

# Introduction to Constraint Programming

Helmut Simonis

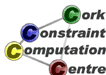
Cork Constraint Computation Centre  
Computer Science Department  
University College Cork  
Ireland

CP Meets CAV, Turunc, June 25th, 2012



# Licence

This work is licensed under the Creative Commons Attribution-Noncommercial-Share Alike 3.0 Unported License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/3.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.



# Acknowledgments

The author is partially supported by Science Foundation Ireland (Grant Number 05/IN/1886). This material was developed as part of the ECLiPSe ELearning course:

<http://4c.ucc.ie/~hsimonis/ELearning/index.htm>

Support from Cisco Systems and the Silicon Valley Community Foundation is gratefully acknowledged.



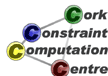
# Objectives

- Overview of Core Constraint Programming
- Three Main Concepts
  - Constraint Propagation
  - Global Constraints
  - Customizing Search
- Based on Examples, not Formal Description



# Outline

- Why Constraint Programming?
- Constraint Propagation
- Global Constraints
- Customizing Search
- What is missing?



# Examples in ECLiPSe

- Open sourced constraint programming language
- Development goes back to 1985
- ECRC, ICL, IC-Parc, PTL, Cisco
- <http://www.eclipse-clp.org/>
- Specialities
  - Develop new solvers for specific domains
  - Integration with MIP



# ECLIPSe ELearning Course

- Self-study course in constraint programming
- Supported by Cisco Systems and Silicon Valley Community Foundation
- Multi-media format, video lectures, slides, handout etc
- <http://4c.ucc.ie/~hsimonis/ELearning/index.htm>



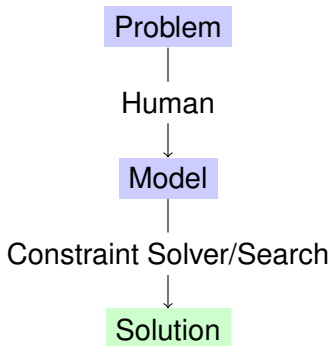
# Constraint Programming - in a nutshell

- Declarative description of problems with
  - *Variables* which range over (finite) sets of values
  - *Constraints* over subsets of variables which restrict possible value combinations
  - A *solution* is a value assignment which satisfies all constraints
- Constraint propagation/reasoning
  - Removing inconsistent values for variables
  - Detect failure if constraint can not be satisfied
  - Interaction of constraints via shared variables
  - Incomplete
- Search
  - User controlled assignment of values to variables
  - Each step triggers constraint propagation
- Different domains require/allow different methods

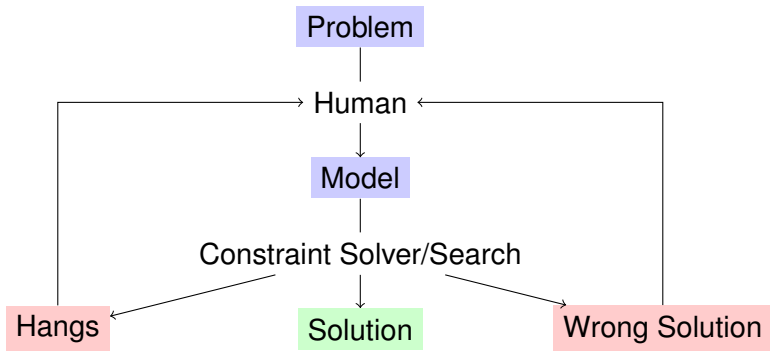




# Basic Process



# More Realistic



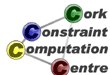
# Dual Role of Model

- Allows Human to Express Problem
  - Close to Problem Domain
  - Constraints as Abstractions
- Allows Solver to Execute
  - Variables as Communication Mechanism
  - Constraints as Algorithms



# Part I

## Basic Constraint Propagation



# Example 1: SEND+MORE=MONEY

- Example of Finite Domain Constraint Problem
- Models and Programs
- Constraint Propagation and Search
- Some Basic Constraints: linear arithmetic, alldifferent, disequality
- A Built-in search
- Visualizers for variables, constraints and search



# Outline

- 1 Problem
- 2 Program
- 3 Constraint Setup
- 4 Search
- 5 Points to Remember

# Problem Definition

## A Crypt-Arithmetic Puzzle

We begin with the definition of the SEND+MORE=MONEY puzzle. It is often shown in the form of a hand-written addition:

$$\begin{array}{rcccc} & S & E & N & D \\ + & M & O & R & E \\ \hline M & O & N & E & Y \end{array}$$

# Rules

- Each character stands for a digit from 0 to 9.
- Numbers are built from digits in the usual, positional notation.
- Repeated occurrence of the same character denote the same digit.
- Different characters denote different digits.
- Numbers do not start with a zero.
- The equation must hold.

$$\begin{array}{rcccc}
 & S & E & N & D \\
 + & M & O & R & E \\
 \hline
 M & O & N & E & Y
 \end{array}$$





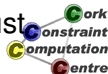
# Outline

- 1 Problem
- 2 Program
- 3 Constraint Setup
- 4 Search
- 5 Points to Remember



# Model

- Each character is a variable, which ranges over the values 0 to 9.
- An *alldifferent* constraint between all variables, which states that two different variables must have different values. This is a very common constraint, which we will encounter in many other problems later on.
- Two *disequality constraints* (variable  $X$  must be different from value  $V$ ) stating that the variables at the beginning of a number can not take the value 0.
- An arithmetic *equality constraint* linking all variables with the proper coefficients and stating that the equation must hold.



# Program Sendmory

```
:- module(sendmory). ⇒ Define Module
```

```
:- export(sendmory/1).
```

```
:- lib(ic).
```

```
sendmory(L) :-
```

```
    L = [S,E,N,D,M,O,R,Y],
```

```
    L :: 0..9,
```

```
    alldifferent(L),
```

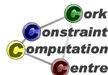
```
    S #\= 0, M #\= 0,
```

```
    1000*S + 100*E + 10*N + D +
```

```
    1000*M + 100*O + 10*R + E #=
```

```
    10000*M + 1000*O + 100*N + 10*E + Y,
```

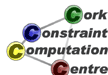
```
    labeling(L).
```



# Program Sendmory

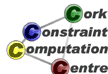
```
:- module (sendmory) .
:- export (sendmory/1) . ⇨ Make predicate visible
:- lib (ic) .

sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent (L),
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
```



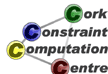
# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib(ic). ⇒ Use ic library
sendmory(L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent(L),
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling(L) .
```



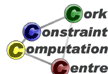
# Program Sendmory

```
:- module (sendmory) .  
:- export (sendmory/1) .  
:- lib (ic) .  
sendmory(L) :- $\Rightarrow$  Predicate definition  
    L = [S,E,N,D,M,O,R,Y],  
    L :: 0..9,  
    alldifferent (L) ,  
    S #\= 0, M #\= 0,  
    1000*S + 100*E + 10*N + D +  
    1000*M + 100*O + 10*R + E #=  
    10000*M + 1000*O + 100*N + 10*E + Y,  
    labeling (L) .
```



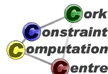
# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory(L) :-
    L = [S,E,N,D,M,O,R,Y],  $\Rightarrow$  Define list
    L :: 0..9,
    alldifferent (L) ,
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
```



# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9, ⇨ Define integer domain 0..9
    alldifferent (L) ,
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
```





# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .

sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent (L),  $\Leftrightarrow$  Digits must be different
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
```



# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
```

```
sendmory (L) :-
```

```
    L = [S,E,N,D,M,O,R,Y],
```

```
    L :: 0..9,
```

```
    alldifferent (L),
```

*S #\= 0, M #\= 0, ⇨ Numbers don't start with 0*

```
    1000*S + 100*E + 10*N + D +
```

```
    1000*M + 100*O + 10*R + E #=
```

```
    10000*M + 1000*O + 100*N + 10*E + Y,
```

```
    labeling (L) .
```

$$\begin{array}{r} \phantom{+} S \phantom{E} \phantom{N} \phantom{D} \\ + M \phantom{O} \phantom{R} \phantom{E} \\ \hline M \phantom{O} \phantom{N} \phantom{E} \phantom{Y} \end{array}$$


# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
```

```
sendmory (L) :-
```

```
    L = [S,E,N,D,M,O,R,Y],
```

```
    L :: 0..9,
```

```
    alldifferent (L),
```

```
    S #\= 0, M #\= 0,
```

```
    1000*S + 100*E + 10*N + D +
```

```
    1000*M + 100*O + 10*R + E #=
```

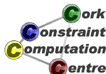
```
    10000*M + 1000*O + 100*N + 10*E + Y,
```

```
    labeling (L) .
```

$$\begin{array}{r} \phantom{+} \phantom{M} \phantom{O} \phantom{N} \phantom{E} \\ + \phantom{M} \phantom{O} \phantom{N} \phantom{E} \\ \hline M \phantom{O} \phantom{N} \phantom{E} \phantom{Y} \end{array}$$

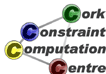

# Program Sendmory

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent (L),
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) . ↪ built-in search routine
```



# Program Sendmory

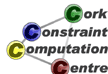
```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent (L),
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
```



# General Program Structure

```

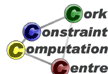
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y], ⇨ Variables
    L :: 0..9,
    alldifferent (L) ,
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
  
```



# General Program Structure

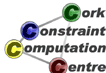
```

:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent (L), ⇨ Constraints
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) .
  
```



# General Program Structure

```
:- module (sendmory) .
:- export (sendmory/1) .
:- lib (ic) .
sendmory (L) :-
    L = [S,E,N,D,M,O,R,Y],
    L :: 0..9,
    alldifferent (L),
    S #\= 0, M #\= 0,
    1000*S + 100*E + 10*N + D +
    1000*M + 100*O + 10*R + E #=
    10000*M + 1000*O + 100*N + 10*E + Y,
    labeling (L) . ⇨ Search
```





# Choice of Model

- This is *one* model, not *the* model of the problem
- Many possible alternatives
- Choice often depends on your constraint system
  - Constraints available
  - Reasoning attached to constraints
- Not always clear which is the *best* model
- Often: Not clear what is the *problem*



## Running the program

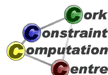
- To run the program, we have to enter the query
  - `sendmory:sendmory(L).`
- Result
  - `L = [9, 5, 6, 7, 1, 0, 8, 2]`
  - `yes (0.00s cpu, solution 1, maybe more)`

# Question

- But how did the program come up with this solution?

# Outline

- 1 Problem
- 2 Program
- 3 Constraint Setup**
  - Domain Definition
  - Alldifferent Constraint
  - Disequality Constraints
  - Equality Constraint
- 4 Search
- 5 Points to Remember



# Domain Definition

$L = [S, E, N, D, M, O, R, Y],$

$L :: 0..9,$

$[S, E, N, D, M, O, R, Y] \in \{0..9\}$

# Domain Visualization

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Domain Visualization

Rows =  
Variables

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Domain Visualization

Columns = Values

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										



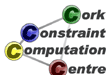
# Domain Visualization

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M			Cells=		State					
O										
R										
Y										

# Alldifferent Constraint

`alldifferent (L) ,`

- Built-in of `ic` library
- No initial propagation possible
- *Suspends*, waits until variables are changed
- When variable is fixed, remove value from domain of other variables
- *Forward checking*



# Alldifferent Visualization

Uses the same representation as the domain visualizer

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Disequality Constraints

$$S \neq 0, M \neq 0,$$

Remove value from domain

$$S \in \{1..9\}, M \in \{1..9\}$$

Constraints solved, can be removed



# Domains after Disequality

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Equality Constraint

- Normalization of linear terms
  - Single occurrence of variable
  - Positive coefficients
- Propagation



# Normalization

$$\begin{array}{rcccc}
 & 1000 * S_+ & 100 * E_+ & 10 * N_+ & D \\
 + & 1000 * M_+ & 100 * O_+ & 10 * R_+ & E \\
 \hline
 10000 * M_+ & 1000 * O_+ & 100 * N_+ & 10 * E_+ & Y
 \end{array}$$

# Normalization

$$\begin{array}{rcccc}
 & 1000*S+ & 100*E+ & 10*N+ & D \\
 & +1000*M+ & 100*O+ & 10*R+ & E \\
 \hline
 10000*M+ & 1000*O+ & 100*N+ & 10*E+ & Y
 \end{array}$$



# Normalization

$$\begin{array}{r}
 1000 * S + 100 * E + 10 * N + D \\
 + 100 * O + 10 * R + E \\
 \hline
 9000 * M + 1000 * O + 100 * N + 10 * E + Y
 \end{array}$$

# Normalization

$$\begin{array}{r}
 1000 * S + 100 * E + 10 * N + D \\
 + 100 * O + 10 * R + E \\
 \hline
 9000 * M + 1000 * O + 100 * N + 10 * E + Y
 \end{array}$$

# Normalization

$$\begin{array}{rcccc}
 1000 * S + & 100 * E + & 10 * N + & D & \\
 & & + 10 * R + & E & \\
 \hline
 9000 * M + & \mathbf{900 * O} + & 100 * N + & 10 * E + & Y
 \end{array}$$



# Normalization

$$\begin{array}{r}
 1000 * S + \quad 100 * E + \quad \quad \quad D \\
 \quad \quad \quad \quad \quad + \quad 10 * R + \quad E \\
 \hline
 9000 * M + \quad 900 * O + \quad \mathbf{90 * N} + \quad 10 * E + \quad Y
 \end{array}$$



# Normalization

$$\begin{array}{r}
 1000 * S + \quad \mathbf{100 * E} + \quad \quad \quad D \\
 \quad \quad \quad \quad \quad + \quad 10 * R + \quad \quad \quad \mathbf{E} \\
 \hline
 9000 * M + \quad 900 * O + \quad 90 * N + \quad \mathbf{10 * E} + \quad Y
 \end{array}$$



# Normalization

$$\begin{array}{r}
 1000 * S + 91 * E + \phantom{10 * R} \\
 \phantom{1000 * S +} + 10 * R \\
 \hline
 9000 * M + 900 * O + 90 * N + \phantom{10 * R} \\
 \phantom{9000 * M + 900 * O + 90 * N +} Y
 \end{array}$$



## Simplified Equation

$$1000 * S + 91 * E + 10 * R + D = 9000 * M + 900 * O + 90 * N + Y$$



# Propagation

$$1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9} =$$
$$9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}$$

# Propagation

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{1000..9918} = \underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..89919}$$

# Propagation

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{9000..9918} =$$

$$\underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..9918}$$

# Propagation

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{9000..9918} = \underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..9918}$$

Deduction:

$$M = 1, S = 9, O \in \{0..1\}$$

# Propagation

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{9000..9918} = \underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..9918}$$

Deduction:

$$M = 1, S = 9, O \in \{0..1\}$$

Why? [Skip](#)



## Consider lower bound for $S$

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{9000..9918} = \underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..9918}$$

- Lower bound of equation is 9000
- Rest of lhs (left hand side) ( $91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}$ ) is at most 918
- $S$  must be greater or equal to  $\frac{9000-918}{1000} = 8.082$ 
  - otherwise lower bound of equation not reached by lhs
- $S$  is integer, therefore  $S \geq \lceil \frac{9000-918}{1000} \rceil = 9$
- $S$  has upper bound of 9, so  $S = 9$

## Consider upper bound of $M$

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{9000..9918} = \underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..9918}$$

- Upper bound of equation is 9918
- Rest of rhs (right hand side)  $900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}$  is at least 0
- $M$  must be smaller or equal to  $\frac{9918-0}{9000} = 1.102$
- $M$  must be integer, therefore  $M \leq \lfloor \frac{9918-0}{9000} \rfloor = 1$
- $M$  has lower bound of 1, so  $M = 1$

## Consider upper bound of $O$

$$\underbrace{1000 * S^{1..9} + 91 * E^{0..9} + 10 * R^{0..9} + D^{0..9}}_{9000..9918} = \underbrace{9000 * M^{1..9} + 900 * O^{0..9} + 90 * N^{0..9} + Y^{0..9}}_{9000..9918}$$

- Upper bound of equation is 9918
- Rest of rhs (right hand side)  $9000 * 1 + 90 * N^{0..9} + Y^{0..9}$  is at least 9000
- $O$  must be smaller or equal to  $\frac{9918-9000}{900} = 1.02$
- $O$  must be integer, therefore  $O \leq \lfloor \frac{9918-9000}{900} \rfloor = 1$
- $O$  has lower bound of 0, so  $O \in \{0..1\}$



# Propagation of equality: Result

	0	1	2	3	4	5	6	7	8	9
S		-	-	-	-	-	-	-	-	☀
E										
N										
D										
M		☀	-	-	-	-	-	-	-	-
O			✘	✘	✘	✘	✘	✘	✘	✘
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S		-	-	-	-	-	-	-	-	✱
E										
N										
D										
M		✱	-	-	-	-	-	-	-	-
O			✕	✕	✕	✕	✕	✕	✕	✕
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										✱
E										
N										
D										
M		✱								
O										
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M		☀								
O										
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O	☀									
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O	*									
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

$$O = 0, [E, R, D, N, Y] \in \{2..8\}$$

## Waking the equality constraint

- Triggered by assignment of variables
- *or* update of lower or upper bound



## Removal of constants

$$1000 * 9 + 91 * E^{2..8} + 10 * R^{2..8} + D^{2..8} = \\ 9000 * 1 + 900 * 0 + 90 * N^{2..8} + Y^{2..8}$$

# Removal of constants

$$1000 * 9 + 91 * E^{2..8} + 10 * R^{2..8} + D^{2..8} = \\ 9000 * 1 + 900 * 0 + 90 * N^{2..8} + Y^{2..8}$$

## Removal of constants

$$91 * E^{2..8} + 10 * R^{2..8} + D^{2..8} = 90 * N^{2..8} + Y^{2..8}$$

# Propagation of equality (Iteration 1)

$$\underbrace{91 * E^{2..8} + 10 * R^{2..8} + D^{2..8}}_{204..816} = \underbrace{90 * N^{2..8} + Y^{2..8}}_{182..728}$$

# Propagation of equality (Iteration 1)

$$\underbrace{91 * E^{2..8} + 10 * R^{2..8} + D^{2..8}}_{204..728} = 90 * N^{2..8} + Y^{2..8}$$

# Propagation of equality (Iteration 1)

$$\underbrace{91 * E^{2..8} + 10 * R^{2..8} + D^{2..8} = 90 * N^{2..8} + Y^{2..8}}_{204..728}$$

$$N \geq 3 = \lceil \frac{204 - 8}{90} \rceil, E \leq 7 = \lfloor \frac{728 - 22}{91} \rfloor$$

## Propagation of equality (Iteration 2)

$$91 * E^{2..7} + 10 * R^{2..8} + D^{2..8} = 90 * N^{3..8} + Y^{2..8}$$

## Propagation of equality (Iteration 2)

$$\underbrace{91 * E^{2..7} + 10 * R^{2..8} + D^{2..8}}_{204..725} = \underbrace{90 * N^{3..8} + Y^{2..8}}_{272..728}$$



## Propagation of equality (Iteration 2)

$$\underbrace{91 * E^{2..7} + 10 * R^{2..8} + D^{2..8}}_{272..725} = 90 * N^{3..8} + Y^{2..8}$$

## Propagation of equality (Iteration 2)

$$\underbrace{91 * E^{2..7} + 10 * R^{2..8} + D^{2..8}}_{272..725} = 90 * N^{3..8} + Y^{2..8}$$

$$E \geq 3 = \lceil \frac{272 - 88}{91} \rceil$$

## Propagation of equality (Iteration 3)

$$91 * E^{3..7} + 10 * R^{2..8} + D^{2..8} = 90 * N^{3..8} + Y^{2..8}$$

## Propagation of equality (Iteration 3)

$$\underbrace{91 * E^{3..7} + 10 * R^{2..8} + D^{2..8}}_{295..725} = \underbrace{90 * N^{3..8} + Y^{2..8}}_{272..728}$$

## Propagation of equality (Iteration 3)

$$\underbrace{91 * E^{3..7} + 10 * R^{2..8} + D^{2..8}}_{295..725} = 90 * N^{3..8} + Y^{2..8}$$

## Propagation of equality (Iteration 3)

$$\underbrace{91 * E^{3..7} + 10 * R^{2..8} + D^{2..8}}_{295..725} = 90 * N^{3..8} + Y^{2..8}$$

$$N \geq 4 = \lceil \frac{295 - 8}{90} \rceil$$

## Propagation of equality (Iteration 4)

$$91 * E^{3..7} + 10 * R^{2..8} + D^{2..8} = 90 * N^{4..8} + Y^{2..8}$$

## Propagation of equality (Iteration 4)

$$\underbrace{91 * E^{3..7} + 10 * R^{2..8} + D^{2..8}}_{295..725} = \underbrace{90 * N^{4..8} + Y^{2..8}}_{362..728}$$



## Propagation of equality (Iteration 4)

$$\underbrace{91 * E^{3..7} + 10 * R^{2..8} + D^{2..8}}_{362..725} = 90 * N^{4..8} + Y^{2..8}$$

## Propagation of equality (Iteration 4)

$$\underbrace{91 * E^{3..7} + 10 * R^{2..8} + D^{2..8}}_{362..725} = 90 * N^{4..8} + Y^{2..8}$$

$$E \geq 4 = \lceil \frac{362 - 88}{91} \rceil$$

## Propagation of equality (Iteration 5)

$$91 * E^{4..7} + 10 * R^{2..8} + D^{2..8} = 90 * N^{4..8} + Y^{2..8}$$

## Propagation of equality (Iteration 5)

$$\underbrace{91 * E^{4..7} + 10 * R^{2..8} + D^{2..8}}_{386..725} = \underbrace{90 * N^{4..8} + Y^{2..8}}_{362..728}$$

## Propagation of equality (Iteration 5)

$$\underbrace{91 * E^{4..7} + 10 * R^{2..8} + D^{2..8}}_{386..725} = 90 * N^{4..8} + Y^{2..8}$$

# Propagation of equality (Iteration 5)

$$\underbrace{91 * E^{4..7} + 10 * R^{2..8} + D^{2..8}}_{386..725} = 90 * N^{4..8} + Y^{2..8}$$

$$N \geq 5 = \left\lceil \frac{386 - 8}{90} \right\rceil$$

## Propagation of equality (Iteration 6)

$$91 * E^{4..7} + 10 * R^{2..8} + D^{2..8} = 90 * N^{5..8} + Y^{2..8}$$

## Propagation of equality (Iteration 6)

$$\underbrace{91 * E^{4..7} + 10 * R^{2..8} + D^{2..8}}_{386..725} = \underbrace{90 * N^{5..8} + Y^{2..8}}_{452..728}$$



## Propagation of equality (Iteration 6)

$$\underbrace{91 * E^{4..7} + 10 * R^{2..8} + D^{2..8}}_{452..725} = 90 * N^{5..8} + Y^{2..8}$$

## Propagation of equality (Iteration 6)

$$\underbrace{91 * E^{4..7} + 10 * R^{2..8} + D^{2..8}}_{452..725} = 90 * N^{5..8} + Y^{2..8}$$

$$N \geq 5 = \lceil \frac{452 - 8}{90} \rceil, E \geq 4 = \lceil \frac{452 - 88}{91} \rceil$$

No further propagation at this point

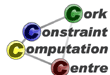
# Domains after setup

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Outline

- 1 Problem
- 2 Program
- 3 Constraint Setup
- 4 Search**
  - Step 1
  - Step 2
  - Further Steps
  - Solution

5 Points to Remember



# labeling built-in

`labeling ([S, E, N, D, M, O, R, Y])`

- Try variable is order given
- Try values starting from smallest value in domain
- When failing, backtrack to last open choice
- *Chronological Backtracking*
- *Depth First search*



# Search Tree Step 1

S  
9  
E

Variable  $S$  already fixed



## Step 2, Alternative $E = 4$

Variable  $E \in \{4..7\}$ , first value tested is 4



# Assignment $E = 4$

	0	1	2	3	4	5	6	7	8	9
S										
E					☀	-	-	-		
N										
D										
M										
O										
R										
Y										



# Propagation of $E = 4$ , equality constraint

$$91 * 4 + 10 * R^{2..8} + D^{2..8} = 90 * N^{5..8} + Y^{2..8}$$

# Propagation of $E = 4$ , equality constraint

$$\underbrace{91 * 4 + 10 * R^{2..8} + D^{2..8}}_{386..452} = \underbrace{90 * N^{5..8} + Y^{2..8}}_{452..728}$$

# Propagation of $E = 4$ , equality constraint

$$\underbrace{91 * 4 + 10 * R^{2..8} + D^{2..8}}_{452} = 90 * N^{5..8} + Y^{2..8}$$

# Propagation of $E = 4$ , equality constraint

$$\underbrace{91 * 4 + 10 * R^{2..8} + D^{2..8}}_{452} = 90 * N^{5..8} + Y^{2..8}$$

$$N = 5, Y = 2, R = 8, D = 8$$

# Result of equality propagation

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Propagation of all different

	0	1	2	3	4	5	6	7	8	9	
S											
E											
N											
D											
M											
O											
R											
Y											

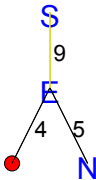
# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N						*	-	-		
D			-	-	-	-	-	-	*	
M										
O										
R			-	-	-	-	-	-	*	
Y			*	-	-	-	-	-		

Alldifferent fails!

## Step 2, Alternative $E = 5$

Return to last open choice,  $E$ , and test next value





# Assignment $E = 5$

	0	1	2	3	4	5	6	7	8	9
S										
E					-	☀	-	-		
N										
D										
M										
O										
R										
Y										

# Propagation of all different

	0	1	2	3	4	5	6	7	8	9
S										
E					-	☀	-	-		
N										
D										
M										
O										
R										
Y										

# Propagation of all different

	0	1	2	3	4	5	6	7	8	9
S										
E						*				
N										
D										
M										
O										
R										
Y										

# Propagation of all different

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

$$N \neq 5, N \geq 6$$

# Propagation of equality

$$91 * 5 + 10 * R^{2..8} + D^{2..8} = 90 * N^{6..8} + Y^{2..8}$$

# Propagation of equality

$$\underbrace{91 * 5 + 10 * R^{2..8} + D^{2..8}}_{477..543} = \underbrace{90 * N^{6..8} + Y^{2..8}}_{542..728}$$

# Propagation of equality

$$\underbrace{91 * 5 + 10 * R^{2..8} + D^{2..8}}_{542..543} = 90 * N^{6..8} + Y^{2..8}$$

# Propagation of equality

$$\underbrace{91 * 5 + 10 * R^{2..8} + D^{2..8}}_{542..543} = 90 * N^{6..8} + Y^{2..8}$$

$$N = 6, Y \in \{2, 3\}, R = 8, D \in \{7..8\}$$



# Result of equality propagation

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Propagation of `alldifferent`

	0	1	2	3	4	5	6	7	8	9	
S											
E											
N											
D											
M											
O											
R											
Y											

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

# Propagation of alldifferent

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D								*		
M										
O										
R										
Y										

# Propagation of `alldifferent`

	0	1	2	3	4	5	6	7	8	9
S										
E										
N										
D										
M										
O										
R										
Y										

$$D = 7$$

# Propagation of equality

$$91 * 5 + 10 * 8 + 7 = 90 * 6 + Y^{2..3}$$

# Propagation of equality

$$\underbrace{91 * 5 + 10 * 8 + 7}_{542} = \underbrace{90 * 6 + Y^{2..3}}_{542..543}$$

# Propagation of equality

$$\underbrace{91 * 5 + 10 * 8 + 7}_{542} = 90 * 6 + Y^{2..3}$$



# Propagation of equality

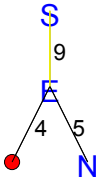
$$\underbrace{91 * 5 + 10 * 8 + 7 = 90 * 6 + Y^{2..3}}_{542}$$

$$Y = 2$$

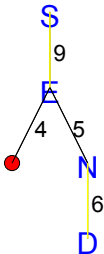
# Last propagation step

	0	1	2	3	4	5	6	7	8	9	
S											
E											
N											
D											
M											
O											
R											
Y											

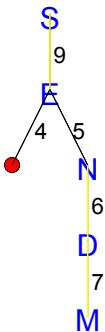
# Further Steps: Nothing more to do



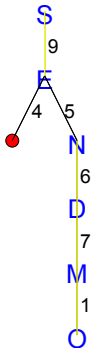
# Further Steps: Nothing more to do



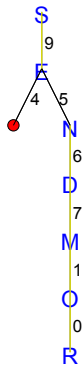
# Further Steps: Nothing more to do



# Further Steps: Nothing more to do



# Further Steps: Nothing more to do

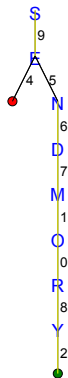


# Further Steps: Nothing more to do

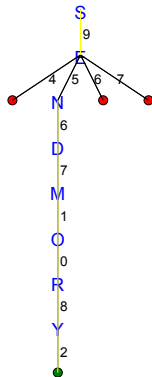




# Further Steps: Nothing more to do



# Complete Search Tree



# Solution

$$\begin{array}{r} 9 \ 5 \ 6 \ 7 \\ + \ 1 \ 0 \ 8 \ 5 \\ \hline 1 \ 0 \ 6 \ 5 \ 2 \end{array}$$

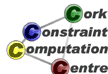
# Outline

- 1 Problem
- 2 Program
- 3 Constraint Setup
- 4 Search
- 5 Points to Remember



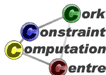
## Points to Remember

- Constraint models are expressed by variables and constraints.
- Problems can have many different models, which can behave quite differently. Choosing the best model is an art.
- Constraints can take many different forms.
- Propagation deals with the interaction of variables and constraints.
- It removes some values that are inconsistent with a constraint from the domain of a variable.
- Constraints only communicate via shared variables.



## Points to Remember

- Propagation usually is not sufficient, search may be required to find a solution.
- Propagation is data driven, and can be quite complex even for small examples.
- The default search uses chronological depth-first backtracking, systematically exploring the complete search space.
- The search choices and propagation are interleaved, after every choice some more propagation may further reduce the problem.



## Part II

# Global Constraints

## Example 2: Sudoku

- Global Constraints
  - Powerful modelling abstractions
  - Non-trivial propagation
  - Different consistency levels
- Example: Sudoku puzzle





# Outline

- 6 Problem
- 7 Initial Propagation (Forward Checking)
- 8 Improved Reasoning
- 9 Search

# Problem Definition

## Sudoku

Fill in numbers from 1 to 9 so that each row, column and block contain each number exactly once

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

# Problem Definition

## Sudoku

Fill in numbers from 1 to 9 so that each row, column and block contain each number exactly once

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

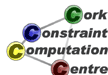
4	2	8	5	6	3	1	7	9
3	5	9	1	7	2	4	6	8
7	6	1	4	8	9	5	3	2
1	4	6	3	9	8	2	5	7
5	9	2	7	4	1	3	8	6
8	3	7	6	2	5	9	4	1
2	7	4	9	5	6	8	1	3
6	8	3	2	1	4	7	9	5
9	1	5	8	3	7	6	2	4

# Model

- A variable for each cell, ranging from 1 to 9
- A 9x9 matrix of variables describing the problem
- Preassigned integers for the given hints
- `alldifferent` constraints for each row, column and 3x3 block

## Reminder: `alldifferent`

- Argument: list of variables
- Meaning: variables are pairwise different
- Reasoning: Forward Checking (FC)
  - When variable is assigned to value, remove the value from all other variables
  - If a variable has only one possible value, then it is assigned
  - If a variable has no possible values, then the constraint fails
  - Constraint is checked whenever one of its variables is assigned
  - Equivalent to decomposition into binary disequality constraints



## Main Program

```
model (Matrix) :-  
    Matrix[1..9,1..9] :: 1..9,  
    (for(I,1,9),  
     param(Matrix) do  
         alldifferent(Matrix[I,1..9]),  
         alldifferent(Matrix[1..9,I])  
    ),  
    (multifor([I,J],[1,1],[7,7],[3,3]),  
     param(Matrix) do  
         alldifferent(flatten(Matrix[I..I+2,J..J+2]))  
    ),  
    flatten_array(Matrix,List),  
    labeling(List).
```

# Domain Visualizer

- Problem shown as matrix
- Each cell corresponds to a variable
- Instantiated: Shows integer value (large)
- Uninstantiated: Shows values in domain

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

# Outline

- 6 Problem
- 7 Initial Propagation (Forward Checking)
- 8 Improved Reasoning
- 9 Search



# Initial State (Forward Checking)

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

# Propagation Steps (Forward Checking)

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

» Skip Animation



# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>1 2 3</sup> 5 6 9	<sup>1 3</sup> 5 7	<sup>1 2</sup> 6 7	<sup>1 3</sup> 5 6 9
<sup>3</sup> 5 6 9	<sup>2</sup> 5 6	<sup>2 3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9	<sup>3</sup> 4 5 8	<sup>2</sup> 4 6 9	<sup>3</sup> 5 6 8 9
<sup>1</sup> 5 6 7 9	<sup>1</sup> 5 6	<sup>1</sup> 5	<sup>4 5</sup> 7 9	8	<sup>1</sup> 5 6 7 9	<sup>4 5</sup> 7	3	2
<sup>1 3</sup> 7 9	<sup>4</sup>	6	<sup>4 3</sup> 7 9	<sup>4 3</sup> 7 9	8	2	5	<sup>1 3</sup> 7 9
<sup>1 3</sup> 7 9	9	<sup>1 2 3</sup> 4 5 7	<sup>2 3</sup> 4 5 7	<sup>2 3</sup> 4 6 7	<sup>1 2 3</sup> 5 6 7	<sup>1 3</sup> 4 5 7	8	<sup>1 3</sup> 5 6 7
<sup>1</sup> 5 8	3	7	6	<sup>2</sup> 4	<sup>1 2</sup> 5	9	<sup>1 2</sup> 4	<sup>1</sup> 5 8
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 7 8	<sup>1 3</sup> 4 6 9	<sup>1 3</sup> 5 6 8 9
<sup>3</sup> 7 8 9	<sup>2</sup> 8	<sup>2 3</sup> 9	<sup>2 3</sup> 7 8 9	<sup>5</sup> 7 8 9	1	4	<sup>3</sup> 7 8	<sup>2</sup> 7 9
<sup>1 3</sup> 7 8 9	<sup>1 2</sup> 5 8	<sup>1 2 3</sup> 5 9	<sup>2 3</sup> 7 8 9	<sup>2 3</sup> 5 9	<sup>1 2 3</sup> 7 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>2 3</sup> 5 6 9	<sup>1 3</sup> 5 7	<sup>1 2</sup> 6 7 9	<sup>1 3</sup> 5 6 9
<sup>3</sup> 5 6 9	<sup>2</sup> 5 6	<sup>2 3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9	<sup>3</sup> 4 5 8	<sup>2</sup> 4 6 9	<sup>3</sup> 5 6 8 9
<sup>1</sup> 5 6 7 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	4 5	8	<sup>1</sup> 5 6 9	<sup>4 5</sup> 7	3	2
<sup>1 3</sup> 7 9	<sup>1</sup> 4	6	<sup>4 3</sup> 7 9	<sup>4 3</sup> 7 9	8	2	5	<sup>1 3</sup> 7 9
<sup>1 3</sup> 5 6 7	9	<sup>1 2 3</sup> 4 5 7	<sup>2 3</sup> 4 5 7	<sup>2 3</sup> 4 6 7	<sup>1 2 3</sup> 5 6 7	<sup>1 3</sup> 4 5 7	8	<sup>1 3</sup> 5 6 7
<sup>1</sup> 5 8	3	7	6	<sup>2</sup> 4	<sup>1 2</sup> 5	9	<sup>1 2</sup> 4	<sup>1</sup> 5 8
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 7 8	<sup>1 3</sup> 4 6 9	<sup>1 3</sup> 6 8 9
<sup>3</sup> 5 6 7 8 9	<sup>2</sup> 5 6 8	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	1	4	<sup>3</sup> 5 7 7 8	<sup>2</sup> 6 7 7 9	<sup>3</sup> 5 6 7 8 9
<sup>1 3</sup> 5 8 7 8 9	<sup>1 2</sup> 5 8	<sup>1 2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9	<sup>1 2 3</sup> 5 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>2 3</sup> 5 6 9 7	<sup>1</sup> 5	<sup>1</sup> 7	<sup>1</sup> 6 5 6 9 7
<sup>3</sup> 5 6 9	<sup>2</sup> 5 6	<sup>2 3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9 8	<sup>4 5</sup> 8	<sup>4</sup> 6 9	<sup>5 6</sup> 8 9
<sup>1</sup> 5 6 7 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	<sup>4 5</sup> 9	8	<sup>5 6</sup> 4 5 9 7	<sup>1</sup> 4 5	3	2
1	4	6	<sup>4</sup> 7 9	<sup>3</sup> 4 9	8	2	5	<sup>1 3</sup> 7 9
5	9	2	<sup>2 3</sup> 4 5 7	<sup>2 3</sup> 4 6	<sup>1 2 3</sup> 5 6 7	<sup>1 3</sup> 4 5 7	8	<sup>1 3</sup> 5 6 7
8	3	7	<sup>2</sup> 4	<sup>1 2</sup> 5	9	<sup>1 2</sup> 4	<sup>1</sup> 5 8	
2	7	<sup>1 3</sup> 4 9 8 9	<sup>3</sup> 4 9 8 9	5	<sup>1 3</sup> 6 4 9 8	<sup>1 3</sup> 4 6 9 8	<sup>1</sup> 6 8 9	<sup>3</sup> 6 8 9
<sup>3</sup> 5 6 7 8 9	<sup>2</sup> 5 6 8	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	1	4	<sup>5 3</sup> 7 8 7 9	<sup>2</sup> 6 9 7 8 9	<sup>3</sup> 5 6 7 8 9
<sup>1</sup> 5 7 8 9	<sup>1 2</sup> 5 8	<sup>1 2 3</sup> 5 9	<sup>2 3</sup> 5 7 8 9	<sup>2 3</sup> 5 9	<sup>1 2 3</sup> 5 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation





# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>2 3</sup> 5 6 9	<sup>1</sup> 5 7	<sup>1</sup> 6 7 9	<sup>1</sup> 5 6 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9	<sup>4 5</sup> 4 8	<sup>4</sup> 6 9	<sup>5 6</sup> 5 8 9
<sup>7</sup> 6 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	<sup>4 5</sup> 4 9	8	<sup>5 6</sup> 4 5 9	<sup>1</sup> 4 5 7	3	2
1	4	6	<sup>7 9</sup> 3 9	<sup>3</sup> 3 9	8	2	5	<sup>3</sup> 7 9
5	9	2	<sup>4 7</sup> 3 7	<sup>4 6</sup> 4 7	<sup>3 1</sup> 3 7	<sup>3 1</sup> 3 7	8	<sup>1 3</sup> 1 6 7
8	3	7	<sup>2 4</sup> 6 4	<sup>1 2</sup> 5 5	9	<sup>1 2</sup> 4 5	<sup>1</sup> 1 5	
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 4 8 9	5	<sup>1 3</sup> 6 4 9	<sup>1 3</sup> 4 8 9	<sup>1 3</sup> 4 6 8 9	<sup>1 3</sup> 6 8 9
<sup>3</sup> 6 7 9	<sup>2</sup> 5 8	<sup>3</sup> 5 9	<sup>2 3</sup> 7 8 9	1	4	<sup>5 3</sup> 7 8 9	<sup>2</sup> 6 7 9	<sup>3</sup> 5 6 8 9
<sup>3</sup> 7 9	<sup>1 2</sup> 5 8	<sup>1</sup> 5 9	<sup>2 3</sup> 7 8 9	<sup>2 3</sup> 5 9	<sup>1 2 3</sup> 7 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>2 3</sup> 5 6 9 7	<sup>1</sup> 5	<sup>1</sup> 7	<sup>1</sup> 6 5 6 9 7
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9 8	<sup>4</sup> 4 5 8	<sup>4</sup> 6 9	<sup>5 6</sup> 5 6 8 9
<sup>7</sup> 6 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	<sup>4 5</sup> 4 5 9	8	<sup>5 6</sup> 4 5 9 7	<sup>1</sup> 4 5	3	2
1	4	6	<sup>3</sup> 7 9	<sup>3</sup> 9	8	2	5	<sup>3</sup> 7 9
5	9	2	<sup>4</sup> 7	<sup>4</sup> 7	<sup>3 1</sup> 3 7	<sup>1 3</sup> 4 7	8	<sup>1 3</sup> 6 7
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 4 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 4 8	<sup>1</sup> 6 9	<sup>3</sup> 8 9
<sup>3</sup> 6 7	<sup>2</sup> 5 8	<sup>3</sup> 5 9	<sup>2 3</sup> 7 8 9	1	4	<sup>5</sup> 7 8	<sup>2</sup> 7 9	<sup>3</sup> 5 6 7 8 9
<sup>3</sup> 7 9	<sup>1 2</sup> 5 8	<sup>1</sup> 5 9	<sup>2 3</sup> 7 8 9	<sup>2 3</sup> 9	<sup>1 2 3</sup> 5 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>3</sup> 6 9	<sup>2 3</sup> 6 9	<sup>1</sup> 6 7	<sup>1</sup> 7 9	<sup>5 6</sup> 7 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 6 9	<sup>4 5</sup> 7 8	<sup>6</sup> 7 9	<sup>5 6</sup> 8 9
<sup>7</sup> 6 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	<sup>4 5</sup> 7 9	8	<sup>6 4 5</sup> 7 9	<sup>3</sup> 7 9	2	
1	4	6	<sup>7</sup> 8 9	<sup>3</sup> 7 9	8	2	5	<sup>3</sup> 7 9
5	9	2	<sup>4</sup> 7 9	<sup>3</sup> 7 9	<sup>3 1</sup> 7 9	<sup>3</sup> 7 9	8	<sup>3</sup> 7 9
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 4 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 7 8	<sup>1</sup> 6 9	<sup>3</sup> 6 8 9
<sup>3</sup> 6 7	<sup>2</sup> 5 8	<sup>3</sup> 5 9	<sup>2 3</sup> 7 8 9	1	4	<sup>5</sup> 7 8	<sup>3 2</sup> 7 9	<sup>3</sup> 5 6 8 9
<sup>7</sup> 6 9	<sup>3 1 2</sup> 5 8	<sup>1 3</sup> 5 9	<sup>2 3</sup> 7 8 9	<sup>3</sup> 7 9	<sup>1 2 3</sup> 7 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>3</sup> 6 9	<sup>2 3</sup> 6 9	<sup>1</sup> 5 7	<sup>1</sup> 6 7	<sup>5 6</sup> 7 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 6 9	<sup>4 5</sup> 8	<sup>6</sup> 9	<sup>5 6</sup> 8 9
<sup>7</sup> 9	<sup>1</sup> 6 9	<sup>1</sup> 5 9	<sup>4 5</sup> 9	8	<sup>6 4 5</sup> 9 7	<sup>3</sup> 8	2	
1	4	6	<sup>7</sup> 9	<sup>3</sup> 9	8	2	5	<sup>3</sup> 7
5	9	2	<sup>4</sup> 7	<sup>3</sup> 7	<sup>3 1</sup> 7	<sup>3</sup> 8	<sup>3</sup> 7	<sup>6</sup> 9
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 4 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 8	<sup>1</sup> 6 9	<sup>3</sup> 8 9
<sup>3</sup> 6 9	<sup>5 6</sup> 8	<sup>3</sup> 9	<sup>2 3</sup> 7 8 9	<sup>5</sup> 9	1	4	<sup>5</sup> 7 8	<sup>3</sup> 6 7 8 9
<sup>3</sup> 9	<sup>1</sup> 5 8	<sup>1</sup> 9	<sup>3</sup> 7 8 9	<sup>2 3</sup> 5 9	<sup>3</sup> 9	<sup>1 2 3</sup> 7 9	<sup>6</sup> 7	<sup>1 2</sup> 9

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>3</sup> 6 9	<sup>2 3</sup> 6 9	<sup>1</sup> 5 7	<sup>1</sup> 6 7	<sup>5 6</sup> 7 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 6 9	<sup>4 5</sup> 8	<sup>6</sup> 9	<sup>5 6</sup> 8 9
<sup>7</sup> 9	<sup>1</sup> 6 9	<sup>1</sup> 5 9	<sup>4 5</sup> 9	8	<sup>6 4 5</sup> 9 7	<sup>3</sup> 8	2	
1	4	6	<sup>7</sup> 9	<sup>3</sup> 9	8	2	5	<sup>3</sup> 7
5	9	2	<sup>4</sup> 7	<sup>3</sup> 7	<sup>3 1</sup> 7	<sup>3</sup> 8	<sup>3</sup> 7	<sup>6</sup> 9
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>3</sup> 9	<sup>1 3</sup> 8	<sup>1</sup> 9	<sup>3</sup> 8 9
<sup>3</sup> 6 9	<sup>5 6</sup> 8	<sup>5</sup> 9	<sup>3 2 3</sup> 7 8 9	1	4	<sup>5</sup> 7 8	<sup>2</sup> 7	<sup>3</sup> 6 8 9
<sup>3</sup> 9	<sup>1</sup> 5 8	<sup>1</sup> 9	<sup>3</sup> 7 8 9	<sup>2 3</sup> 9 7	<sup>3</sup> 9	<sup>2 3</sup> 6 7	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>3</sup> 6 9	<sup>2 3</sup> 6 9	<sup>1</sup> 5 7	<sup>1</sup> 6 7	<sup>5 6</sup> 7 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 6 9	<sup>4 5</sup> 8	<sup>6</sup> 9	<sup>5 6</sup> 8 9
<sup>7</sup> 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	<sup>4 5</sup> 9	8	<sup>6 4 5</sup> 9 7	3	2	
1	4	6	<sup>7</sup> 9	<sup>3</sup> 9	8	2	5	<sup>3</sup> 7
5	9	2	<sup>4</sup> 7	<sup>3</sup> 7	<sup>3 1</sup> 7	<sup>3</sup> 7	8	<sup>3</sup> 6 7
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>3 1</sup> 6 9	<sup>3 1</sup> 8 9	<sup>1</sup> 9	<sup>3</sup> 8 9
<sup>3</sup> 6 9	<sup>5 6</sup> 8	<sup>5</sup> 9	<sup>3 2 3</sup> 7 8 9	1	4	<sup>5 3</sup> 7 8	<sup>2</sup> 7	<sup>3</sup> 5 7 8 9
<sup>3 1</sup> 9	<sup>5</sup> 8	<sup>1 3</sup> 5 9	<sup>2 3</sup> 7 8 9	<sup>3</sup> 9	<sup>2 3</sup> 7 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

## After Setup (Forward Checking)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	3	<sup>2 3</sup> 6 9	<sup>1</sup> 5 7	<sup>1</sup> 6 7	<sup>5 6</sup> 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 6 9	<sup>4 5</sup> 8	<sup>6</sup> 9	<sup>5 6</sup> 8 9
<sup>7</sup> 6 9	<sup>1</sup> 5 6	<sup>1</sup> 5 9	<sup>4 5</sup> 9	8	<sup>1</sup> 6 9	<sup>4 5</sup> 7	3	2
1	4	6	<sup>7</sup> 9	<sup>3</sup> 9	8	2	5	<sup>3</sup> 7
5	9	2	<sup>4</sup> 7	<sup>3</sup> 4	<sup>3</sup> 7	<sup>1 3</sup> 7	<sup>3</sup> 7	8
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4	<sup>3</sup> 9	5	<sup>3</sup> 6 9	<sup>1 3</sup> 8	<sup>1</sup> 9	<sup>3</sup> 8 9
<sup>3</sup> 6 9	<sup>5 6</sup> 8	<sup>3</sup> 9	<sup>2 3</sup> 7 8 9	1	4	<sup>3</sup> 7 8	<sup>2</sup> 7 9	<sup>3</sup> 7 8 9
<sup>3</sup> 9	<sup>1</sup> 5 8	<sup>1</sup> 5 9	<sup>2 3</sup> 7 8 9	3	<sup>2 3</sup> 9	6	<sup>1 2</sup> 7 9	4

# Outline

- 6 Problem
- 7 Initial Propagation (Forward Checking)
- 8 Improved Reasoning**
  - Domain Consistency
  - Comparison
- 9 Search





## Can we do better?

- The alldifferent constraint is missing propagation
  - How can we do more propagation?
  - Do we know when we derive all possible information from the constraint?
- Constraints only interact by changing domains of variables

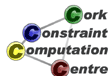


## A Simpler Example

```
:-lib(ic) .
```

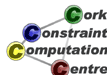
```
top:-
```

```
  X :: 1..2,  
  Y :: 1..2,  
  Z :: 1..3,  
  alldifferent ([X, Y, Z]),  
  writeln ([X, Y, Z]) .
```



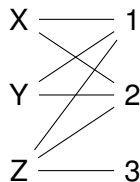
## Using Forward Checking

- No variable is assigned
- No reduction of domains
- But, values 1 and 2 can be removed from Z
- This means that Z is assigned to 3

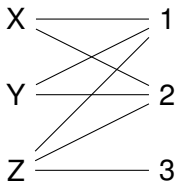


## Visualization of all different as Graph

- Show problem as graph with two types of nodes
  - Variables on the left
  - Values on the right
- If value is in domain of variable, show link between them
- This is called a *bipartite* graph



## A Simpler Example



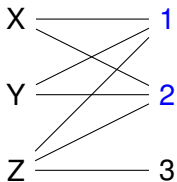
Value Graph for

$X :: 1..2,$

$Y :: 1..2,$

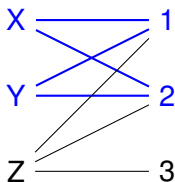
$Z :: 1..3$

## A Simpler Example



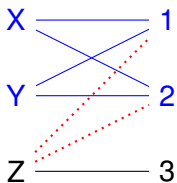
Check interval [1,2]

## A Simpler Example



- Find variables completely contained in interval
- There are two: X and Y
- This uses up the capacity of the interval

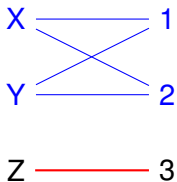
## A Simpler Example



No other variable can use that interval



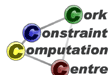
## A Simpler Example



Only one value left in domain of Z,  
this can be assigned

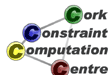
## Idea (Hall Intervals)

- Take each interval of possible values, say size  $N$
- Find all  $K$  variables whose domain is completely contained in interval
- If  $K > N$  then the constraint is infeasible
- If  $K = N$  then no other variable can use that interval
- Remove values from such variables if their bounds change
- If  $K < N$  do nothing
- Re-check whenever domain bounds change



# Implementation

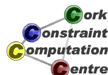
- Problem: Too many intervals ( $O(n^2)$ ) to consider
- Solution:
  - Check only those intervals which update bounds
  - Enumerate intervals incrementally
  - Starting from lowest(highest) value
  - Using sorted list of variables
- Complexity:  $O(n \log(n))$  in standard implementations
- Important: Only looks at min/max bounds of variables



# Bounds Consistency

## Definition

A constraint achieves *bounds consistency*, if for the lower and upper bound of every variable, it is possible to find values for all other variables between their lower and upper bounds which satisfy the constraint.



## Can we do better?

- Bounds consistency only considers min/max bounds
- Ignores “holes” in domain
- Sometimes we can improve propagation looking at those holes



## Another Simple Example

```
:-lib(ic) .
```

```
top:-
```

```
X :: [1,3],
```

```
Y :: [1,3],
```

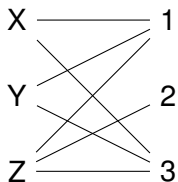
```
Z :: 1..3,
```

```
alldifferent ([X,Y,Z]),
```

```
writeln ([X,Y,Z]) .
```



## Another Simple Example



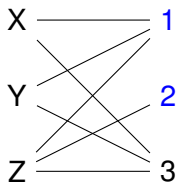
Value Graph for

$X :: [1, 3],$

$Y :: [1, 3],$

$Z :: 1..3$

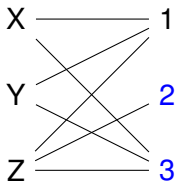
## Another Simple Example



- Check interval  $[1,2]$
- No domain of a variable completely contained in interval
- No propagation

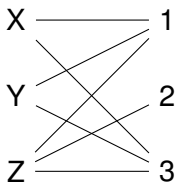


## Another Simple Example



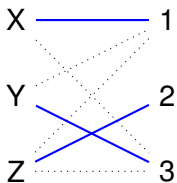
- Check interval [2,3]
- No domain of a variable completely contained in interval
- No propagation

## Another Simple Example



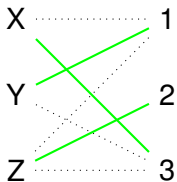
But, more propagation is possible,  
there are only two solutions

## Another Simple Example



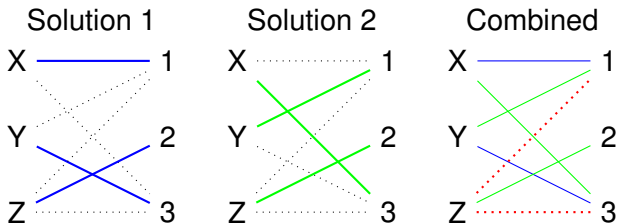
Solution 1: assignment in blue

## Another Simple Example



Solution 2: assignment in green

## Another Simple Example



Combining solutions shows that  $Z=1$  and  $Z=3$  are not possible.

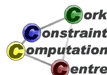
## Another Simple Example

Can we deduce this without enumerating solutions?



## Solutions and maximal matchings

- A *Matching* is subset of edges which do not coincide in any node
- No matching can have more edges than number of variables
- Every solution corresponds to a *maximal matching* and vice versa
- If a link does not belong to some maximal matching, then it can be removed



# Implementation

- Possible to compute all links which belong to some matching
  - Without enumerating all of them!
- Enough to compute **one** maximal matching
- Requires algorithm for *strongly connected components*
- Extra work required if more values than variables
- All links (values in domains) which are not supported can be removed
- Complexity:  $O(n^{1.5}d)$



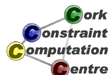


# Domain Consistency

## Definition

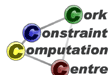
A constraint achieves *domain consistency*, if for every variable and for every value in its domain, it is possible to find values in the domains of all other variables which satisfy the constraint.

- Also called *generalized arc consistency (GAC)*
- or *hyper arc consistency*



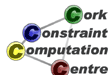
## Can we still do better?

- NO! This extracts all information from this one constraint
- We could perhaps improve speed, but not propagation
- But possible to use different model
- Or model interaction of multiple constraints



# Should all constraints achieve domain consistency?

- Domain consistency is usually more expensive than bounds consistency
  - Overkill for simple problems
  - Nice to have choices
- For some constraints achieving domain consistency is NP-hard
  - We have to live with more restricted propagation



# Initial State (Domain Consistency)

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

# Propagation Steps (Domain Consistency)

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

» Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation





# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation





# Propagation Steps (Domain Consistency)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 7 9	<sup>2 3</sup> 6 7 9	<sup>1 2 3</sup> 5 6 7 9	<sup>1 3</sup> 5 7 9	<sup>1 2</sup> 6 7 9	<sup>1 3</sup> 5 6 7 9
<sup>3</sup> 5 6 9	<sup>2</sup> 5 6	<sup>2 3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9	<sup>3</sup> 4 5 8	<sup>2</sup> 4 6 9	<sup>3</sup> 5 6 8 9
7	<sup>1</sup> 5 6	<sup>1</sup> 5 9	4	8	<sup>1</sup> 5 6 9	<sup>4 5</sup>	3	2
<sup>1 3</sup> 9	4	6	<sup>3</sup> 7 9	<sup>3</sup> 9	8	2	5	<sup>1 3</sup> 7 9
<sup>1 3</sup> 5 6	9	<sup>1 2 3</sup> 4 5 7	<sup>2 3</sup> 4 5 7	<sup>2 3</sup> 4 6 7	<sup>1 2 3</sup> 5 6 7	<sup>1 3</sup> 4 5 7	8	<sup>1 3</sup> 5 6 7
<sup>1</sup> 5 8	3	7	6	<sup>2</sup> 4	<sup>1 2</sup> 5	9	<sup>1 2</sup> 4	<sup>1</sup> 5 8
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 4 8	<sup>1 3</sup> 4 6 9	<sup>1 3</sup> 6 8 9
<sup>3</sup> 5 6 8 9	<sup>2</sup> 5 6 8	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 7 8 9	1	4	<sup>3</sup> 5 7 8	<sup>2</sup> 6 7 9	<sup>3</sup> 5 6 7 8 9
<sup>1 3</sup> 5 8 9	<sup>1 2</sup> 5 9	<sup>1 2 3</sup> 5 7 8 9	<sup>2 3</sup> 7 8 9	<sup>2 3</sup> 9	<sup>1 2 3</sup> 5 7 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>2 3</sup> 5 6 9	<sup>1 3</sup> 5 7	<sup>1 2</sup> 6 9	<sup>1 3</sup> 5 6 9
<sup>3</sup> 5 6 9	<sup>2</sup> 5 6	<sup>2 3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9	<sup>3</sup> 4 5 8	<sup>2</sup> 4 6 9	<sup>3</sup> 5 6 8 9
7	<sup>1</sup> 5 6	<sup>1</sup> 5 9	4	8	<sup>5 6</sup> 9	<sup>1</sup> 5 9	3	2
<sup>1 3</sup> 9	4	6	<sup>7 9</sup>	<sup>3</sup> 9	8	2	5	<sup>1 3</sup> 7 9
<sup>1 3</sup> 5 6	9	<sup>1 2 3</sup> 4 5	<sup>2 3</sup> 7 5	<sup>2 3</sup> 4 6	<sup>1 2 3</sup> 5 6 7	<sup>1 3</sup> 4 5 7	8	<sup>1 3</sup> 5 6 7
<sup>1</sup> 5 8	3	7	6	<sup>2</sup> 4	<sup>1 2</sup> 5	9	<sup>1 2</sup> 4	<sup>1</sup> 5 8
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>1 3</sup> 6 9	<sup>1 3</sup> 4 8	<sup>1</sup> 4 6 9	<sup>1 3</sup> 6 8 9
<sup>3</sup> 5 6 8 9	<sup>2</sup> 5 6 8	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	1	4	<sup>5 3</sup> 7 8	<sup>2</sup> 7 9	<sup>3</sup> 5 6 7 8 9
<sup>1 3</sup> 5 8 9	<sup>1 2</sup> 5 8	<sup>1 2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9	<sup>1 2 3</sup> 5 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>2 3</sup> 6 9	<sup>2 3</sup> 5 6 9 7	<sup>1</sup> 5	<sup>1</sup> 7	<sup>1</sup> 6 5 6 9 7
<sup>3</sup> 5 6 9	<sup>2</sup> 5 6	<sup>2 3</sup> 5 9	1	7	<sup>2 3</sup> 5 6 9 8	<sup>4</sup> 5 8	<sup>4</sup> 6 9	<sup>5 6</sup> 8 9
7	<sup>1</sup> 5 6	<sup>1</sup> 5 9	4	8	<sup>5 6</sup> 9	<sup>1</sup> 5	3	2
1	4	6	<sup>7</sup> 9	<sup>3</sup> 9	8	2	5	<sup>1 3</sup> 7 9
5	9	2	<sup>2 3</sup> 7	<sup>2 3</sup> 5 4 6 7	<sup>1 2 3</sup> 5 6 7	<sup>1 3</sup> 4 5 7	8	<sup>1 3</sup> 5 6 7
8	3	7	6	<sup>2</sup> 4	<sup>1 2</sup> 5	9	<sup>1 2</sup> 4	<sup>1</sup> 5 8
2	7	4	<sup>3</sup> 8 9	5	<sup>1 3</sup> 6 4 9 8	<sup>1 3</sup> 4	<sup>1 3</sup> 6 9 8 9	<sup>1 3</sup> 6 8 9
<sup>3</sup> 5 6 8 9	<sup>2</sup> 5 6 8	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	1	4	<sup>5</sup> 7 8	<sup>2</sup> 7 9	<sup>3</sup> 5 6 7 8 9
<sup>1</sup> 5 8 9	<sup>1 2</sup> 5 8	<sup>1 2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	<sup>2 3</sup> 5 9 7 8 9	<sup>1 2 3</sup> 5 9 7 8 9	6	<sup>1 2</sup> 7 9	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

4	2	8	5	6	3	1	1	1
<sup>3</sup> <sub>6</sub> <sub>9</sub>	5	<sup>3</sup> <sub>5</sub> <sub>9</sub>	1	7	2	4	6	8
7	6	1	4	8	9	5	3	2
1	4	6	<sup>3</sup> <sub>7</sub> <sub>9</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>7</sub> <sub>9</sub>
5	9	2	<sup>3</sup> <sub>7</sub>	4	1	<sup>1</sup> <sub>4</sub> <sub>7</sub>	8	6
8	3	7	6	2	5	9	4	1
2	7	4	<sup>3</sup> <sub>8</sub> <sub>9</sub>	5	6	8	1	<sup>1</sup> <sub>3</sub> <sub>6</sub> <sub>8</sub> <sub>9</sub>
6	8	<sup>3</sup> <sub>5</sub> <sub>9</sub>	2	1	4	<sup>3</sup> <sub>5</sub> <sub>7</sub> <sub>8</sub>	<sup>2</sup> <sub>6</sub> <sub>7</sub> <sub>9</sub>	5
<sup>3</sup> <sub>9</sub>	1	5	8	<sup>2</sup> <sub>3</sub> <sub>9</sub>	7	6	2	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

4	2	8	5	6	3	1		
<sup>3</sup> <sub>9</sub>	5	<sup>3</sup> <sub>9</sub>	1	7	2	4	<sup>7 9</sup> <sub>7 9</sub>	8
7	6	1	4	8	9	5	3	2
1	4	6	<sup>3</sup> <sub>7 9</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>7</sub>
5	9	2	<sup>3</sup> <sub>7</sub>	4	1	<sup>3</sup> <sub>7</sub>	8	6
8	3	7	6	2	5	9	4	1
2	7	4	<sup>3</sup> <sub>9</sub>	5	6	8	1	<sup>3</sup> <sub>9</sub>
6	8	<sup>3</sup> <sub>9</sub>	2	1	4	<sup>3</sup> <sub>7</sub>	<sup>7 9</sup> <sub>7 9</sub>	5
<sup>3</sup> <sub>9</sub>	1	5	8	<sup>3</sup> <sub>9</sub>	7	6	2	4

◀ Back to Start

▶▶ Skip Animation

# Propagation Steps (Domain Consistency)

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

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

4	2	8	5	6	3	1		
<sup>3</sup> <sub>9</sub>	5	<sup>3</sup> <sub>9</sub>	1	7	2	4	<sup>7</sup> <sub>9</sub>	<sup>7</sup> <sub>9</sub>
7	6	1	4	8	9	5	3	2
1	4	6	<sup>3</sup> <sub>7</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>7</sub>
5	9	2	<sup>3</sup> <sub>7</sub>	4	1	<sup>3</sup> <sub>7</sub>	8	6
8	3	7	6	2	5	9	4	1
2	7	4	<sup>3</sup> <sub>9</sub>	5	6	8	1	<sup>3</sup> <sub>9</sub>
6	8	<sup>3</sup> <sub>9</sub>	2	1	4	<sup>3</sup> <sub>7</sub>	<sup>7</sup> <sub>9</sub>	5
<sup>3</sup> <sub>9</sub>	1	5	8	<sup>3</sup> <sub>9</sub>	7	6	2	4

◀ Back to Start

▶▶ Skip Animation



# Propagation Steps (Domain Consistency)

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

◀ Back to Start

## After Setup (Domain Consistency)

4	2	8	5	6	3	1	<sup>7</sup> <sub>9</sub>	<sup>7</sup> <sub>9</sub>
<sup>3</sup> <sub>9</sub>	5	<sup>3</sup> <sub>9</sub>	1	7	2	4	6	8
7	6	1	4	8	9	5	3	2
1	4	6	<sup>3</sup> <sub>7 9</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>7</sub>
5	9	2	<sup>3</sup> <sub>7</sub>	4	1	<sup>3</sup> <sub>7</sub>	8	6
8	3	7	6	2	5	9	4	1
2	7	4	<sup>3</sup> <sub>9</sub>	5	6	8	1	<sup>3</sup> <sub>9</sub>
6	8	<sup>3</sup> <sub>9</sub>	2	1	4	<sup>3</sup> <sub>7 9</sub>	<sup>7</sup> <sub>9</sub>	5
<sup>3</sup> <sub>9</sub>	1	5	8	<sup>3</sup> <sub>9</sub>	7	6	2	4

# Comparison

## Forward Checking

4	1,2	8	2,3	3	2,3	1	1		
3	2	5	3	6	2,3	6	5	1	6
6	5	6	1	7	4	5	6	5	6
7	6	5	9	4	5	9	8	3	2
1	4	6	7	3	3	8	2	5	3
5	9	2	4	4	3	1	3	3	3
8	3	7	6	2	5	9	4	1	
2	7	1	3	3	3	1	3	1	3
6	5	6	5	9	7	8	0	8	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0
3	1	3	2	3	3	2	3	1	3
1	5	6	5	9	7	8	0	8	0
9	8	9	7	8	0	7	8	0	0
5	9	2	4	4	3	1	3	3	3
6	5	6	5	9	7	8	0	8	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0

## Bounds Consistency

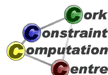
4	1,2	8	5	6	2,3	1	1		
3	2	5	3	6	2,3	7	3	7	9
7	6	1	5	4	8	9	5	3	2
1	4	6	7	3	3	8	2	5	3
5	9	2	7	4	3	1	7	8	6
8	3	7	6	2	5	9	4	1	
2	7	4	8	0	5	6	1	3	1
6	5	6	5	9	7	8	0	8	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0
3	1	3	2	3	3	2	0	8	0
9	8	9	7	8	0	7	8	0	0

## Domain Consistency

4	2	8	5	6	3	1			
3	5	3	1	7	2	4	6	8	
7	6	1	4	8	9	5	3	2	
1	4	6	7	3	3	8	2	5	3
5	9	2	7	4	1	7	8	6	
8	3	7	6	2	5	9	4	1	
2	7	4	8	0	5	6	8	1	3
6	8	3	2	1	4	7	7	9	5
3	1	5	8	3	7	6	2	4	
9	8	9	7	8	0	7	8	0	0
3	1	5	8	3	7	6	2	4	
9	8	9	7	8	0	7	8	0	0
3	1	5	8	3	7	6	2	4	
9	8	9	7	8	0	7	8	0	0
3	1	5	8	3	7	6	2	4	
9	8	9	7	8	0	7	8	0	0

## Typical?

- This does not always happen
- Sometimes, two methods produce same amount of propagation
- Possible to predict in certain special cases
- In general, tradeoff between speed and propagation
- Not always fastest to remove inconsistent values early
- But often required to find a solution at all

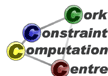


# Outline

- 6 Problem
- 7 Initial Propagation (Forward Checking)
- 8 Improved Reasoning
- 9 Search

## Simple search routine

- Enumerate variables in given order
- Try values starting from smallest one in domain
- Complete, chronological backtracking



# Search Tree (Forward Checking)

2

4	<sup>1 2</sup> 5 6	8	<sup>2 3</sup> 5 9	<sup>3</sup> 6 9	<sup>2 3</sup> 6 9	<sup>1</sup> 5 7	<sup>1</sup> 6 9	<sup>5 6</sup> 7 9
<sup>3</sup> 6 9	<sup>2</sup> 5 6	<sup>3</sup> 5 9	1	7	<sup>2 3</sup> 6 9	<sup>4 5</sup> 8 9	<sup>6</sup> 7 9	<sup>5 6</sup> 8 9
<sup>1</sup> 6 7	<sup>1 5 6</sup> 7 9	<sup>1</sup> 5 9	<sup>4 5</sup> 6 9	8	<sup>1</sup> 6 7	<sup>4 5</sup> 6 9	3	2
1	4	6	<sup>3</sup> 7 9	<sup>3</sup> 6 9	8	2	5	<sup>3</sup> 7 9
5	9	2	<sup>4</sup> 7 9	<sup>4</sup> 7 9	<sup>3 1</sup> 6 7	<sup>3</sup> 7 9	8	<sup>3</sup> 6 7
8	3	7	6	2	5	9	4	1
2	7	<sup>1 3</sup> 4 9	<sup>3</sup> 8 9	5	<sup>3</sup> 6 9	<sup>1 3</sup> 7 8	<sup>3 1</sup> 8 9	<sup>3</sup> 8 9
<sup>3</sup> 6 9	<sup>5 6</sup> 7 8	<sup>3</sup> 5 9	<sup>2 3</sup> 7 8 9	1	4	<sup>3</sup> 7 8	<sup>2</sup> 7 9	<sup>3</sup> 6 7 8 9
<sup>3 1</sup> 6 9	<sup>1 3</sup> 5 8	<sup>1 3</sup> 5 9	<sup>2 3</sup> 7 8 9	<sup>3</sup> 6 7 9	<sup>2 3</sup> 7 9	<sup>1 2</sup> 6 7 9	<sup>1 2</sup> 7 9	4

# Search Tree (Forward Checking)

2  
 1  
 4

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



# Search Tree (Forward Checking)

2  
 1  
 4  
 2  
 5

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

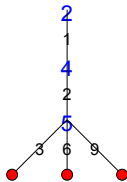
# Search Tree (Forward Checking)



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

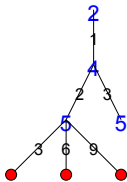


# Search Tree (Forward Checking)



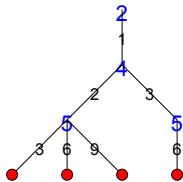
4	1	8	2	9	3	5	6	5	6
3	2	3	1	7	3	4	6	5	6
6	5	9	7	9	6	4	8	9	8
7	9	5	4	5	8	6	4	5	3
1	4	6	7	9	8	2	5	7	3
5	9	2	4	3	1	3	8	7	6
8	3	7	6	2	5	9	4	1	3
2	7	4	9	8	5	6	9	8	9
3	5	6	3	3	1	4	3	2	3
9	8	5	9	7	8	9	7	8	9
3	5	1	3	3	3	2	3	1	2
9	8	5	9	7	8	9	6	7	4

# Search Tree (Forward Checking)



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

# Search Tree (Forward Checking)



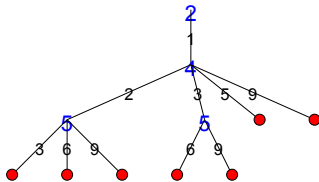
4	1	8	3	6	2	5	6	5
3 2 6 5 6 9	3 5 9	1	7	2 6 9	4 5 8	6 9	7 9	6 5 6 8 9
6 5 6 7 9	5 4 5 9	8	1	6 4 5 9 7	3	2		
1	4	6		3 8	2	5		3
5	9	2	4 4 7	3 1 3 7	3	8		3 6
8	3	7	6	2	5	9	4	1
2	7	1 3 4 9	8 9	5	3 1 3 6 8	1 3 1 8		3 8 9
3 2 6 5 6 9 8	3 2 5 9 7 8 9	1	4	3 2 7 8	5	6	5 6 7 8 9	3 5 8 9
3 5 6 9 8	1 3 2 9 7 8 9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





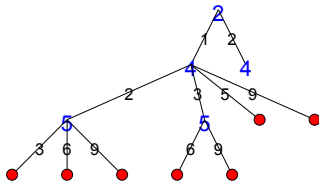


# Search Tree (Forward Checking)



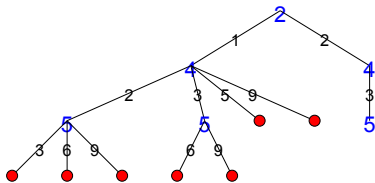
4	1	8	9	3	2	3	5	6	5	6
3	2	3	1	7	2	3	6	7	9	9
6	5	6	9	6	6	4	5	6	5	6
9	9	9	9	9	9	8	9	9	8	9
7	9	9	9	8	6	4	5	3	2	3
1	4	6	8	2	5	5	7	7	6	3
5	9	2	4	7	8	1	3	3	3	3
8	3	7	6	2	5	9	4	1	7	6
2	7	4	9	5	6	9	8	9	8	9
3	5	6	9	1	4	3	2	3	5	3
9	8	9	7	8	7	8	7	9	7	8
3	5	1	3	2	3	3	2	3	1	2
9	8	9	7	8	9	9	7	9	6	4

# Search Tree (Forward Checking)



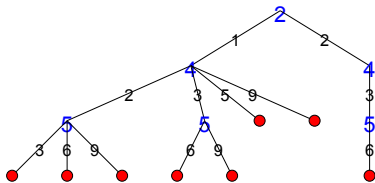
4	2	8	2 3 5 9	3 6 6 9 9	2 3 6 6 9 7	1 5 5 7 7	1 6 6 9 7	5 6 9 9
3 6 6 9 9	2 5 6	3 5 9	1	7	2 3 6 6 9 9	4 5 8 8	6 9 9	5 6 8 9
1 6 6 7 9	5 6	1 5 9	4 5 9 9	8	1 6 4 9 7	4 5 8 8	3	2
1 4 6	6	7 9	3 9 9	8	2	5	3	
5 9	2	4 7	3 7 7	1 3 7 7	3 8	5	3	3
8 3	7 6	2 5	9 4	1				
2 7	1 3 4 9	3 8 9	3 8 9	2 5 6 8	3 9 8	1 3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

# Search Tree (Forward Checking)



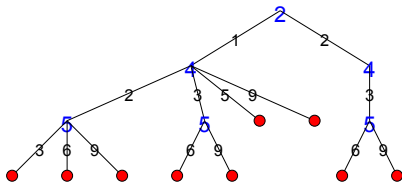
4	2	8	3	3	1	1		
3	5	5	1	7	2	3		
6	9	9	9	6	4	5	6	5
7	6	1	5	4	5	8	6	4
9	9	9	9	9	9	9	9	9
1	4	6	8	2	5			
5	9	2	4	7	9	3	1	3
7	9	9	9	9	9	9	9	9
8	3	7	6	2	5	9	4	1
2	7	4	9	5	6	8	9	8
3	6	5	3	1	4	5	3	2
9	8	9	7	8	9	7	8	9
3	1	1	3	2	3			
9	5	5	9	7	8	9	6	7
9	8	9	7	8	9	9	7	9

# Search Tree (Forward Checking)



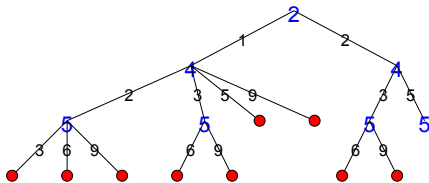
4	2	8	3	6	6	1	1	5	6
3	5	5	1	7	2	6	4	6	5
6	9	9	9	9	9	8	8	9	9
1	5	1	5	4	5	8	6	4	3
7	9	9	9	9	9	9	7	9	2
1	4	6		8	2	5			3
5	9	2	4	4	3	3	3	8	3
8	3	7	6	2	5	9	4	1	6
2	7	1	3	5	3	1	3	1	3
6	5	5	2	1	4	5	3	2	3
9	8	9	7	8	9	7	8	7	9
3	1	1	3	2	3	2	3	1	2
9	5	5	9	7	8	9	9	7	9
						6	7	9	4

# Search Tree (Forward Checking)



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

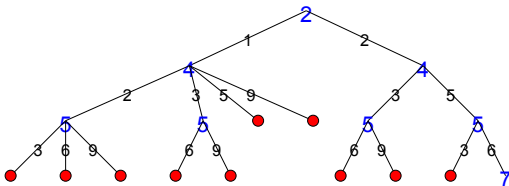
# Search Tree (Forward Checking)



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



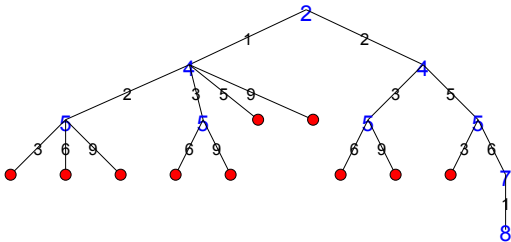
# Search Tree (Forward Checking)



4	2	8	5	6	3	<sup>1</sup>	<sup>1</sup>	<sup>6</sup>	<sup>6</sup>
<sup>3</sup>	<sup>5</sup>	<sup>5</sup>	1	7	2	<sup>4</sup>	<sup>5</sup>	<sup>6</sup>	<sup>5</sup>
<sup>6</sup>	<sup>1</sup>	<sup>1</sup>	4	8	9	<sup>4</sup>	<sup>5</sup>	<sup>6</sup>	<sup>5</sup>
<sup>9</sup>	<sup>9</sup>	<sup>9</sup>	<sup>3</sup>	<sup>3</sup>	<sup>7</sup>	<sup>8</sup>	<sup>3</sup>	<sup>2</sup>	<sup>3</sup>
1	4	6	7	8	9	4	5	3	2
5	9	2	<sup>4</sup>	<sup>4</sup>	1	<sup>7</sup>	8	<sup>7</sup>	<sup>6</sup>
8	3	7	<sup>3</sup>	<sup>3</sup>	2	5	9	4	1
2	7	<sup>1</sup>	<sup>3</sup>	<sup>3</sup>	5	6	<sup>1</sup>	<sup>3</sup>	<sup>1</sup>
<sup>3</sup>	<sup>5</sup>	<sup>5</sup>	<sup>2</sup>	<sup>3</sup>	1	4	<sup>5</sup>	<sup>3</sup>	<sup>2</sup>
<sup>6</sup>	<sup>8</sup>	<sup>9</sup>	<sup>7</sup>	<sup>8</sup>	<sup>9</sup>	<sup>7</sup>	<sup>8</sup>	<sup>7</sup>	<sup>9</sup>
<sup>9</sup>	<sup>1</sup>	<sup>1</sup>	<sup>2</sup>	<sup>3</sup>	1	4	<sup>5</sup>	<sup>6</sup>	<sup>5</sup>
<sup>3</sup>	<sup>5</sup>	<sup>5</sup>	<sup>2</sup>	<sup>3</sup>	<sup>3</sup>	7	6	<sup>1</sup>	<sup>2</sup>
<sup>9</sup>	<sup>8</sup>	<sup>9</sup>	<sup>7</sup>	<sup>8</sup>	<sup>9</sup>	<sup>9</sup>	<sup>7</sup>	<sup>9</sup>	<sup>4</sup>

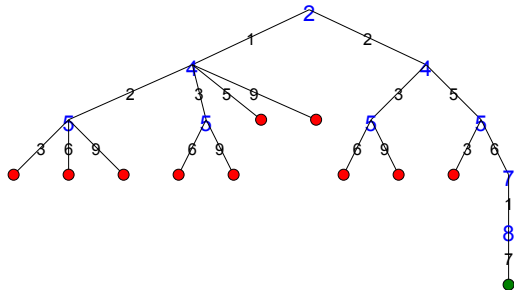


# Search Tree (Forward Checking)



4	2	8	5	6	3	1	<sup>1</sup>		
<sup>3</sup> 6 9	<sup>5</sup> 6 <sup>5</sup> 9	<sup>5</sup> 9	1	7	2	<sup>4</sup> 5 <sup>8</sup> 9	<sup>6</sup> 8 <sup>9</sup> 9	<sup>5</sup> 6 <sup>8</sup> 9	
<sup>7</sup>	<sup>1</sup> 5 <sup>6</sup> 9	<sup>1</sup> 5	4	8	9	<sup>1</sup> 5 <sup>7</sup>	3	2	<sup>3</sup>
1	4	6	<sup>7</sup> 9 <sup>8</sup> 9	<sup>3</sup> 3 <sup>9</sup> 9	8	2	5	<sup>7</sup>	<sup>3</sup>
5	9	2	<sup>7</sup>	<sup>4</sup>	1	<sup>7</sup>	8	<sup>7</sup>	<sup>6</sup>
8	3	7	6	2	5	9	4	1	
2	7	<sup>1</sup> 3 <sup>4</sup> 9 <sup>8</sup> 9	<sup>3</sup> 3 <sup>8</sup> 9	5	6	<sup>1</sup> 3 <sup>8</sup>	<sup>3</sup> 1 <sup>9</sup> 9	<sup>3</sup>	<sup>3</sup>
<sup>3</sup> 6 9	<sup>5</sup> 6 <sup>8</sup> 9	<sup>5</sup> 9	<sup>2</sup> 3 <sup>8</sup> 9	1	4	<sup>5</sup> 5 <sup>7</sup> 8	<sup>3</sup> 2 <sup>7</sup> 9	<sup>5</sup> 6 <sup>8</sup> 9	<sup>3</sup>
<sup>3</sup> 1 <sup>8</sup>	<sup>1</sup> 3 <sup>5</sup> 9	<sup>2</sup> 3 <sup>8</sup> 9	<sup>3</sup>	7	6	<sup>1</sup> 2 <sup>9</sup>			4

# Search Tree (Forward Checking)



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

# Search Tree (Bounds Consistency)

2

4	<sup>1 2</sup>	8	5	6	<sup>2 3 1</sup>	<sup>1</sup>			
<sup>3 2</sup> <sub>9</sub>	<sup>5</sup>	<sup>3</sup>	1	7	<sup>2 3</sup>	<sup>4 5</sup> <sub>8</sub>	<sup>6</sup> <sub>9</sub>	<sup>5</sup> <sub>8 9</sub>	
7	6	<sup>1 5</sup>	4	8	9	<sup>1 5</sup>	3	2	
1	4	6	<sup>3</sup> <sub>7 9</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>7</sub>	
5	9	2	<sup>3</sup> <sub>7</sub>	<sup>4</sup> <sub>3</sub>	1	<sup>3</sup> <sub>7</sub>	8	6	
8	3	7	6	2	5	9	4	1	
2	7	4	<sup>3</sup> <sub>8 9</sub>	5	6	<sup>1 3 1</sup> <sub>8</sub>		<sup>9</sup> <sub>8 9</sub>	
6	<sup>5</sup> <sub>8</sub>	<sup>3</sup> <sub>9 8 9</sub>	<sup>2 3</sup>	1	4	<sup>3 2</sup> <sub>7 8</sub>	<sup>5</sup> <sub>7 9</sub>	<sup>3</sup> <sub>7 8 9</sub>	
<sup>3 1</sup> <sub>9</sub>	<sup>5</sup> <sub>8</sub>	<sup>1 5</sup>	<sup>2</sup> <sub>8</sub>		7	6	<sup>1 2</sup>	4	

# Search Tree (Bounds Consistency)

2  
 ↑  
 ●

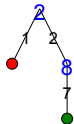
4	1 <sup>1 2</sup>	8	5	6	<sup>2 3</sup> 1	<sup>1</sup>			
<sup>3</sup> 2 9	<sup>2</sup> 5	<sup>3</sup> 9	1	7	<sup>2 3</sup> 7	<sup>4 5</sup> 8	<sup>6</sup> 9	<sup>5</sup> 8 9	
7	6 <sup>1 5</sup>	4	8	9	<sup>1 5</sup> 3	2			
1	4	6	<sup>3</sup> 7 <sup>3</sup> 9	<sup>3</sup> 8	2	5	<sup>3</sup> 7		
5	9	2	<sup>7</sup> 4 <sup>3</sup> 9	<sup>4</sup> 1	<sup>7</sup> 8	6			
8	3	7	6	2	5	9	4	1	
2	7	4	<sup>3</sup> 5 <sup>8 9</sup>	6	<sup>1 3</sup> 7 <sup>1</sup> 8	<sup>9</sup> 8	<sup>8 9</sup>		
6	<sup>5</sup> 8 <sup>8 9</sup>	<sup>3</sup> 2 <sup>2 3</sup> 9	1	4	<sup>3</sup> 7 <sup>5</sup> 8	<sup>2</sup> 9	<sup>3</sup> 7 <sup>8 9</sup>	<sup>5</sup> 8 9	
<sup>3</sup> 9 <sup>1</sup> 8	<sup>1</sup> 5	<sup>2</sup> 8		7	6	<sup>1 2</sup> 4			

# Search Tree (Bounds Consistency)



4	2	8	5	6	3	1	<sup>1</sup>	
<sup>3</sup> <sub>9</sub>	5	<sup>3</sup> <sub>9</sub>	1	7	2	4	6	<sup>5</sup> <sub>8 9</sub>
7	6	1	4	8	9	5	3	2
1	4	6	<sup>3</sup> <sub>7 9</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>7</sub>
5	9	2	<sup>3</sup> <sub>7</sub>	<sup>3</sup> <sub>4</sub>	1	<sup>3</sup> <sub>7</sub>	8	6
8	3	7	6	2	5	9	4	1
2	7	4	<sup>3</sup> <sub>8 9</sub>	5	6	8	1	<sup>3</sup> <sub>8 9</sub>
6	8	<sup>3</sup> <sub>9</sub>	2	1	4	<sup>3</sup> <sub>7 8</sub>	<sup>2</sup> <sub>7 9</sub>	5
<sup>3</sup> <sub>9</sub>	1	5	8	<sup>3</sup> <sub>9</sub>	7	6	2	4

# Search Tree (Bounds Consistency)



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

# Search Tree (Domain Consistency)

8

4	2	8	5	6	3	1			
<sup>3</sup> <sub>9</sub>	5	<sup>3</sup> <sub>9</sub>	1	7	2	4	6	8	
7	6	1	4	8	9	5	3	2	
1	4	6	<sup>3</sup> <sub>9</sub>	<sup>3</sup> <sub>9</sub>	8	2	5	<sup>3</sup> <sub>9</sub>	
5	9	2	<sup>3</sup> <sub>9</sub>	4	1	<sup>3</sup> <sub>9</sub>	8	6	
8	3	7	6	2	5	9	4	1	
2	7	4	<sup>3</sup> <sub>9</sub>	5	6	8	1	<sup>3</sup> <sub>9</sub>	
6	8	<sup>3</sup> <sub>9</sub>	2	1	4	<sup>3</sup> <sub>9</sub>		5	
<sup>3</sup> <sub>9</sub>	1	5	8	<sup>3</sup> <sub>9</sub>	7	6	2	4	

# Search Tree (Domain Consistency)

8  
7  
●

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



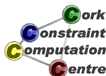
## Tradeoff: nodes and effort

- How many nodes do we need to explore?
- How much effort do we spend in each node?
- Extreme 1: SAT, do very little reasoning in each node, but do many nodes very rapidly
- Extreme 2: MIP, do a lot of reasoning in root node, and in each node, reduce number of nodes to explore
- Constraint Programming: Choice of balance tuneable for problem



# Global Constraint Catalog

- `http://www.emn.fr/z-info/sdemasse/gccat/index.html`
- Description of 399 global constraints, 3250 pages
- Not all of them are widely used
- Detailed, meta-data description of constraints in Prolog



# Key Global Constraints

- alldifferent
- cumulative
- cycle
- diffn
- element
- global\_cardinality
- minimum\_weight\_alldifferent
- nvalue
- sort



## alldifferent ( $L$ )

- A collection of variables  $L$  are pairwise different
- Algorithm: Flow, refinements
- Use: Everywhere
- Similar: `permutation`, `alldifferent_except_0`,  
`lex_alldifferent`



## cumulative(Tasks, Limit)

- A set of tasks with start times  $s_i$ , durations  $d_i$  and resource requirements  $r_i$  do not exceed resource limit *Limit* at any time
- Algorithm: compulsory parts, energy, edge-finding, not-first/not-last
- Use: Scheduling, Placement
- Similar: `disjunctive`

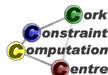
## cycle(N, Succ)

- A graph given by the successors  $s_i$  of nodes  $i$  contains  $N$  cycles
- Algorithm: `alldifferent`, strongly connected components
- Use: Transportation, Scheduling
- Similar: `circuit`, `tree`



## diffn (Obj)

- $n$ -dimensional objects given by origin  $\langle x_{i1}, \dots, x_{in} \rangle$  and size  $\langle d_{i1}, \dots, d_{in} \rangle$  do not overlap
- Algorithm: sweep, compulsory parts
- Use: placement
- Similar: geost



## element( $X, L, C$ )

- $C$  is the  $x_{th}$  element of  $L$
- Algorithm: basic
- Use: functional dependencies, cost
- Similar: `table`



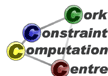
## `global_cardinality(L, Values)`

- Count how often certain values occur in the collection of variables  $L$
- Algorithm: Flow, refinements
- Use: Tlmetabling
- Similar: generalizes `alldifferent`, `among_seq`



```
minimum_weight_alldifferent(L, Matrix,  
Cost)
```

- *Cost* is the cost of the assignment of the variables in *L*, the cost of each entry is given by the *Matrix* of cost values. The entries in *L* are pairwise different.
- Algorithm: Hungarian Method, Flow, Simplex
- Use: Resource allocation
- Similar: `global_cardinality_with_costs`



## `nvalue(N, L)`

- Count the number  $N$  of distinct values in a collection of variables  $L$
- Algorithm: specific, bounds-consistency only
- Use: Assignment problems
- Similar: `same`

## sort( $L, K$ )

- $K$  is the sorted collection of the variables in  $L$
- Algorithm: specific
- Use: Building block for reformulation of other constraints

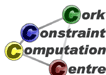
# Why are there so many global constraints?

- Algorithmic aspect
  - More specific restrictions allow more refined algorithms
- Modelling aspect
  - Capture exactly the properties of the problem we are after
- Families of constraints, restrictions and generalizations



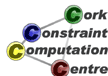
# Constraint Programming to the rescue!

- Constraint Seeker tool (Beldiceanu, Simonis 2011)
- <http://seeker.mines-nantes.fr/>
- Given positive and negative examples, produce ranked list of possible matching global constraints
- Itself a collection of constraint programs



## Basis of modelling tool

- Model Seeker (Beldiceanu, Simonis 2012)
- From positive sample solutions find potential models for problem
- Expressed as conjunction of global constraints
- For highly structured problems
- Do **you** have sample solutions?



## Part III

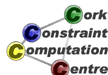
# Customizing Search





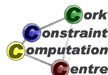
## What we want to introduce

- Importance of search strategy, constraints alone are not enough
- Dynamic variable ordering exploits information from propagation
- Variable and value choice
- Hard to find strategy which works all the time
- `search` builtin, flexible search abstraction
- Different way of improving stability of search routine



# Example Problem

- N-Queens puzzle
- Rather weak constraint propagation
- Many solutions, limited number of symmetries
- Easy to scale problem size



# Outline

- 10 Problem
- 11 Program
- 12 Naive Search
- 13 Improvements



# Problem Definition

## 8-Queens

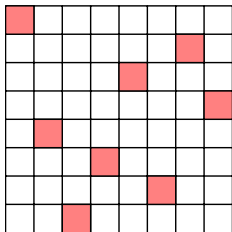
Place 8 queens on an  $8 \times 8$  chessboard so that no queen attacks another. A queen attacks all cells in horizontal, vertical and diagonal direction. Generalizes to boards of size  $N \times N$ .



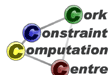
# Problem Definition

## 8-Queens

Place 8 queens on an  $8 \times 8$  chessboard so that no queen attacks another. A queen attacks all cells in horizontal, vertical and diagonal direction. Generalizes to boards of size  $N \times N$ .



Solution for board size  $8 \times 8$



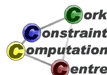
# Outline

- 10 Problem
- 11 Program
  - Model
- 12 Naive Search
- 13 Improvements



# Basic Model

- Cell based Model
  - A 0/1 variable for each cell to say if it is occupied or not
  - Constraints on rows, columns and diagonals to enforce no-attack
  - $N^2$  variables,  $6N - 2$  constraints
- Column (Row) based Model
  - A 1..N variable for each column, stating position of queen in the column
  - Based on observation that each column must contain exactly one queen
  - $N$  variables,  $N^2/2$  binary constraints



# Model

assign  $[X_1, X_2, \dots, X_N]$

s.t.

$$\forall 1 \leq i \leq N : X_i \in 1..N$$

$$\forall 1 \leq i < j \leq N : X_i \neq X_j$$

$$\forall 1 \leq i < j \leq N : X_i \neq X_j + i - j$$

$$\forall 1 \leq i < j \leq N : X_i \neq X_j + j - i$$

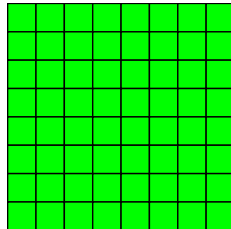


# Outline

- 10 Problem
- 11 Program
- 12 Naive Search
- 13 Improvements

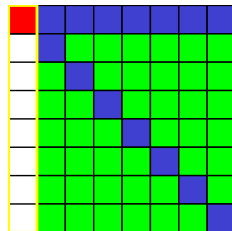
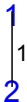
# Default Strategy

1

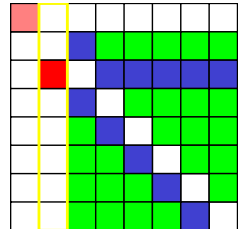
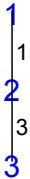


» Skip Animation

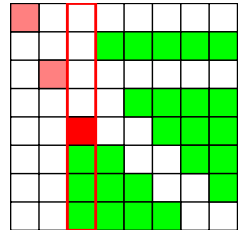
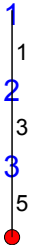
# Default Strategy



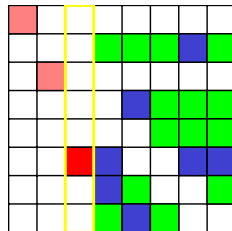
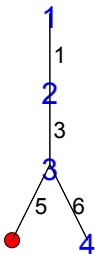
# Default Strategy



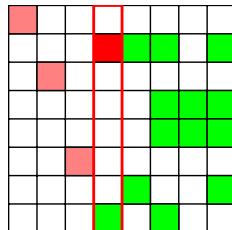
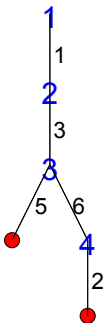
# Default Strategy



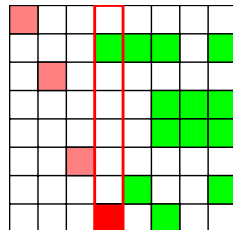
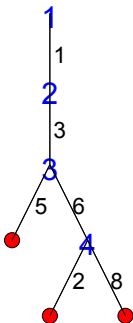
# Default Strategy



# Default Strategy

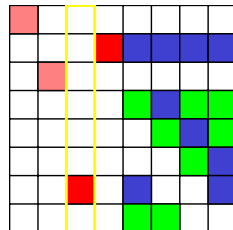
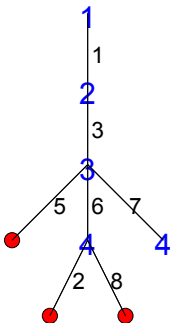


# Default Strategy



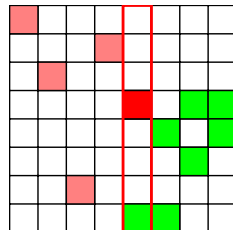
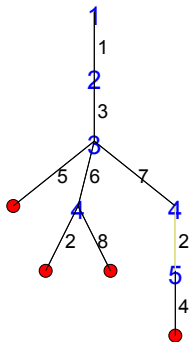


# Default Strategy

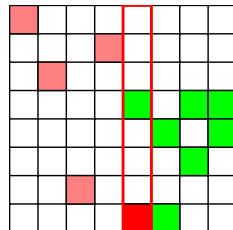
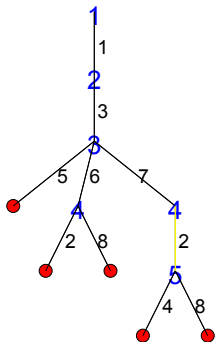




# Default Strategy



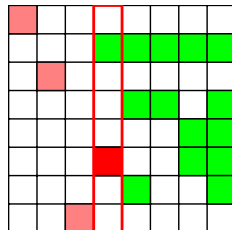
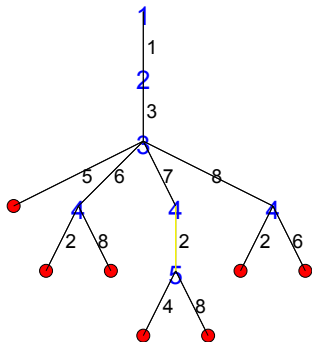
# Default Strategy







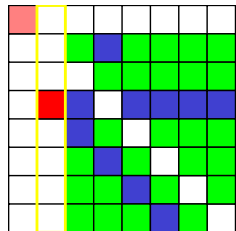
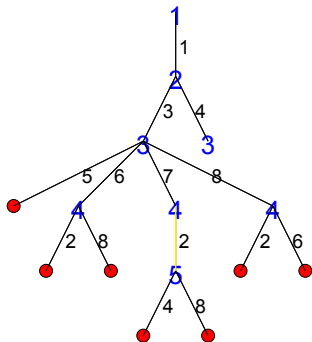
# Default Strategy



◀ Back to Start

▶ Skip Animation

# Default Strategy



◀ Back to Start

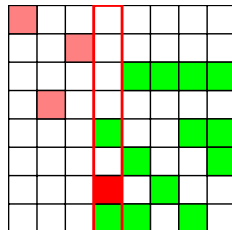
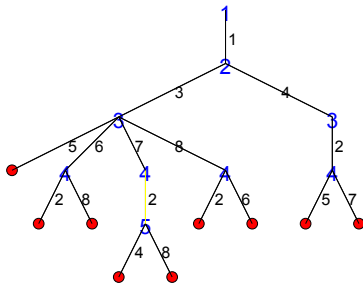
▶ Skip Animation





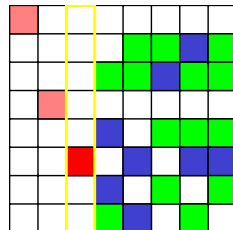
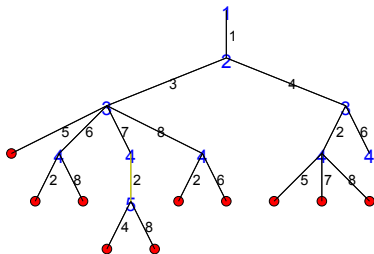


# Default Strategy





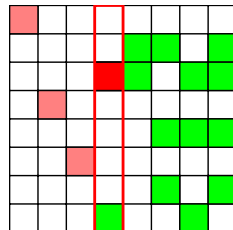
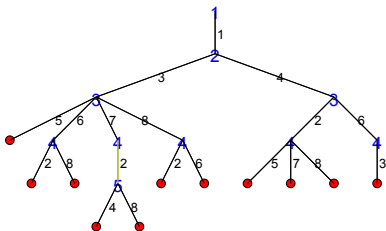
# Default Strategy



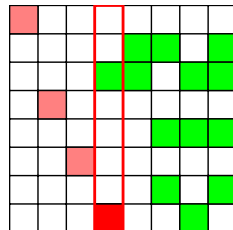
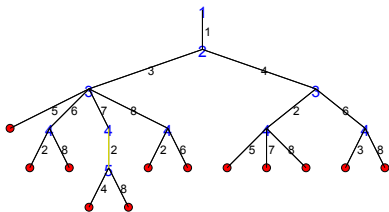
◀ Back to Start

▶ Skip Animation

# Default Strategy



# Default Strategy



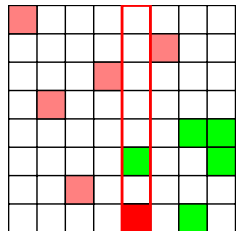
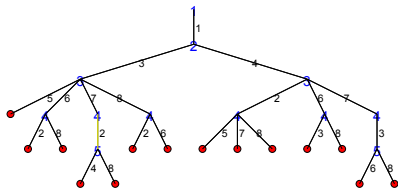








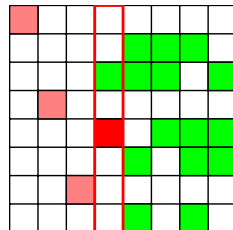
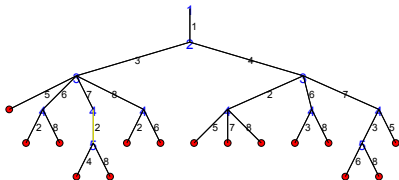
# Default Strategy



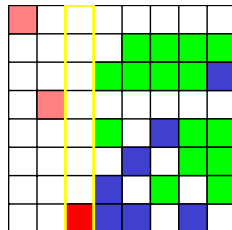
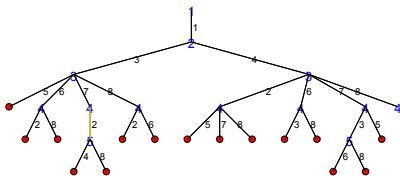
◀ Back to Start

▶▶ Skip Animation

# Default Strategy



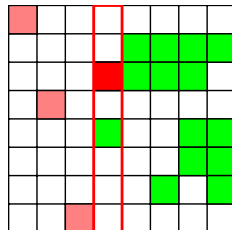
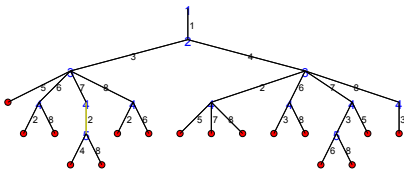
# Default Strategy



◀ Back to Start

▶▶ Skip Animation

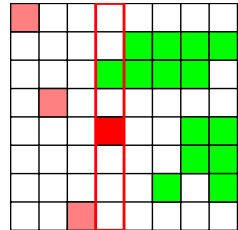
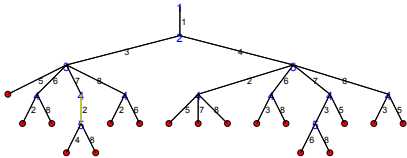
# Default Strategy



◀ Back to Start

▶▶ Skip Animation

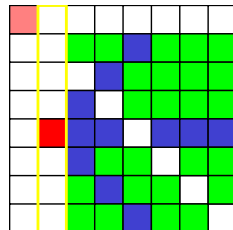
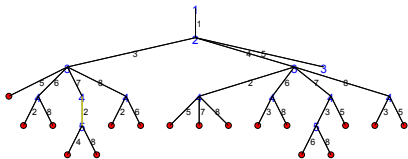
# Default Strategy



◀ Back to Start

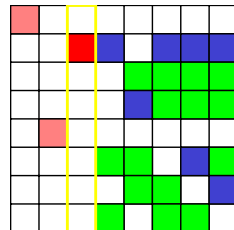
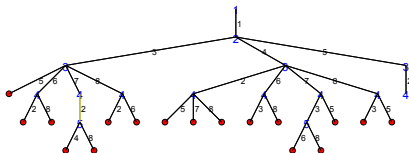
▶▶ Skip Animation

# Default Strategy



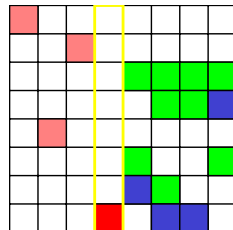
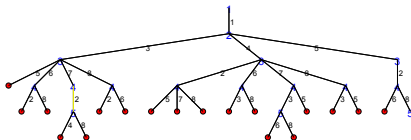


# Default Strategy

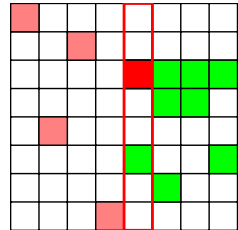
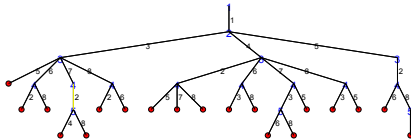




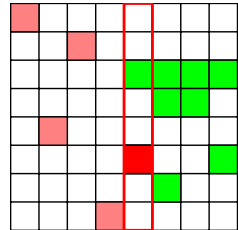
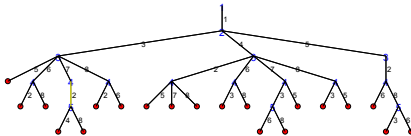
# Default Strategy



# Default Strategy



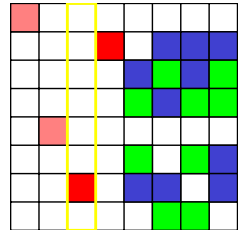
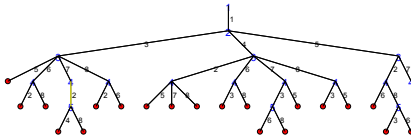
# Default Strategy



◀ Back to Start

▶▶ Skip Animation

# Default Strategy

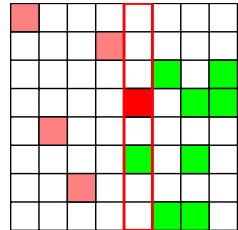
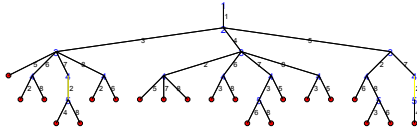


◀ Back to Start

▶▶ Skip Animation



# Default Strategy



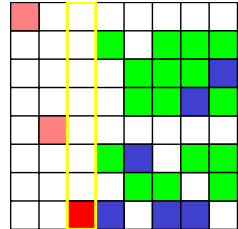
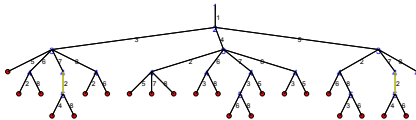
◀ Back to Start

▶▶ Skip Animation

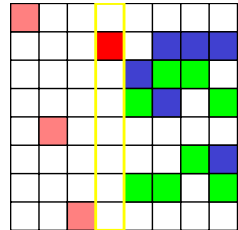
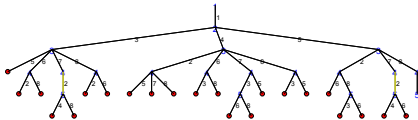




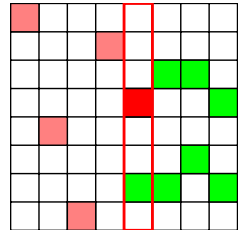
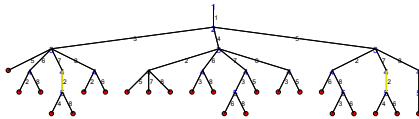
# Default Strategy



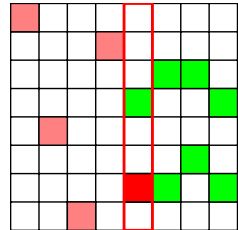
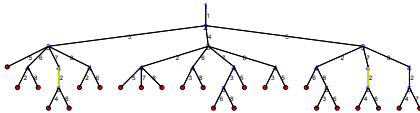
# Default Strategy



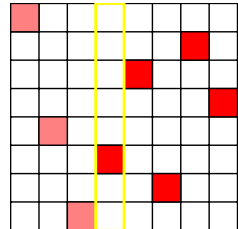
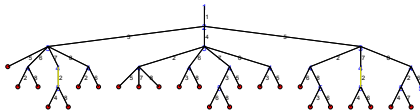
# Default Strategy



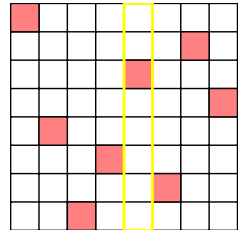
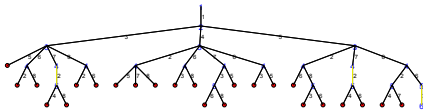
# Default Strategy



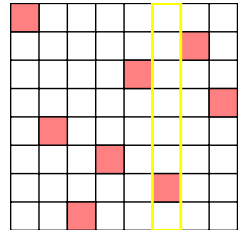
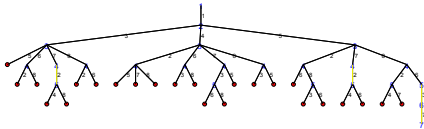
# Default Strategy



# Default Strategy

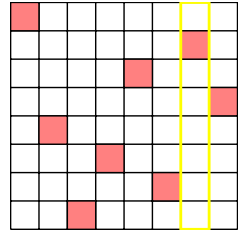
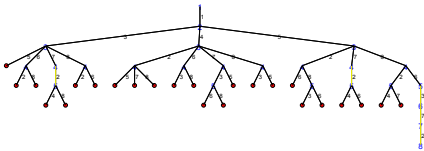


# Default Strategy





# Default Strategy







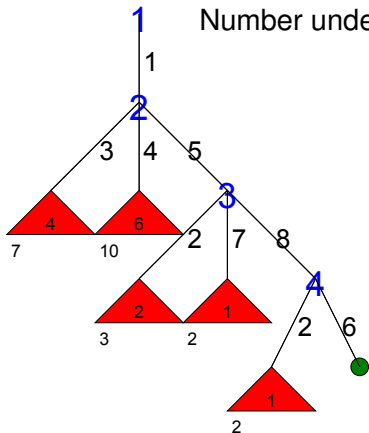
# Observations

- Even for small problem size, tree can become large
- Not interested in all details
- Ignore all automatically fixed variables
- For more compact representation abstract failed sub-trees



# Compact Representation

Number inside triangle: Number of choices  
Number under triangle: Number of failures

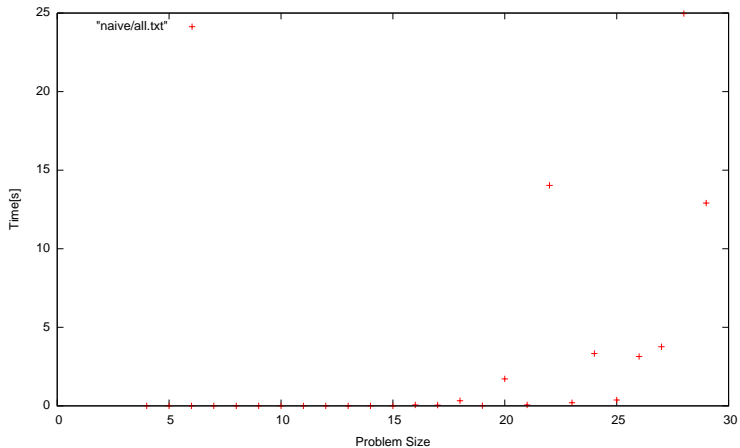


## Exploring other board sizes

- How stable is the model?
- Try all sizes from 4 to 100
- Timeout of 100 seconds



# Naive Strategy, Problem Sizes 4-100



# Observations

- Time very reasonable up to size 20
- Sizes 20-30 times very variable
- Not just linked to problem size
- No size greater than 30 solved within timeout





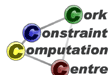
# Outline

- 10 Problem
- 11 Program
- 12 Naive Search
- 13 Improvements
  - Dynamic Variable Choice
  - Improved Heuristics
  - Making Search More Stable



## Possible Improvements

- Better constraint reasoning
  - Remodelling problem with 3 `alldifferent` constraints
  - Global reasoning as described before
  - Not explored here
- Better control of search
  - Static vs. dynamic variable ordering
  - Better value choice
  - Not using complete depth-first chronological backtracking



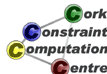
# Static vs. Dynamic Variable Ordering

- Heuristic Static Ordering
  - Sort variables before search based on heuristic
    - Most important decisions
    - Smallest initial domain
- Dynamic variable ordering
  - Use information from constraint propagation
  - Different orders in different parts of search tree
  - Use all information available



# First Fail strategy

- Dynamic variable ordering
- At each step, select variable with smallest domain
- Idea: If there is a solution, better chance of finding it
- Idea: If there is no solution, smaller number of alternatives
- Needs tie-breaking method

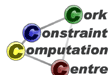


## Modification of Program

```
:-module(nqueen).
:-export(top/0).
:-lib(ic).
```

```
top:-
    nqueen(8,L), writeln(L).
```

```
nqueen(N,L):-
    length(L,N),
    L :: 1..N,
    alldifferent(L),
    noattack(L),
    labeling(L). ↪ replace with
```



## Modification of Program

```
:-module(nqueen).
:-export(top/0).
:-lib(ic).
```

```
top:-
    nqueen(8,L), writeln(L).
```

```
nqueen(N,L):-
    length(L,N),
    L :: 1..N,
    alldifferent(L),
    noattack(L),
    search(L,0,first_fail,indomain,complete,[]).
```



## Variable Choice

- Determines the order in which variables are assigned
- `input_order` assign variables in static order given
- `first_fail` select variable with smallest domain first
- `most_constrained` like `first_fail`, tie break based on number of constraints in which variable occurs
- Others, including programmed selection



## Value Choice

- Determines the order in which values are tested for selected variables
- `indomain` Start with smallest value, on backtracking try next larger value
- `indomain_max` Start with largest value
- `indomain_middle` Start with value closest to middle of domain
- `indomain_random` Choose values in random order



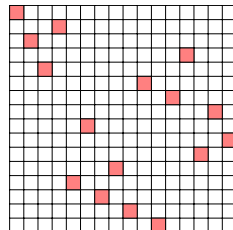
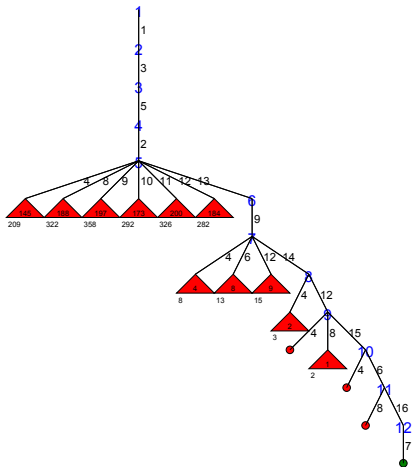


# Comparison

- Board size 16x16
- Naive (Input Order) Strategy
- First Fail variable selection

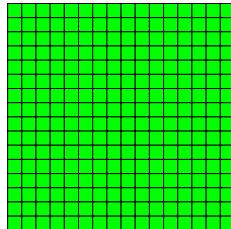


# Naive (Input Order) Strategy (Size 16)



# FirstFail Strategy (Size 16)

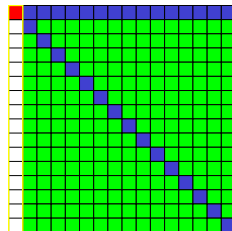
1



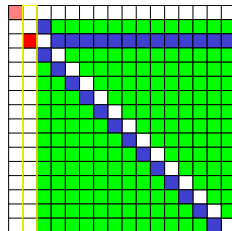
» Skip Animation

# FirstFail Strategy (Size 16)

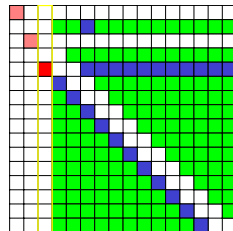
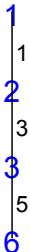
1  
|  
1  
2



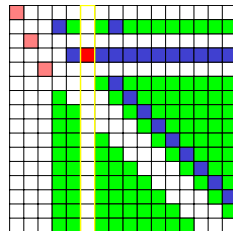
# FirstFail Strategy (Size 16)



# FirstFail Strategy (Size 16)

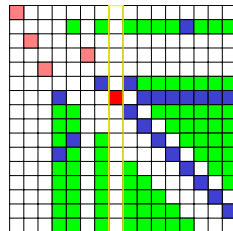


# FirstFail Strategy (Size 16)



# FirstFail Strategy (Size 16)

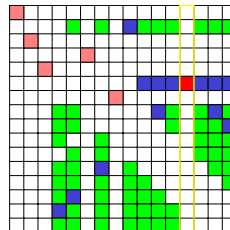
1  
1  
2  
3  
3  
5  
6  
4  
8  
7  
13





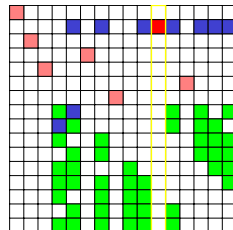
# FirstFail Strategy (Size 16)

1  
1  
2  
3  
3  
5  
6  
4  
8  
7  
13  
6  
11

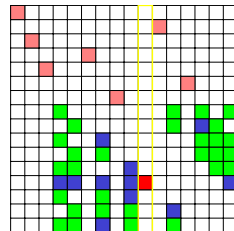
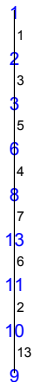


# FirstFail Strategy (Size 16)

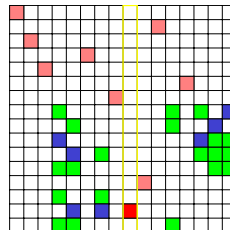
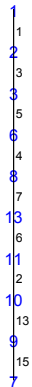
1  
1  
2  
3  
3  
5  
6  
4  
8  
7  
13  
6  
11  
2  
10



# FirstFail Strategy (Size 16)



# FirstFail Strategy (Size 16)

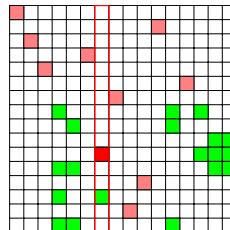
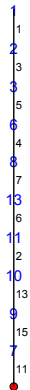


◀ Back to Start

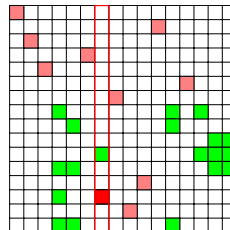
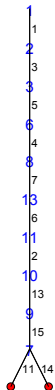
▶▶ Skip Animation



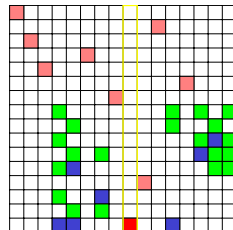
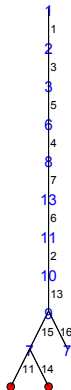
# FirstFail Strategy (Size 16)



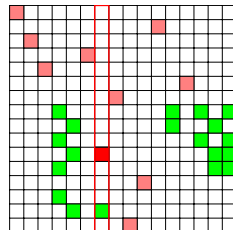
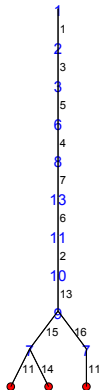
# FirstFail Strategy (Size 16)



# FirstFail Strategy (Size 16)

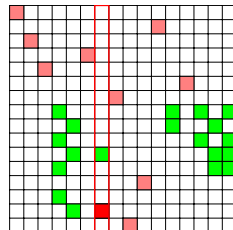
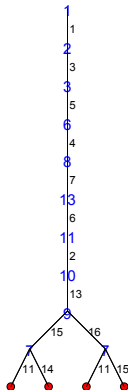


# FirstFail Strategy (Size 16)

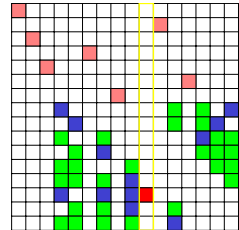
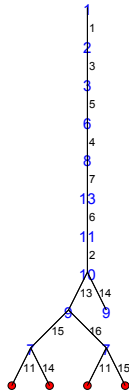




# FirstFail Strategy (Size 16)



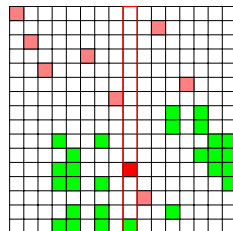
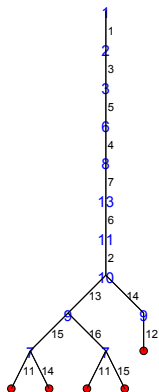
# FirstFail Strategy (Size 16)



◀ Back to Start

▶ Skip Animation

# FirstFail Strategy (Size 16)

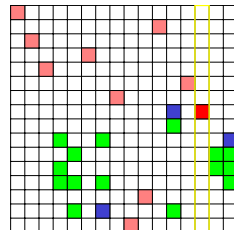
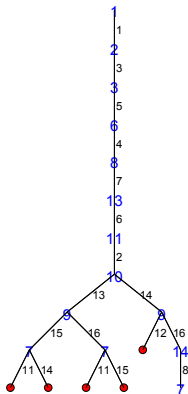


◀ Back to Start

▶▶ Skip Animation



# FirstFail Strategy (Size 16)

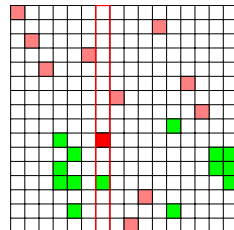
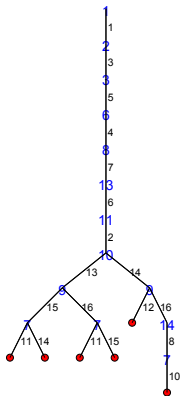


◀ Back to Start

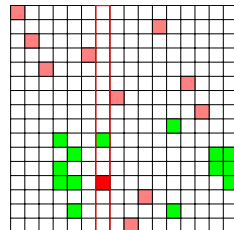
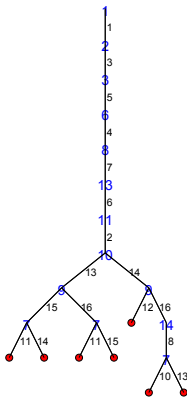
▶ Skip Animation



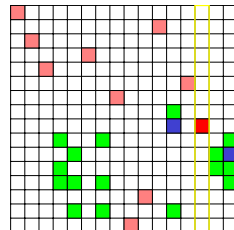
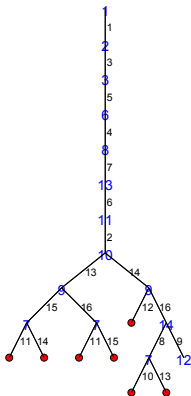
# FirstFail Strategy (Size 16)



# FirstFail Strategy (Size 16)



# FirstFail Strategy (Size 16)



◀ Back to Start

▶ Skip Animation

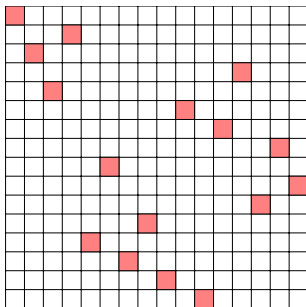




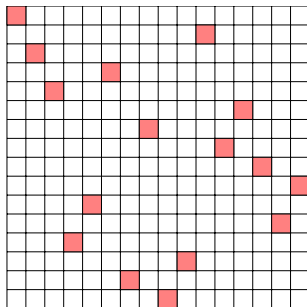


# Comparing Solutions

Naive

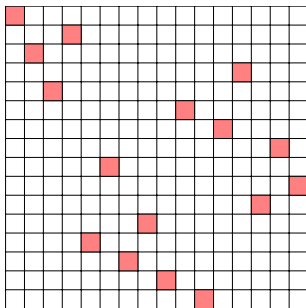


First Fail

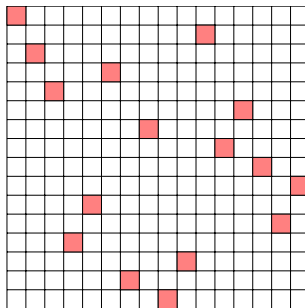


# Comparing Solutions

Naive

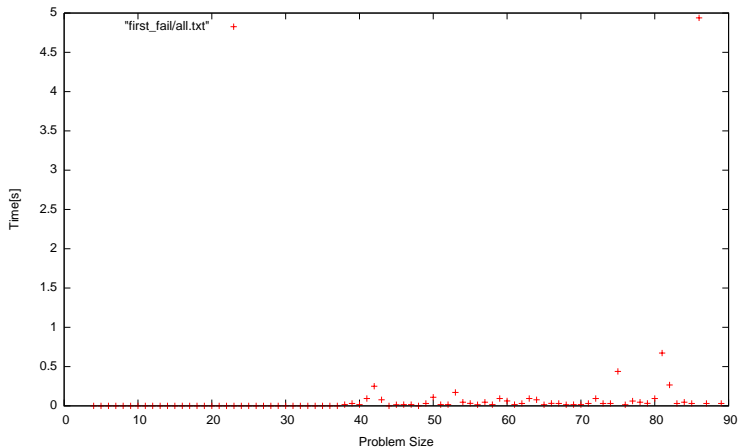


First Fail



Solutions are different!

# FirstFail, Problem Sizes 4-100

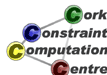


# Observations

- This is much better
- But some sizes are much harder
- Timeout for sizes 88, 91, 93, 97, 98, 99

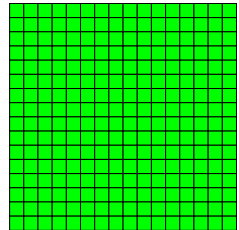
## Can we do better?

- Improved initial ordering
  - Queens on edges of board are easier to assign
  - Do hard assignment first, keep simple choices for later
  - Begin assignment in middle of board
- Matching value choice
  - Values in the middle of board have higher impact
  - Assign these early at top of search tree
  - Use `indomain_middle` for this



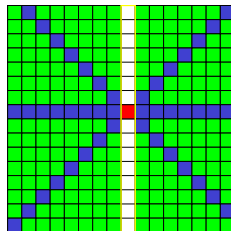
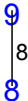
# Start from Middle (Size 16)

9



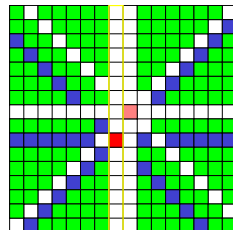
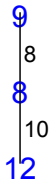
» Skip Animation

# Start from Middle (Size 16)

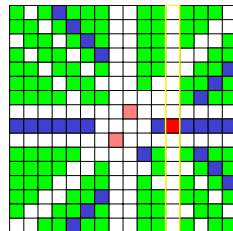
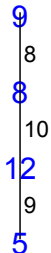




# Start from Middle (Size 16)

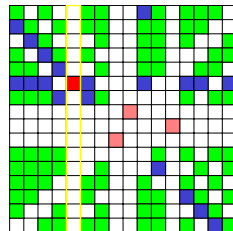


# Start from Middle (Size 16)

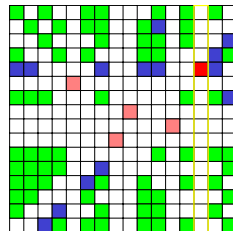
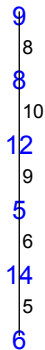


# Start from Middle (Size 16)

9  
8  
8  
10  
12  
9  
5  
6  
14

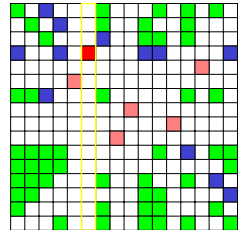


# Start from Middle (Size 16)

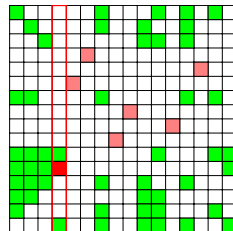
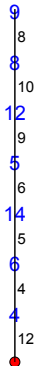


# Start from Middle (Size 16)

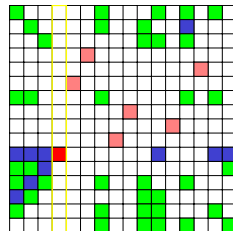
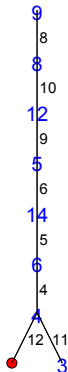
9  
8  
8  
10  
12  
9  
5  
6  
14  
5  
6  
4  
4



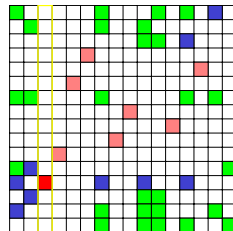
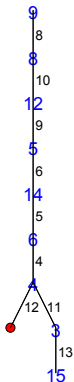
# Start from Middle (Size 16)



# Start from Middle (Size 16)

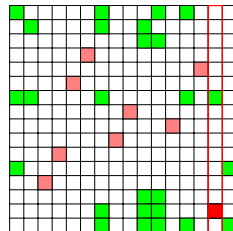
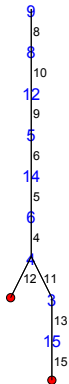


# Start from Middle (Size 16)

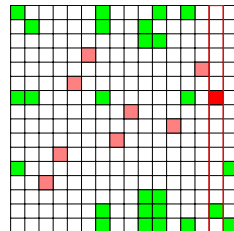
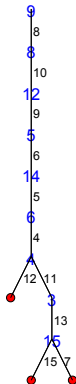




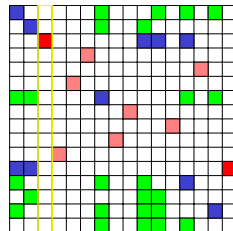
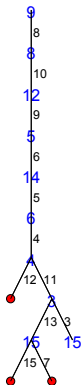
# Start from Middle (Size 16)



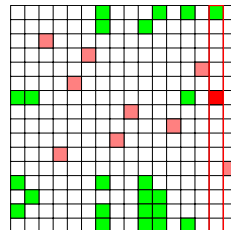
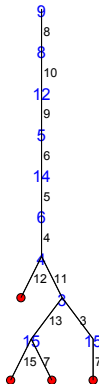
# Start from Middle (Size 16)



# Start from Middle (Size 16)



# Start from Middle (Size 16)

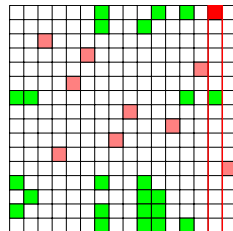
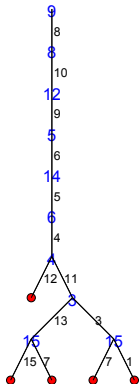


◀ Back to Start

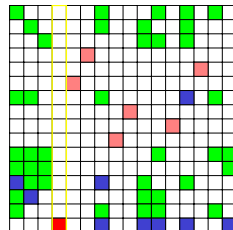
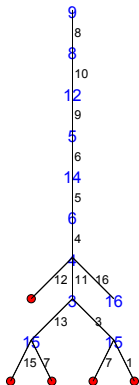
▶▶ Skip Animation



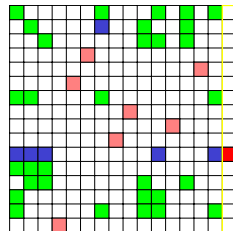
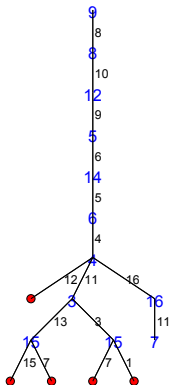
# Start from Middle (Size 16)



# Start from Middle (Size 16)



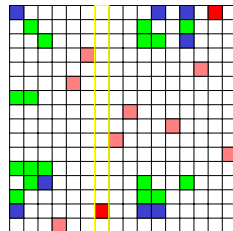
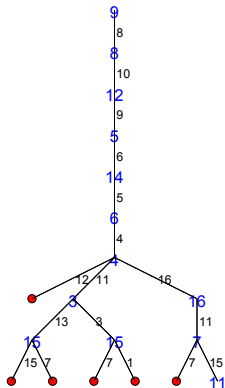
# Start from Middle (Size 16)



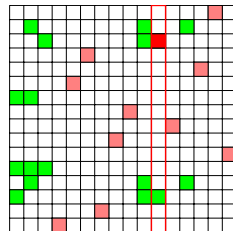
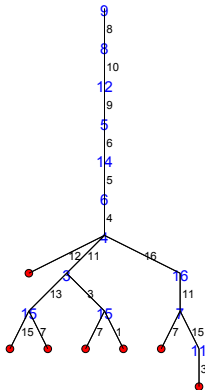




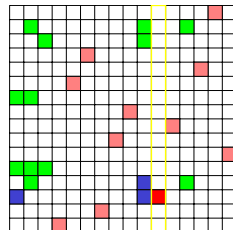
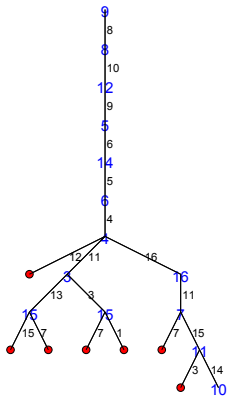
# Start from Middle (Size 16)



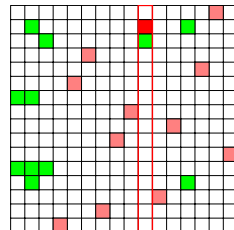
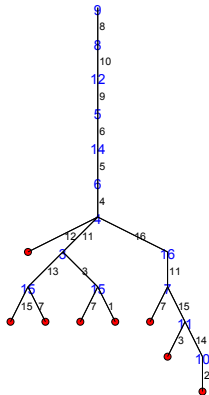
# Start from Middle (Size 16)



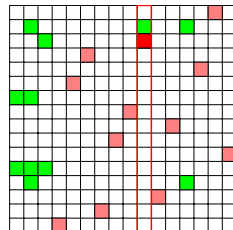
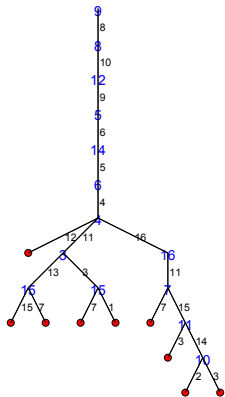
# Start from Middle (Size 16)



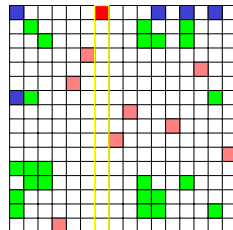
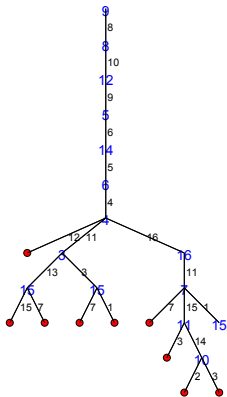
# Start from Middle (Size 16)



# Start from Middle (Size 16)

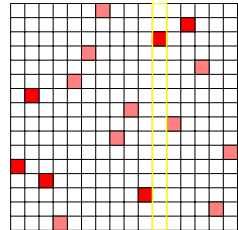
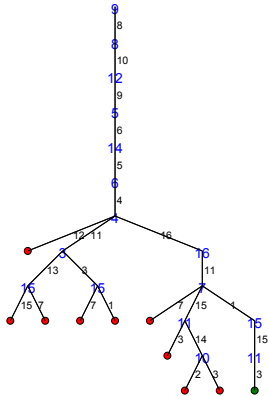


# Start from Middle (Size 16)





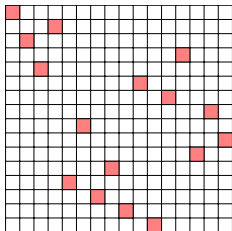
# Start from Middle (Size 16)



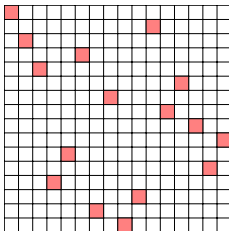


# Comparing Solutions

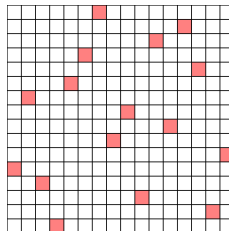
Naive



First Fail

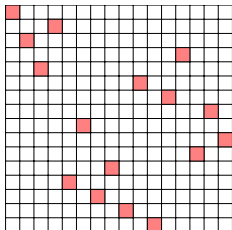


Middle

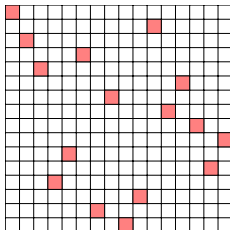


# Comparing Solutions

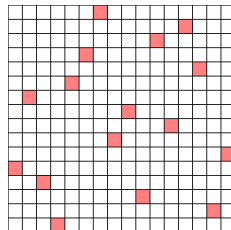
Naive



First Fail

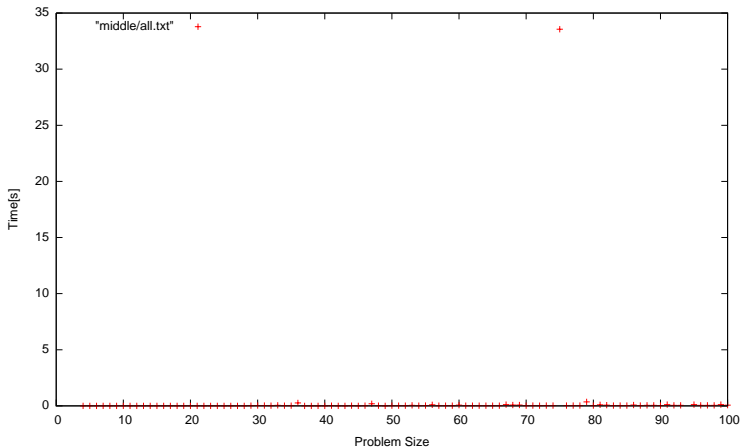


Middle



Again, solutions are different!

# Middle, Problem Sizes 4-100



# Observations

- Not always better than first fail
- For size 16, trees are similar size
- Timeout only for size 94
- But still, one strategy does not work for all problem sizes
- There are ways to resolve this!



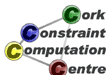
## Approach 1: Heuristic Portfolios

- Try multiple strategies for the same problem
- With multi-core CPUs, run them in parallel
- Only one needs to be successful for each problem



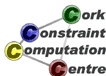
## Approach 2: Restart with Randomization

- Only spend limited number of backtracks for a search attempt
- When this limit is exceeded, restart at beginning
- Requires randomization to explore new search branch
- Randomize variable choice by random tie break
- Randomize value choice by shuffling values
- Needs strategy when to restart



## Approach 3: Partial Search

- Abandon depth-first, chronological backtracking
- Don't get locked into a failed sub-tree
- A wrong decision at a level is not detected, and we have to explore the complete subtree below to undo that wrong choice
- Explore more of the search tree
- Spend time in promising parts of tree



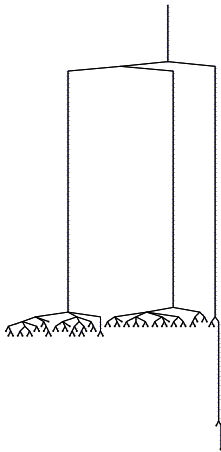
## Example: Credit Search

- Explore top of tree completely, based on credit
- Start with fixed amount of credit
- Each node consumes one credit unit
- Split remaining credit amongst children
- When credit runs out, start bounded backtrack search
- Each branch can use only  $K$  backtracks
- If this limit is exceeded, jump to unexplored top of tree

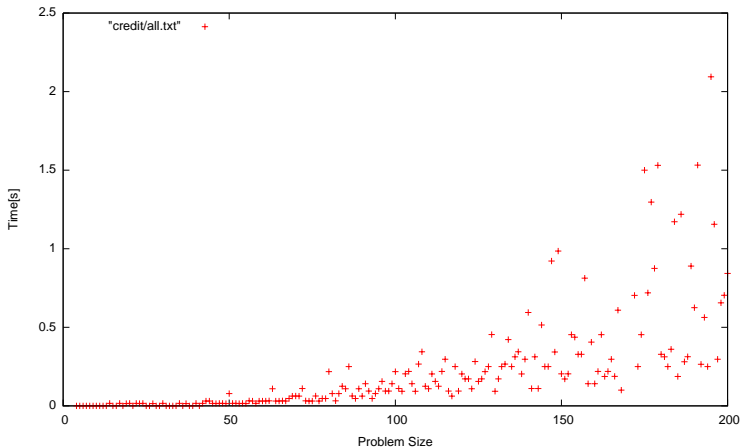




# Credit, Search Tree Problem Size 94



# Credit, Problem Sizes 4-200



## Points to Remember

- Choice of search can have huge impact on performance
- Dynamic variable selection can lead to large reduction of search space
- Packaged search can do a lot, but programming search adds even more
- Depth-first chronological backtracking not always best choice
- How to control this explosion of search alternatives?



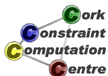
# Part IV

What is missing?



# Many Specialized Topics

- How to design efficient core engine
- Hybrids with LP/MIP tools
- Hybrids with SAT
- Symmetry breaking
- Use of MDD/BDD to encode sets of solutions
- High level modelling tools
- Debugging/visualization



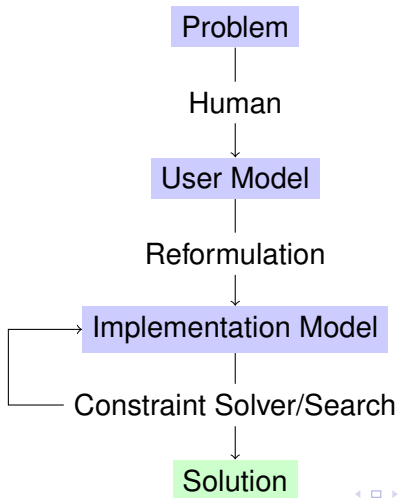
# Reformulation

- Just because the user has modelled it this way, it doesn't mean we have to solve it that way
  - Replace some constraint(s) by other, equivalent constraints
  - Because we don't have that constraint in our system
  - For performance



- While solving the problem we can learn how to strengthen the model/search
  - Understand which constraints/method contribute to propagation and change schedule
  - Learn no-good constraints by explaining failure
  - Adapt search strategy based on search experience

# Refined Process





# What is CP actually used for?

<http://hsimonis.wordpress.com>

The screenshot shows a Mozilla Firefox browser window with the address bar displaying <http://hsimonis.wordpress.com/>. The page title is "Constraint Applications Blog by Helmut Simonis | Lots of Constraint Applications - Mozilla Firefox". The main content area features a large landscape photograph of a snowy town at sunset. Below the photo is a navigation menu with links: Home, Classification, Constraint Systems, NACE Codes, and About. The main article is titled "CP Conference Application Track: Call for Papers" and is dated "Posted on March 23, 2011 by hsimonis". It includes a "Rate This" section with five stars and a "Rate This" button. The article text begins with "If you are reading this blog, then you probably are a prime candidate for the following call for papers:" and provides a link to <http://www.dmi.unipg.it/cp2011/cfa.html>. To the right of the article is an "Email Subscription" section with a text input field, a "Sign me up!" button, and links for "RSS - Posts" and "RSS - Comments". The browser's status bar at the bottom shows "Done".

