \neq x_2, x_2 \neq x_3, x_3 \neq x_1$

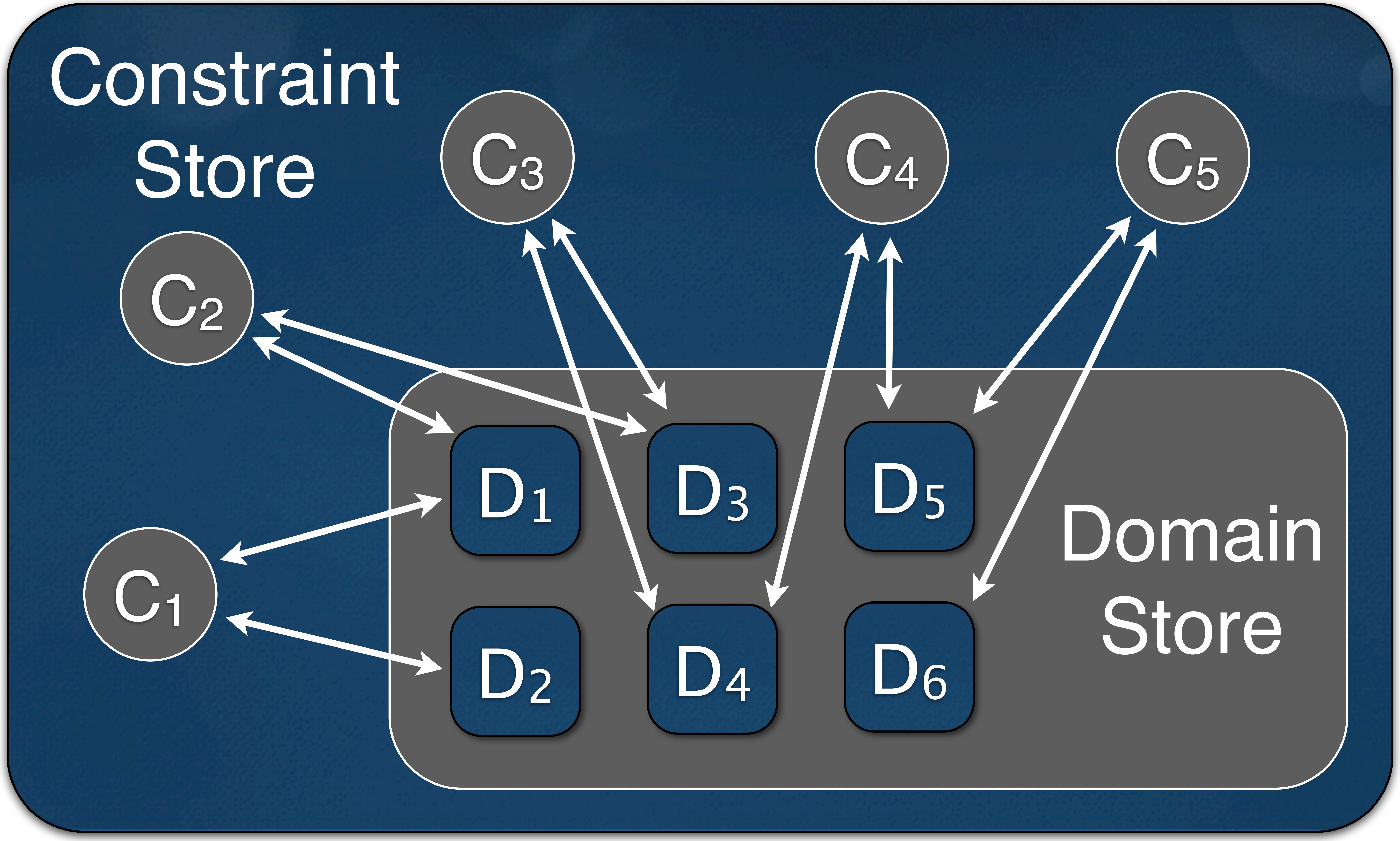
$$\exists\, v_1 \in D_1, \ldots, v_n \in D_n :$$
$$\text{c}(x_1 = v_1, \ldots, x_n = v_n) = \text{true}$$

$$\neq (x_1 = 1, x_2 = 2) = \text{true}$$
$$\neq (x_2 = 1, x_3 = 2) = \text{true}$$
$$\neq (x_3 = 1, x_1 = 2) = \text{true}$$

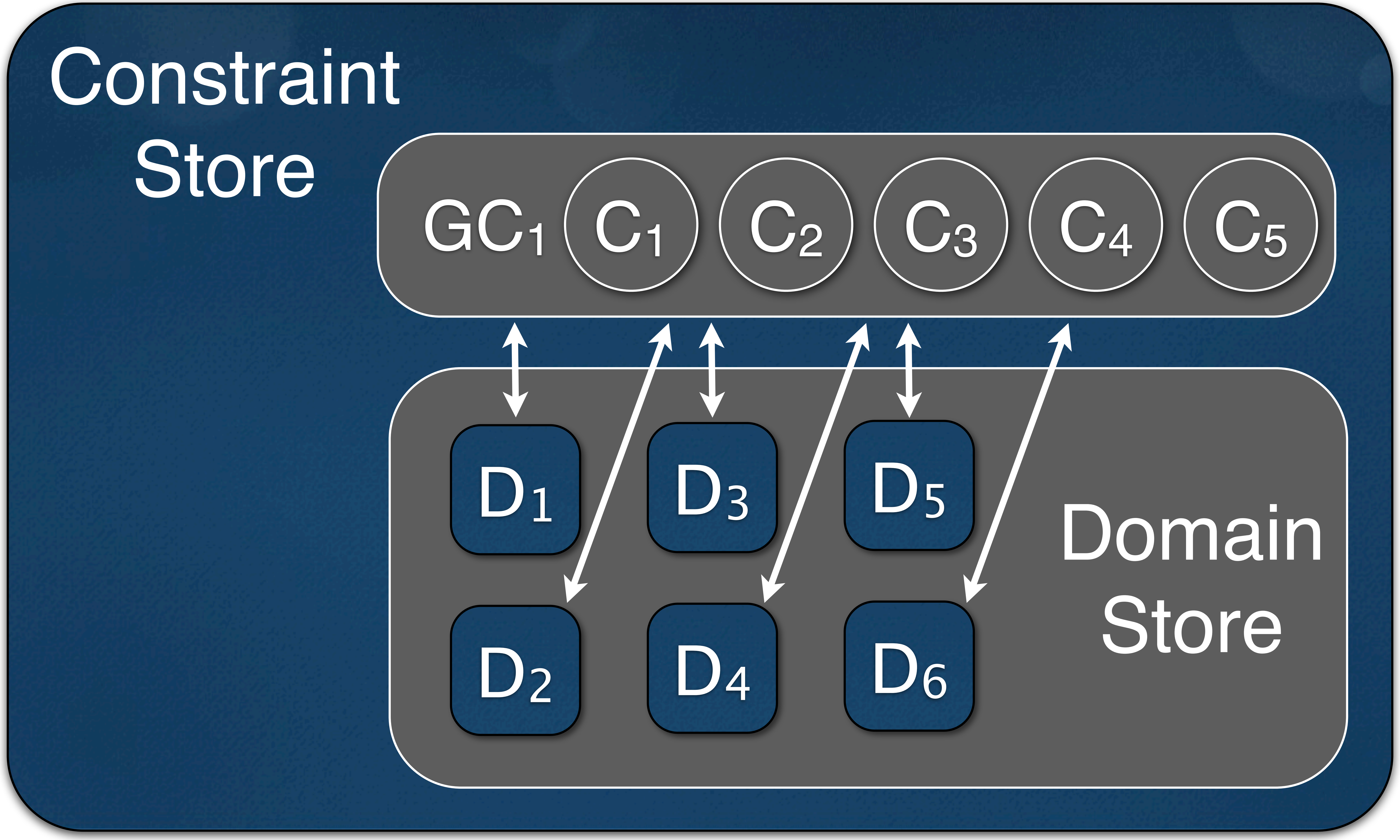
Computational Paradigm



Computational Paradigm



Computational Paradigm



Why Global Constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n

Why Global Constraints?

- Given
  - a constraint $c(x_1,\ldots,x_n)$
  - x_1 in D_1, ..., x_n in D_n
- Pruning

Why Global Constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n
- Pruning
  - Given v_i in D_i, does there exist a solution such that $x_i=v_i$?

Why Global Constraints?

- Given
  - a constraint $c(x_1,...,x_n)$
  - x_1 in D_1, ..., x_n in D_n
- Pruning
  - Given v_i in D_i, does there exist a solution such that $x_i=v_i$?
  - Can I find values in the variable domains such that the constraint holds?

Why Global Constraints?

- Given
  - a constraint $c(x_1,\ldots,x_n)$
  - x_1 in D_1, ..., x_n in D_n
- Pruning
  - Given v_i in D_i, does there exist a solution such that $x_i=v_i$?
  - Can I find values in the variable domains such that the constraint holds?

$$\text{given: } v_i \in D_i,\ x_i = v_i$$

$$\begin{aligned} \exists\, & v_1 \in D_1, \ldots, v_{i-1} \in D_{i-1}, \\ & v_{i+1} \in D_{i+1}, \ldots, v_n \in D_n : \\ & \mathrm{c}(x_1 = v_1, \ldots, x_n = v_n) = \text{true} \end{aligned}$$

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], x_2 in [1..2], x_3 in [1..3]

$$\text{alldifferent}(x_1, \ldots, x_n)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2, 3\}$$

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], x_2 in [1..2], x_3 in [1..3]

$$\text{alldifferent}(x_1, \ldots, x_n)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2, 3\}$$

- Pruning?

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], x_2 in [1..2], x_3 in [1..3]

$$\text{alldifferent}(x_1, \ldots, x_n)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2, 3\}$$

- Pruning?

$$x_3 \neq 1 \quad x_3 \neq 2$$

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], x_2 in [1..2], x_3 in [1..3]

$$\text{alldifferent}(x_1, \ldots, x_n)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2, 3\}$$

- Pruning?

$$x_3 \neq 1 \quad x_3 \neq 2$$

Why Global Constraints?

- Given
  - a constraint alldifferent(x_1,...,x_3)
  - x_1 in [1..2], x_2 in [1..2], x_3 in [1..3]

$$\text{alldifferent}(x_1, \ldots, x_n)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2, 3\}$$

- Pruning?

$$x_3 \neq 1 \quad x_3 \neq 2$$

- However $x_1 \neq x_2, x_2 \neq x_3, x_3 \neq x_1$
  - Do not produce any pruning

Why Global Constraints?

Global constraints make it possible to prune the search space more.

$$\text{alldifferent}(x_1, \ldots, x_n)$$

$$D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2, 3\}$$

- Pruning?

  $x_3 \neq 1 \quad x_3 \neq 2$

- However $x_1 \neq x_2, x_2 \neq x_3, x_3 \neq x_1$
  - Do not produce any pruning

Why Global Constraints?

- The million-dollar question
  - Can we detect feasibility and prune global constraints efficiently?

Why Global Constraints?

- The million-dollar question
  - Can we detect feasibility and prune global constraints efficiently?
- It depends on the constraint obviously
  - sometimes we can
  - sometimes we need to relax our standards
    - the pruning may be suboptimal
    - the pruning may take exponential time
    - ....

Sudoku

			1		2	9		
				9		3		1
					8			6
				3				
	6	2						
	7	9		1	6			
		8		6				7
		4				1	9	
					4		2	

Sudoku

```
range R = 1..9;
var{int} s[R,R] in R;
solve {
 //constraints on fixed positions
 forall(i in R)
   alldifferent(all(j in R) s[i,j]);
 forall(j in R)
   alldifferent(all(i in R) s[i,j]);
 forall(i in 0..2,j in 0..2)
   alldifferent(all(r in i*3+1..i*3+3,
                     c in j*3+1..j*3+3) s[r,c]);
}
```

Sudoku

			1		2	9		
				9		3		1
					8			6
				3				
	6	2						
	7	9		1	6			
		8		6				7
		4				1	9	
					4		2	

Sudoku

			1		2	9		
				9		3		1
					8			6
				3				
	6	2						
	7	9		1	6			
		8		6				7
		4				1	9	
					4		2	

Sudoku

			1		2	9		
				9		3		1
					8			6
				3				
	6	2						
	7	9		1	6			
		8		6				7
		4		2		1	9	
					4		2	

Sudoku

8	3	6	1		2	9		4
2	4		6	9		3	8	1
	9		3	4	8	2		6
	8			3			6	
	6	2					1	
	7	9		1	6		4	
9	2	8	5	6	1	4	3	7
6	5	4	7	2	3	1	9	8
7	1	3	9	8	4		2	5

Sudoku

8	3	6	1	5	2	9		4
2	4		6	9		3	8	1
	9		3	4	8	2		6
	8			3			6	
	6	2					1	
	7	9		1	6		4	
9	2	8	5	6	1	4	3	7
6	5	4	7	2	3	1	9	8
7	1	3	9	8	4		2	5

Sudoku

8	3	6	1	5	2	9	7	4
2	4	5	6	9	7	3	8	1
1	9	7	3	4	8	2	5	6
	8	1		3		7	6	
	6	2		7			1	
	7	9		1	6		4	
9	2	8	5	6	1	4	3	7
6	5	4	7	2	3	1	9	8
7	1	3	9	8	4	6	2	5

Sudoku

8	3	6	1	5	2	9	7	4
2	4	5	6	9	7	3	8	1
1	9	7	3	4	8	2	5	6
4	8	1		3		7	6	
	6	2		7			1	
	7	9		1	6		4	
9	2	8	5	6	1	4	3	7
6	5	4	7	2	3	1	9	8
7	1	3	9	8	4	6	2	5

Sudoku

8	3	6	1	5	2	9	7	4
2	4	5	6	9	7	3	8	1
1	9	7	3	4	8	2	5	6
4	8	1	2	3	5	7	6	9
5	6	2	4	7	9	8	1	3
3	7	9	8	1	6	5	4	2
9	2	8	5	6	1	4	3	7
6	5	4	7	2	3	1	9	8
7	1	3	9	8	4	6	2	5

Table Constraints

- The simplest global constraint

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

Total Possibilities?

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Table Constraint	X	Y	Z
Combination 1	1	1	5
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

$\boxed{Z \neq 5}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Table Constraint	X	Y	Z
Combination 1	1	1	5
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

$$Z \neq 5$$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Table Constraint	X	Y	Z
Combination 1	1	1	5
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

$\boxed{Z \neq 5}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Pruning?

Table Constraint	X	Y	Z
Combination 1	1	1	5
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

$\boxed{Z \neq 5}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Pruning?

Table Constraint	X	Y	Z
~~Combination 1~~	~~1~~	~~1~~	~~5~~
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

$\boxed{Z \neq 5}$

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Pruning?

Table Constraint	X	Y	Z
Combination 1	1	1	5
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Table Constraints

- The simplest global constraint

X $\in \{1, 2\}$

Y $\in \{1, 2\}$

Z $\in \{3, 4, 5\}$

$\rightarrow \boxed{Z \neq 5} \rightarrow \boxed{Y \neq 1}$

Pruning?

Total Possibilities?

$$|\{1,2\}| \times |\{1,2\}| \times |\{3,4,5\}|$$

$$2 \times 2 \times 3 = 12$$

Table Constraint	X	Y	Z
Combination 1	1	1	5
Combination 2	1	2	4
Combination 3	2	2	3
Combination 4	1	2	3

Finding Optimal Solutions

```
enum Countries = { Belgium, Denmark, France,
                   Germany, Netherlands, Luxembourg };
var{int} color[Countries] in 1..4;
minimize
  max(c in Countries) color[c]
subject to {
  color[Belgium] ≠ color[France];
  color[Belgium] ≠ color[Germany];
  color[Belgium] ≠ color[Netherlands];
  color[Belgium] ≠ color[Luxembourg];
  color[Denmark] ≠ color[Germany];
  color[France] ≠ color[Germany];
  color[France] ≠ color[Luxembourg];
  color[Germany] ≠ color[Netherlands];
  color[Germany] ≠ color[Luxembourg];
}
```

Optimization in Constraint Programming?

- Focus of constraint programming
  - feasibility

Optimization in Constraint Programming?

- Focus of constraint programming
  - feasibility
- How to optimize?
  - Solve a sequence of satisfaction problems
  - Find a solution
  - Impose a constraint that the next solution must be better

Optimization in Constraint Programming?

- Focus of constraint programming
  - feasibility
- How to optimize?
  - Solve a sequence of satisfaction problems
  - Find a solution
  - Impose a constraint that the next solution must be better
- Guaranteed to find an optimal solution
  - at least theoretically
  - strong when the new constraint reduces the search space
    - scheduling problems are good examples