Introduction to Constraint Programming

Helmut Simonis

Cork Constraint Computation Centre
Computer Science Department
University College Cork
Ireland

CP Meets CAV, Turunc, June 25th, 2012
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.
The author is partially supported by Science Foundation Ireland (Grant Number 05/IN/I886). 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

- Problem
- Human
- Model
- Constraint Solver/Search
- Solution
More Realistic

Problem → Human → Model → Constraint Solver/Search → Solution/Wrong Solution

- Hangs
- Solution
- Wrong Solution

Helmut Simonis
Introduction to CP
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:

```
  S E N D
+ M O R E
-----
M O N E Y
```
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.
Outline

1. Problem
2. Program
3. Constraint Setup
4. Search
5. Points to Remember
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).
:- 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).
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):-\[Predicate\ definition\]
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).
:- module(sendmory).
:- export(sendmory/1).
:- lib(ic).

sendmory([S,E,N,D,M,O,R,Y], L) :-
  L = [S,E,N,D,M,O,R,Y],
  L :: 0..9,
  alldifferent(L), \[ Digits must be different \]
  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

 sending "S E N D + M O R E = M O N E Y"

:- 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).
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).
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
:- 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).
:- 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)
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

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Domain Visualization

The table represents a domain visualization for the problem of assigning values to variables. The rows correspond to the variables, and the columns represent the domain values (0 to 9). Each cell indicates the possible domain values for the corresponding variable.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Domain Visualization

**Columns = Values**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Domain Visualization

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Cells = State**
Alldifferent Constraint

\[ \text{alldifferent}(L), \]

- Built-in of \texttt{ic} library
- No initial propagation possible
- \textit{Suspends}, waits until variables are changed
- When variable is fixed, remove value from domain of other variables
- \textit{Forward checking}
Alldifferent Visualization

Uses the same representation as the domain visualizer

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Equality Constraint

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

\[
\begin{align*}
1000\times S + 100\times E + 10\times N + D \\
+1000\times M + 100\times O + 10\times R + E \\
10000\times M + 1000\times O + 100\times N + 10\times E + Y
\end{align*}
\]
Normalization

\[
\begin{align*}
1000S & + 100E & + 10N & + D \\
+ 1000M & & & \\
10000M & + 100O & + 10R & + E \\
\hline
10000M & + 100O & + 10N & + 10E & + Y
\end{align*}
\]
Normalization

\[
1000*S + 100*E + 10*N + D
+ 100*O + 10*R + E
\]

\[
9000*M + 1000*O + 100*N + 10*E + Y
\]
Normalization

1000*S+ 100*E+ 10*N+ D
+ 100*O+ 10*R+ E

9000*M+ 1000*O+ 100*N+ 10*E+ Y
Normalization

\[
\begin{align*}
1000*S & + 100*E & + 10*N & + D \\
& + 10*R & & E \\
9000*M & & 900*O & + 100*N & + 10*E & + Y
\end{align*}
\]
Normalization

\[
1000*S + 100*E + 10*N + D + 10*R + E \\
9000*M + 900*O + 100*N + 10*E + Y
\]
Normalization

\[
\begin{align*}
1000*S + 100*E + & \quad D \\
\quad + & \quad 10*R + \quad E \\
9000*M + 900*O + 90*N + & \quad 10*E + \quad Y
\end{align*}
\]
Normalization

\[
\begin{align*}
1000&S+ & 100&E+ & D \\
+ &10&R+ & E \\
9000&M+ & 900&O+ & 90&N+ & 10&E+ & Y
\end{align*}
\]
Normalization

\[
\begin{align*}
1000*S + 91*E + D + 10*R \\
9000*M + 900*O + 90*N + Y
\end{align*}
\]
Simplified Equation

\[1000 \times S + 91 \times E + 10 \times R + D = 9000 \times M + 900 \times O + 90 \times N + Y\]
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

\[
\begin{align*}
1000 \times S^{1..9} &+ 91 \times E^{0..9} + 10 \times R^{0..9} + D^{0..9} = 1000..9918 \\
9000 \times M^{1..9} &+ 900 \times O^{0..9} + 90 \times N^{0..9} + Y^{0..9} = 9000..89919
\end{align*}
\]
Propagation

\[
\begin{align*}
1000 \times S^{1..9} + 91 \times E^{0..9} + 10 \times R^{0..9} + D^{0..9} &= 9000..9918 \\
9000 \times M^{1..9} + 900 \times O^{0..9} + 90 \times N^{0..9} + Y^{0..9} &= 9000..9918
\end{align*}
\]
Propagation

\[
\begin{align*}
1000 \times S^{1..9} + 91 \times E^{0..9} + 10 \times R^{0..9} + D^{0..9} &= 9000..9918 \\
9000 \times M^{1..9} + 900 \times O^{0..9} + 90 \times N^{0..9} + Y^{0..9} &= 9000..9918
\end{align*}
\]

Deduction:

\[M = 1, \ S = 9, \ O \in \{0..1\}\]
Propagation

\[
\begin{align*}
1000 \times S^{1..9} + 91 \times E^{0..9} + 10 \times R^{0..9} + D^{0..9} &= 9000..9918 \\
9000 \times M^{1..9} + 900 \times O^{0..9} + 90 \times N^{0..9} + Y^{0..9} &= 9000..9918
\end{align*}
\]

Deduction:

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

Why? [Skip]
Consider lower bound for $S$

\[
\begin{align*}
1000 \cdot S^{1\ldots9} + 91 \cdot E^{0\ldots9} + 10 \cdot R^{0\ldots9} + D^{0\ldots9} &= 9000 \cdot M^{1\ldots9} + 900 \cdot O^{0\ldots9} + 90 \cdot N^{0\ldots9} + Y^{0\ldots9}
\end{align*}
\]

- Lower bound of equation is 9000
- Rest of lhs (left hand side) $(91 \cdot E^{0\ldots9} + 10 \cdot R^{0\ldots9} + D^{0\ldots9})$ is atmost 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 $S \geq \left\lceil \frac{9000 - 918}{1000} \right\rceil = 9$
- $S$ has upper bound of 9, so $S = 9$
Consider upper bound of $M$

\[
\frac{1000 \times S^{1..9} + 91 \times E^{0..9} + 10 \times R^{0..9} + D^{0..9}}{9000..9918} = \frac{9000 \times M^{1..9} + 900 \times O^{0..9} + 90 \times N^{0..9} + Y^{0..9}}{9000..9918}
\]

- Upper bound of equation is 9918
- Rest of rhs (right hand side) $900 \times O^{0..9} + 90 \times N^{0..9} + Y^{0..9}$ 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 $O$

$$
\frac{1000 \times S_{1..9} + 91 \times E_{0..9} + 10 \times R_{0..9} + D_{0..9}}{9000..9918} = \frac{9000 \times M_{1..9} + 900 \times O_{0..9} + 90 \times N_{0..9} + Y_{0..9}}{9000..9918}
$$

- Upper bound of equation is 9918
- Rest of rhs (right hand side) $9000 \times 1 + 90 \times N_{0..9} + Y_{0..9}$ 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

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>*</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

helmut simonis

introduction to cp
### Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S</strong></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✱</td>
</tr>
<tr>
<td><strong>E</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>N</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>D</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>M</strong></td>
<td>✱</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>O</strong></td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td><strong>R</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Y</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Points to Remember**

- **Domain Definition**
- **Alldifferent Constraint**
- **Disequality Constraints**
- **Equality Constraint**

**Search**

**Constraint Setup**

**Program**

**Problem**

**Domain Definition**

**Alldifferent Constraint**

**Disequality Constraints**

**Equality Constraint**

---

*Helmut Simonis*

*Introduction to CP*
### Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Points to Remember**

- **Domain Definition**
- **Alldifferent Constraint**
- **Disequality Constraints**
- **Equality Constraint**

**Problem**

- Program
- Constraint Setup
- Search
- Points to Remember

**Domain Definition**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Introduction to CP 67**

Helmut Simonis
Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Points to Remember
- Domain Definition
- Alldifferent Constraint
- Disequality Constraints
- Equality Constraint
### Propagation of alldifferent

Here is a table showing the propagation of the alldifferent constraint.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>O</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>O</td>
<td></td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this example, the letters 'S', 'E', 'N', 'D', 'M', 'O', 'R', and 'Y' are set to unique values, with 'M', 'O', and 'R' having specific values assigned to them. The alldifferent constraint ensures that each letter is assigned a unique digit from 0 to 9, with some values fixed at the beginning.
## Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The grid represents a constraint satisfaction problem where each letter represents a digit from 0 to 9. The constraints are as follows:

- **Domain Definition**
  - Each variable (S, E, N, D, M, O, R, Y) must be assigned a unique digit from 0 to 9.

- **Alldifferent Constraint**
  - The alldifferent constraint ensures that each variable is assigned a different digit.

- **Disequality Constraints**
  - Specific disequality constraints are applied to ensure certain digits are not repeated in specific positions.

- **Equality Constraint**
  - Equality constraints may be used to ensure certain digits are equal in specific positions.

The grid shows the initial state of the problem with some constraints already applied, indicated by the filled cells.

*Example:* If we are trying to solve the problem using a constraint satisfaction approach, we would start by assigning values to the variables while ensuring no two variables have the same value and satisfying any additional constraints.
### Propagation of `alldifferent`

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

\[ 1000 \times 9 + 91 \times E^{2..8} + 10 \times R^{2..8} + D^{2..8} = \\
9000 \times 1 + 900 \times 0 + 90 \times N^{2..8} + Y^{2..8} \]
Removal of constants

\[ 91 \times E^{2..8} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{2..8} + Y^{2..8} \]
Propagation of equality (Iteration 1)

\[
91 \times E^{2..8} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{2..8} + Y^{2..8}
\]

\[
\begin{align*}
&91 \times E^{2..8} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{2..8} + Y^{2..8} \\
&204..816 = 182..728
\end{align*}
\]
Propagation of equality (Iteration 1)

\[ 91 \times E^{2..8} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{2..8} + Y^{2..8} \]

204..728
Propagation of equality (Iteration 1)

\[ 91 \times E_{2..8} + 10 \times R_{2..8} + D_{2..8} = 90 \times N_{2..8} + Y_{2..8} \]

\[ 204..728 \]

\[ N \geq 3 = \left\lfloor \frac{204 - 8}{90} \right\rfloor, \quad E \leq 7 = \left\lfloor \frac{728 - 22}{91} \right\rfloor \]
Propagation of equality (Iteration 2)

\[ 91 \times E^{2..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{3..8} + Y^{2..8} \]
Propagation of equality (Iteration 2)

\[
\begin{align*}
91 \times E^{2..7} + 10 \times R^{2..8} + D^{2..8} &= 90 \times N^{3..8} + Y^{2..8} \\
204..725 &= 272..728
\end{align*}
\]
Propagation of equality (Iteration 2)

\[ 91 \times E^{2..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{3..8} + Y^{2..8} \]

\[ 272..725 \]
Propagation of equality (Iteration 2)

\[
91 \times E^{2..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{3..8} + Y^{2..8}
\]

\[
E \geq 3 = \left\lfloor \frac{272 - 88}{91} \right\rfloor
\]
Propagation of equality (Iteration 3)

\[91 \times E^{3..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{3..8} + Y^{2..8}\]
Propagation of equality (Iteration 3)

\[
91 \cdot E_{3..7} + 10 \cdot R_{2..8} + D_{2..8} = 90 \cdot N_{3..8} + Y_{2..8}
\]

\[
= 295..725 + 272..728
\]
Propagation of equality (Iteration 3)

\[ 91 \times E^{3..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{3..8} + Y^{2..8} \]

\[ 295..725 \]
Propagation of equality (Iteration 3)

\[
91 \times E^{3..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{3..8} + Y^{2..8}
\]

\[
N \geq 4 = \left\lfloor \frac{295 - 8}{90} \right\rfloor
\]
Propagation of equality (Iteration 4)

\[ 91 \times E^{3..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8} \]
Propagation of equality (Iteration 4)

\[
91 \cdot E^{3..7} + 10 \cdot R^{2..8} + D^{2..8} = 90 \cdot N^{4..8} + Y^{2..8}
\]

\[
\begin{align*}
295..725 & \quad + \quad 362..728 \\
\end{align*}
\]
Propagation of equality (Iteration 4)

\[
91 \times E^{3..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8}
\]

\[362..725\]
**Propagation of equality (Iteration 4)**

\[
91 \times E^{3..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8}
\]

\[
362..725
\]

\[
E \geq 4 = \left\lfloor \frac{362 - 88}{91} \right\rfloor
\]
Propagation of equality (Iteration 5)

\[ 91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8} \]
Propagation of equality (Iteration 5)

\[ 91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8} \]

386..725 = 362..728
Propagation of equality (Iteration 5)

\[ 91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8} \]

386..725
Propagating equality (Iteration 5)

\[ 91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{4..8} + Y^{2..8} \]

\[ N \geq 5 = \left\lceil \frac{386 - 8}{90} \right\rceil \]
Propagation of equality (Iteration 6)

\[ 91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{5..8} + Y^{2..8} \]
Propagation of equality (Iteration 6)

\[ 91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{5..8} + Y^{2..8} \]
Propagation of equality (Iteration 6)

\[ 91 \cdot E^{4..7} + 10 \cdot R^{2..8} + D^{2..8} = 90 \cdot N^{5..8} + Y^{2..8} \]
\[ 452..725 \]
Propagation of equality (Iteration 6)

\[
\frac{91 \times E^{4..7} + 10 \times R^{2..8} + D^{2..8}}{452..725} = \frac{90 \times N^{5..8} + Y^{2..8}}{90} \\

N \geq 5 = \left\lceil \frac{452 - 8}{90} \right\rceil, \quad E \geq 4 = \left\lceil \frac{452 - 88}{91} \right\rceil
\]

No further propagation at this point
Domains after setup

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>E</td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D</td>
</tr>
<tr>
<td>M</td>
<td></td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M</td>
</tr>
<tr>
<td>O</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>O</td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
</tr>
</tbody>
</table>

Domains after setup.
Outline

1. Problem
2. Program
3. Constraint Setup
4. Search
   - Step 1
   - Step 2
   - Further Steps
   - Solution
5. Points to Remember
Points to Remember

- **labeling built-in**

  \[\text{labeling}([S,E,N,D,M,O,R,Y])\]
  
  - 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*
Variable $S$ already fixed
Step 2, Alternative $E = 4$

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

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>E</strong></td>
</tr>
<tr>
<td><strong>E</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>-</strong></td>
<td><strong>-</strong></td>
</tr>
<tr>
<td><strong>N</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>D</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>M</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>O</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>O</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>R</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>R</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Y</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Helmut Simonis  Introduction to CP
Propagation of $E = 4$, equality constraint

$$91 \cdot 4 + 10 \cdot R^{2..8} + D^{2..8} = 90 \cdot N^{5..8} + Y^{2..8}$$
Propagation of $E = 4$, equality constraint

\[
91 \times 4 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{5..8} + Y^{2..8}
\]

\[
\begin{array}{c}
386..452 \\
452..728
\end{array}
\]
Propagation of $E = 4$, equality constraint

$$91 \times 4 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{5..8} + Y^{2..8}$$
Propagation of $E = 4$, equality constraint

\[ 91 \times 4 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{5..8} + Y^{2..8} \]

\[ 452 \]

$N = 5, Y = 2, R = 8, D = 8$
## Result of equality propagation

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Steps used:

1. **SEN** → **SEN**
2. **N** → **N**
3. **D** → **D**
4. **MORY** → **MORY**
5. **Y** → **Y**

### Constraints

- **SEN** = **SEND**
- **MORY** = **MORY**
- **Y** = **Y**

### Solution

The solution is **SEND = 9567**.
### Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **S** is in the position 0,1.
- **E** is in the position 1,2.
- **N** is in the position 2,3.
- **D** is in the position 3,4.
- **M** is in the position 4,5.
- **O** is in the position 5,6.
- **R** is in the position 6,7.
- **Y** is in the position 7,8.

**Points to Remember**

- **Step 1**: Initialize the problem setup.
- **Step 2**: Implement the alldifferent constraint.
- **Further Steps**: Solve the propagation.
Propagating the alldifferent constraint:

```
<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Alldifferent fails!
Step 2, Alternative $E = 5$

Return to last open choice, $E$, and test next value
Assignment $E = 5$
## Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>-</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The alldifferent constraint ensures that all variables in a given set take different values. The propagation process involves the elimination of illegal values from the domains of the variables. This is often visualized by highlighting illegal values and propagating the constraints across the board.
Propagating the alldifferent constraint

Given the puzzle:

```
SEND
MORY
0123456789
S E N D M O R Y
```

The alldifferent constraint requires that each digit from 0 to 9 appears exactly once.

- **Step 1**: Identify the constraints. For example, the digits in SEND must be unique.
- **Step 2**: Propagate the constraints. For example, if E is 2, then S, N, D cannot be 2.
- **Further Steps**: Continue propagating and checking for solutions.

Helmut Simonis
Introduction to CP
## Propagation of alldifferent

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ N \neq 5, \; N \geq 6 \]
Propagation of equality

\[ 91 \times 5 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{6..8} + Y^{2..8} \]
Propagation of equality

\[
91 \times 5 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{6..8} + Y^{2..8}
\]
Propagation of equality

\[
91 \times 5 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{6..8} + Y^{2..8}
\]

\[
542..543
\]
Propagation of equality

\[
91 \times 5 + 10 \times R^{2..8} + D^{2..8} = 90 \times N^{6..8} + Y^{2..8}
\]

542..543

\[N = 6, \ Y \in \{2, 3\}, \ R = 8, \ D \in \{7..8\}\]
Result of equality propagation

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S × × × × × × × × ×
E × × × × × × × × ×
N × × × × × × × × ×
D × × × × × × × × ×
M × × × × × × × × ×
O × × × × × × × × ×
R × × × × × × × × ×
Y × × × × × × × × ×

Helmut Simonis  Introduction to CP
Problem
Program
Constraint Setup
Search
Points to Remember

Step 1
Step 2
Further Steps
Solution

Propagation of \textit{alldifferent}

\begin{center}
\begin{tabular}{cccccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\hline
S & & & & & & & & & & \textcolor{red}{\star} \\
E & & & & & & & & & & \textcolor{red}{\star} \\
N & & & & & & & & & & \textcolor{red}{\star} \\
D & & & & & & & & & & \\
M & & & & & & & & & & \\
O & & & & & & & & & & \\
R & & & & & & & & & & \textcolor{red}{\star} \\
Y & & & & & & & & & & \textcolor{red}{\star} \\
\end{tabular}
\end{center}
### Propagation of *alldifferent*

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>E</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>1</strong></td>
</tr>
<tr>
<td><strong>N</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>2</strong></td>
</tr>
<tr>
<td><strong>D</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>3</strong></td>
<td><strong>4</strong></td>
</tr>
<tr>
<td><strong>M</strong></td>
<td></td>
<td><strong>5</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>O</strong></td>
<td></td>
<td><strong>6</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>R</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>7</strong></td>
<td><strong>8</strong></td>
</tr>
<tr>
<td><strong>Y</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>9</strong></td>
</tr>
</tbody>
</table>

Constraints:
- **S** = 0
- **E** = 1
- **N** = 2
- **D** = 3
- **M** = 4
- **O** = 5
- **R** = 7
- **Y** = 8

Further constraints:
- **R** = 9
- **D** = 6
- **M** = 4

**Notes:**
- The gray cells represent constraints that have been applied.
- The white cells represent potential solutions that are under consideration.
- The red cells indicate the final solution.
Propagation of \textit{alldifferent}\textsuperscript{'}
### Propagation of `alldifferent`

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ D = 7 \]
Propagation of equality

\[ 91 \times 5 + 10 \times 8 + 7 = 90 \times 6 + Y^{2..3} \]
Propagation of equality

\[
\underbrace{91 \times 5 + 10 \times 8 + 7}_{542} = \underbrace{90 \times 6 + Y^{2..3}}_{542..543}
\]
Propagation of equality

\[ 91 \times 5 + 10 \times 8 + 7 = 90 \times 6 + Y^{2..3} \]

542
Propagation of equality

\[
\underbrace{91 \times 5 + 10 \times 8 + 7} = \underbrace{90 \times 6 + Y^2 \ldots 3}
\]

542

\[Y = 2\]
### Last propagation step

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Solution: `SENDORY`

- The target word is `SENDORY`.

- The puzzle grid is completed with the solution:
  - `S` in row 1, column 0
  - `E` in row 2, column 0
  - `N` in row 3, column 0
  - `D` in row 4, column 0
  - `M` in row 5, column 0
  - `O` in row 6, column 0
  - `R` in row 7, column 0
  - `Y` in row 8, column 0
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

Helmut Simonis  Introduction to CP  138
Solution

\[
\begin{array}{cccc}
9 & 5 & 6 & 7 \\
+ & 1 & 0 & 8 & 5 \\
\hline
1 & 0 & 6 & 5 & 2 \\
\end{array}
\]
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.
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
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
Problem Definition

Sudoku

Fill in numbers from 1 to 9 so that each row, column and block contain each number exactly once
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: \textit{all different}

- 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
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).
**Problem**

Initial Propagation (Forward Checking)
Improved Reasoning
Search

---

**Domain Visualizer**

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

6 Problem

7 Initial Propagation (Forward Checking)

8 Improved Reasoning

9 Search
### Initial State (Forward Checking)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
</tbody>
</table>

**Problem:**
- Initial Propagation (Forward Checking)
- Improved Reasoning
- Search

---

**Introduction to CP 153**

Helmut Simonis
### Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search

---

#### Propagation Steps (Forward Checking)

- **Step 1:**
  - Propagation starts with variable `4`.
  - Values `4, 5, 6` are propagated.

- **Step 2:**
  - Variable `8` is updated.
  - Values `1, 2, 3` are propagated.

- **Step 3:**
  - Variable `1` is updated.
  - Values `7, 8, 9` are propagated.

- **Step 4:**
  - Variable `7` is updated.
  - Values `4, 5, 6` are propagated.

- **Step 5:**
  - Variable `6` is updated.
  - Values `5, 6, 7` are propagated.

- **Step 6:**
  - Variable `5` is updated.
  - Values `5, 6, 7` are propagated.

- **Step 7:**
  - Variable `4` is updated.
  - Values `4, 5, 6` are propagated.

- **Step 8:**
  - Variable `3` is updated.
  - Values `1, 2, 3` are propagated.

- **Step 9:**
  - Variable `2` is updated.
  - Values `1, 2, 3` are propagated.

- **Step 10:**
  - Variable `1` is updated.
  - Values `1, 2, 3` are propagated.

---

*Helmut Simonis*

*Introduction to CP*
Propagation Steps (Forward Checking)
Problem

Initial Propagation (Forward Checking)

Improved Reasoning

Search

Propagation Steps (Forward Checking)
Propagation Steps (Forward Checking)
Propagation Steps (Forward Checking)
### Initial Propagation (Forward Checking)

- **Search**
- **Improved Reasoning**

### Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Back to Start**
- **Skip Animation**

---

**Helmut Simonis**

**Introduction to CP**
### Propagation Steps (Forward Checking)

| 4 | 1 2 5 6 | 8 | 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 |
|---|---|---|---|---|---|---|---|---|
| 2 3 4 5 6 9 | 2 3 4 5 6 9 | 2 3 4 5 6 9 | 2 3 4 5 6 9 | 2 3 4 5 6 9 | 2 3 4 5 6 9 | 2 3 4 5 6 9 | 2 3 4 5 6 9 |
| 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 |
| 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 |
| 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 |
| 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 |
| 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 | 1 2 3 5 6 9 |

---

**Problem**

Initial Propagation (Forward Checking)

Improved Reasoning

Search
## Propagation Steps (Forward Checking)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1 2 5 6</td>
<td>8</td>
<td>1 2 3 5 6</td>
<td>1 2 3 5 6</td>
<td>1 2 3 5 6</td>
<td>1 2 3 5 6</td>
<td>1 2 3 5 6</td>
<td>1 2 3 5 6</td>
<td>1 2 3 5 6</td>
</tr>
<tr>
<td>3</td>
<td>1 4 5 6 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
<td>2 3 5 6 7 9</td>
</tr>
<tr>
<td>1</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
<td>1 4 5 6 7 9</td>
</tr>
<tr>
<td>1</td>
<td>1 4 5 6 7 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
<td>1 3 1 3 9</td>
</tr>
<tr>
<td>1</td>
<td>3 1 4</td>
<td>6</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
</tr>
<tr>
<td>1</td>
<td>3 1 4</td>
<td>6</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
</tr>
<tr>
<td>1</td>
<td>3 1 4</td>
<td>6</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
</tr>
<tr>
<td>1</td>
<td>3 1 4</td>
<td>6</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
</tr>
<tr>
<td>1</td>
<td>3 1 4</td>
<td>6</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
</tr>
<tr>
<td>1</td>
<td>3 1 4</td>
<td>6</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
<td>1 2 3 4 5 6 9</td>
</tr>
</tbody>
</table>

There is a log file at 2020-01-10 09:00:27 that is not part of the main simulation. It is located at "c:windowstemp/logfile.txt".
## Problem

### Initial Propagation (Forward Checking)

### Improved Reasoning

### Search

---

### Propagation Steps (Forward Checking)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>

---

Back to Start

Skip Animation
## Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Initial Propagation (Forward Checking)

Improved Reasoning

Search
### Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

---

**Problem**

Initial Propagation (Forward Checking)

Improved Reasoning

Search

**Search**

Propagation Steps (Forward Checking)
Propagating Steps (Forward Checking)
Propagation Steps (Forward Checking)

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Back to Start  Skip Animation
## Propagation Steps (Forward Checking)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>2</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>9</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>9</td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search
## Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th>4</th>
<th>1 2 5 6</th>
<th>8</th>
<th>2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3</th>
<th>5 6 7 9 7 9 7 9 7 9 7 9 7 9 7 9 7 9 7 9 7 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 2 5 6 4 5 6 1 4 5 7 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 6 9 4 5 9 4 5 9 7 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 9 6 9 5 9 6 9 5 9 7 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 7 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 7 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
<tr>
<td>1 4 5 8 9</td>
<td>1 7 2 3 4 5 6 9 2 3 4 5 9</td>
<td>2 3 4 5 6 8 3 4 5 6 8 9</td>
<td>2 3 4 5 6 8 9</td>
<td></td>
</tr>
</tbody>
</table>
Propagation Steps (Forward Checking)
**Problem**

Initial Propagation (Forward Checking)

Improved Reasoning

Search

---

**Propagation Steps (Forward Checking)**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>7</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

---

Helmut Simonis

Introduction to CP
Propagation Steps (Forward Checking)

Initial Propagation (Forward Checking)
Improved Reasoning
Search
**Problem**

1. **Initial Propagation (Forward Checking)**
2. **Improved Reasoning**
3. **Search**

---

### Propagation Steps (Forward Checking)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

---

**Back to Start**  **Skip Animation**

---

Helmut Simonis  Introduction to CP  174
Propagation Steps (Forward Checking)
## Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

**Problem**

Initial Propagation (Forward Checking)

Improved Reasoning

Search

**Propagation Steps (Forward Checking)**

### Step 1

1. Place the numbers 1, 2, 3, 4, 5, 6, 7, 8, and 9 in the grid.

### Step 2

2. Propagate the values using forward checking.

### Step 3

3. Continue the propagation process until all values are determined.

---

Back to Start

Skip Animation

Helmut Simonis

Introduction to CP
### Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

---

**Initial Propagation (Forward Checking)**

**Improved Reasoning**

**Search**

---

**Back to Start**

---

**Skip Animation**

---

Helmut Simonis

Introduction to CP
Propagation Steps (Forward Checking)
Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search

Propagation Steps (Forward Checking)

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

Back to Start
Skip Animation
### Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The propagation steps for solving a Sudoku puzzle using forward checking are shown above. Each cell represents a digit that can be placed in the corresponding position of the Sudoku grid. The numbers indicate the steps taken to fill in the grid, with each step showing the updated state of the puzzle. The goal is to fill each row, column, and 3x3 subgrid with the digits 1 through 9 without repetition, following the rules of Sudoku.
## Propagation Steps (Forward Checking)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>5</td>
<td>9</td>
<td></td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Problem
- Initial Propagation (Forward Checking)
- Improved Reasoning
- Search

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>5</td>
<td>9</td>
<td></td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Initial Propagation (Forward Checking)

### Improved Reasoning

### Search

#### Propagation Steps (Forward Checking)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Back to Start**

**Helmut Simonis**

**Introduction to CP**

182
After Setup (Forward Checking)
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 alldifferent 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

\[
\begin{align*}
X &: 1..2, \\
Y &: 1..2, \\
Z &: 1..3
\end{align*}
\]
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

- X 1
- Y 2
- Z 3
A Simpler Example

No other variable can use that interval

X ———— 1
Y ———— 2
Z ———— 3
A Simpler Example

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

- $X$ is assigned 1
- $Y$ is assigned 2
- $Z$ is assigned 3
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
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
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

\[
\begin{align*}
X &:: [1, 3], \\
Y &:: [1, 3], \\
Z &:: 1..3
\end{align*}
\]
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

X 1
Y 2
Z 3
Another Simple Example

Solution 1: assignment in blue

X 1
Y 2
Z 3
Another Simple Example

Solution 2: assignment in green

X 1
Y 2
Z 3
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)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>23</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
<td></td>
<td>123</td>
</tr>
</tbody>
</table>
Propagation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)
### Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>
Propagation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)

1. Propagation Steps (Domain Consistency)

2. Improved Reasoning

3. Search

4. Domain Consistency

5. Comparison

6. Initial Propagation (Forward Checking)

7. Problem
Propagation Steps (Domain Consistency)
### Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th>4</th>
<th>1 2 5 6</th>
<th>8</th>
<th>2 3 5 6 7 9</th>
<th>1 2 3 5 6 7 9 7 9</th>
<th>1 2 3 5 6 7 9 7 9</th>
<th>1 2 3 5 6 7 9 7 9</th>
<th>1 2 3 5 6 7 9 7 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 3 5 6 7 9</td>
<td>1 7</td>
<td>2 3 4 5 6 8 9 8 9</td>
<td>2 3 4 5 6 8 9 8 9</td>
<td>2 3 4 5 6 8 9 8 9</td>
<td>2 3 4 5 6 8 9 8 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 4 5 6 9 7 9</td>
<td>8</td>
<td>1 4 5 6 7 9 7 9</td>
<td>1 4 5 6 7 9 7 9</td>
<td>1 4 5 6 7 9 7 9</td>
<td>1 4 5 6 7 9 7 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 3 1 4 7 9</td>
<td>6</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 3 5 6 7 9</td>
<td>8</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 3 1 2 4 5 6 7 9</td>
<td>5</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 3 1 2 4 5 6 7 9</td>
<td>4</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 3 1 2 4 5 6 7 9</td>
<td>6</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 3 1 2 4 5 6 7 9</td>
<td>4</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td>1 2 3 4 5 6 9 7 9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Problem
- Initial Propagation (Forward Checking)
- Improved Reasoning
- Search

### Domain Consistency
- Comparison

---

Back to Start  Skip Animation

Helmut Simonis  Introduction to CP
### Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

#### Initial Propagation (Forward Checking)

- Improved Reasoning
- Search

#### Domain Consistency

Comparison
### Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Problem**

- Initial Propagation (Forward Checking)
- Improved Reasoning
- Search

**Domain Consistency**

**Comparison**

**Propagation Steps**

```
4 1 2 5 6
8 2 3 5 7 9
3 5 6 4 8 9
1 4 5 9 7 9
7 1 4 5 6 9
8 1 4 5 6 7
2 3 1 2 3 6
1 2 3 1 2 8
```
## Propagation Steps (Domain Consistency)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>8</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Back to Start

### Skip Animation

**Helmut Simonis**

*Introduction to CP*
Propagation Steps (Domain Consistency)
## Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

**Initial Propagation (Forward Checking)**

**Improved Reasoning**

**Search**

**Domain Consistency**

**Comparison**
## Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

*Propagation Steps (Domain Consistency)*

- **Initial Propagation (Forward Checking)**
- **Improved Reasoning**
- **Search**
- **Domain Consistency**
- **Comparison**

[Back to Start] [Skip Animation]
## Propagation Steps (Domain Consistency)

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
<th>Column 5</th>
<th>Column 6</th>
<th>Column 7</th>
<th>Column 8</th>
<th>Column 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1 2 5 6</td>
<td>8</td>
<td>2 3 5 6 9</td>
<td>2 3 5 6 9</td>
<td>1 2 3 5 6 9</td>
<td>1 2 3 5 6 9</td>
<td>1 2 3 5 6 9</td>
<td>1 2 3 5 6 9</td>
</tr>
<tr>
<td>3 2</td>
<td>4 5 6 8 9</td>
<td>1 5 6 9 8</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
</tr>
<tr>
<td>1 5 6 9 8</td>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>2 3 1 4 5 9</td>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
</tr>
<tr>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
</tr>
<tr>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
</tr>
<tr>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
</tr>
<tr>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
</tr>
<tr>
<td>1 4 5 9 7</td>
<td>1 4 5 9 7</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
<td>2 3 4 5 9</td>
</tr>
</tbody>
</table>

*Back to Start*  *Skip Animation*
### Propagation Steps (Domain Consistency)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>4</strong></td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>9</td>
<td>8</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

---

[Back to Start] [Skip Animation]
Propagation Steps (Domain Consistency)
### Domain Consistency

#### Initial Propagation (Forward Checking)

#### Improved Reasoning

#### Search

## Propagation Steps (Domain Consistency)

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>7</td>
<td>9</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>8</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>3</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>6</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>7</td>
<td>9</td>
<td>4</td>
</tr>
</tbody>
</table>
### Propagation Steps (Domain Consistency)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

*Back to Start*
Propagation Steps (Domain Consistency)
Propogation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)
Propagation Steps (Domain Consistency)
Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search

Domain Consistency
Comparison

Propagation Steps (Domain Consistency)

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

Back to Start
After Setup (Domain Consistency)
**Comparison**

<table>
<thead>
<tr>
<th>Forward Checking</th>
<th>Bounds Consistency</th>
<th>Domain Consistency</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Forward Checking" /></td>
<td><img src="image2.png" alt="Bounds Consistency" /></td>
<td><img src="image3.png" alt="Domain Consistency" /></td>
</tr>
</tbody>
</table>
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)

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>1</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>2</td>
</tr>
</tbody>
</table>

2

---

Helmut Simonis

Introduction to CP
Search Tree (Forward Checking)
Search Tree (Forward Checking)

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>1</td>
<td>7</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>9</td>
<td>3</td>
<td>8</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>5</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>9</td>
<td>2</td>
<td>7</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>5</td>
<td>8</td>
<td>9</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>8</td>
<td>3</td>
<td>2</td>
<td>9</td>
<td>7</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>1</td>
<td>9</td>
<td>7</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Initial Propagation (Forward Checking)

Improved Reasoning

Search

Helmut Simonis  Introduction to CP
Search Tree (Forward Checking)

Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>8</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>3</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>9</td>
<td>8</td>
</tr>
</tbody>
</table>

Search Tree for Initial Propagation (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)

Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Forward Checking)
Search Tree (Bounds Consistency)

2
Search Tree (Bounds Consistency)
Search Tree (Bounds Consistency)
Search Tree (Bounds Consistency)
### Search Tree (Domain Consistency)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>1</td>
<td>7</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>1</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>6</td>
<td></td>
<td></td>
<td>8</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>7</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>5</td>
<td>9</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>4</td>
<td></td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td></td>
<td>2</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>7</td>
<td>9</td>
<td></td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>8</td>
<td></td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Search Tree (Domain Consistency)
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

- 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
Problem
Initial Propagation (Forward Checking)
Improved Reasoning
Search

$$\text{diffn}(\text{Obj})$$

- $n$-dimensional objects given by origin $< x_{i1}, ..., x_{in} >$ and
  size $< d_{i1}, ..., d_{in} >$ 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 Constraint

global_cardinality(L, Values)

- Count how often certain values occur in the collection of variables \( L \)
- Algorithm: Flow, refinements
- Use: TImetabling
- 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)
- 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
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
assign \[ [X_1, X_2, \ldots, X_N] \]

s.t.

\[
\forall 1 \leq i \leq N : \quad X_i \in 1..N
\]

\[
\forall 1 \leq i < j \leq N : \quad X_i \neq X_j
\]

\[
\forall 1 \leq i < j \leq N : \quad X_i \neq X_j + i - j
\]

\[
\forall 1 \leq i < j \leq N : \quad X_i \neq X_j + j - i
\]
Outline

10 Problem
11 Program
12 Naive Search
13 Improvements
Default Strategy
Default Strategy
Default Strategy
Default Strategy

Problem
Program
Naive Search
Improvements
Default Strategy

Problem
Program
Naive Search
Improvements

Back to Start  Skip Animation

Helmut Simonis  Introduction to CP
Default Strategy
Default Strategy
Default Strategy
Default Strategy

[Diagram showing a tree structure with nodes labeled 1, 2, 3, 4, 5, 6, 7, and 8, and a grid representation of a problem space with some cells marked in red and green.]
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy

Problem
Program
Naive Search
Improvements

Back to Start  Skip Animation

Helmut Simonis  Introduction to CP
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
Default Strategy
First Solution
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 Stategy, Problem Sizes 4-100

Time [s] vs Problem Size for "naive/all.txt"
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)

Helmut Simonis
Introduction to CP
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
FirstFail Strategy (Size 16)
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)
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)
Problem
Program
Naive Search
Improvements

Dynamic Variable Choice
Improved Heuristics
Making Search More Stable

Start from Middle (Size 16)
Start from Middle (Size 16)
Start from Middle (Size 16)
Start from Middle (Size 16)
Problem
Program
Naive Search
Improvements

Dynamic Variable Choice
Improved Heuristics
Making Search More Stable

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)
Problem
Program
Naive Search
Improvements

Dynamic Variable Choice
Improved Heuristics
Making Search More Stable

Start from Middle (Size 16)
Start from Middle (Size 16)
Comparing Solutions

- Naive Search
- Improved Heuristics
- Making Search More Stable

Helmut Simonis
Introduction to CP
Comparing Solutions

Naive

First Fail

Middle

Again, solutions are different!
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

Time [s] vs Problem Size

"credit/all.txt"
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

1. Problem
2. Human
3. User Model
4. Reformulation
5. Implementation Model
6. Constraint Solver/Search
7. Solution
What is CP actually used for?

http://hsimonis.wordpress.com

Constraint Applications Blog by Helmut Simonis

CP Conference Application Track: Call for Papers
Posted on March 22, 2011 by simonis

Rate This

If you are reading this blog, then you probably are a prime candidate for the following call for papers:

http://www.cimi.unige.it/cp2011/cfa.html

Email Subscription
Enter your email address to subscribe to this blog and receive notifications of new posts by email.

Sign me up