# Search and Planning

MEIC @ IST

Artificial Intelligence

2023/24

# Welcome ☺

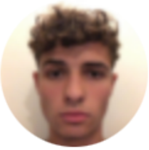
- First year MSc?
- A 7-week journey!
- Learn and have fun

# Course Dynamics

## Faculty

Inês Lynce Senior Lecturer
ines.lynce@tecnico.ulisboa.pt

Daniela Carvalho
daniela.carvalho@tecnico.ulisboa.pt



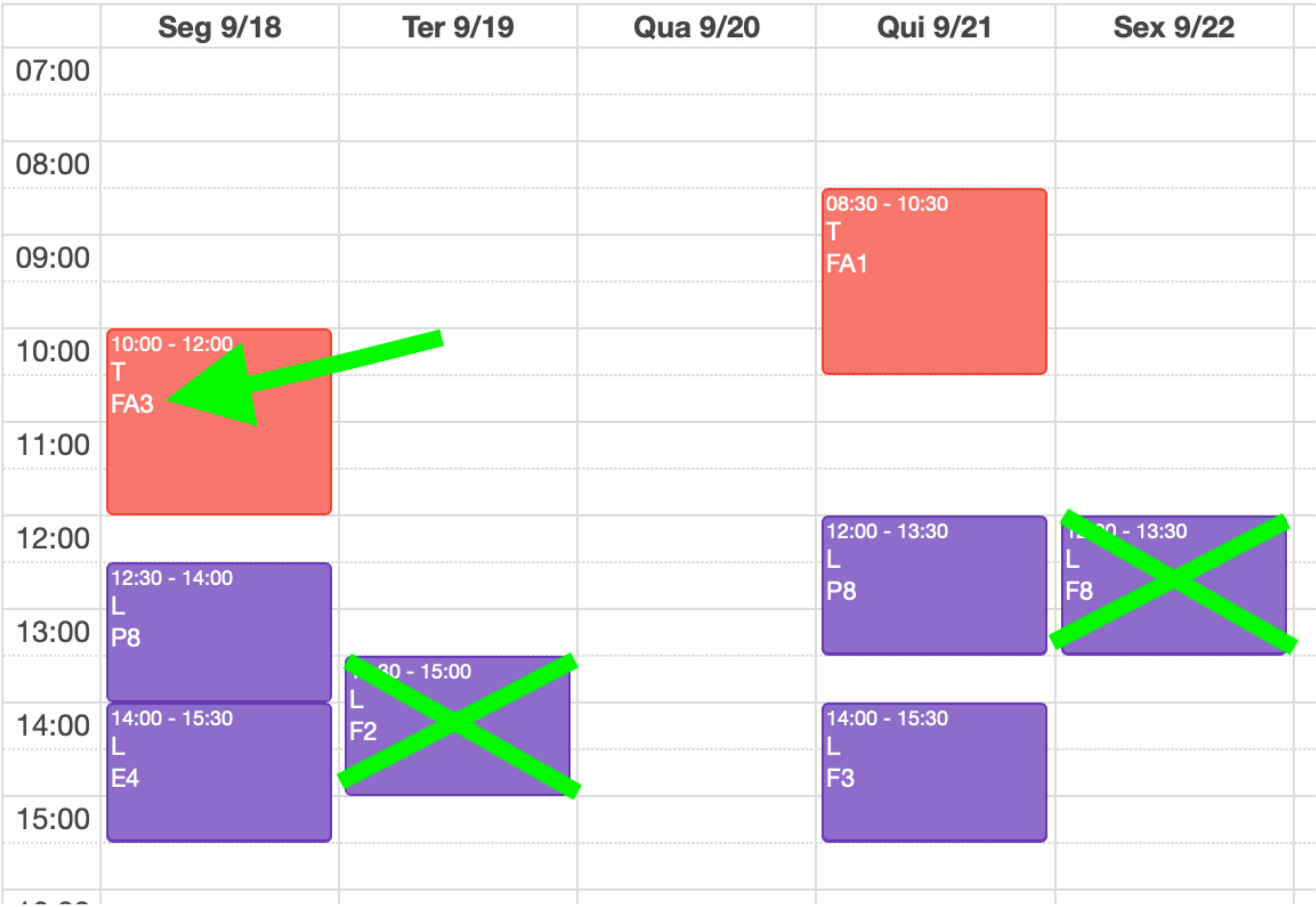
David de Almeida Pissarra
david.pissarra@tecnico.ulisboa.pt

Guilherme Henriques Bandeira Costa
guilherme.bandeira@tecnico.ulisboa.pt

# Course Dynamics: schedule

Practical shifts start on Thursday

- Read minizinc tutorial
- Take laptop if you can



# Course Dynamics: office hours

- To start on the 3rd week
- Zoom or in person

# Course Dynamics: evaluation

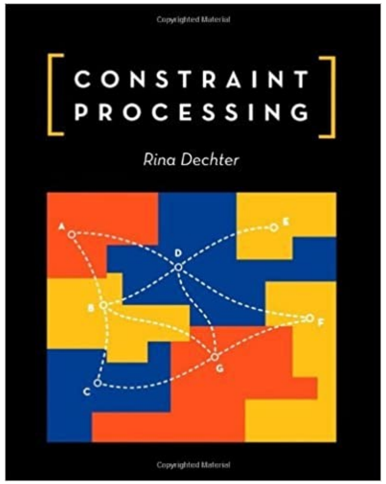
- 50% Project (due Oct 20) >= 9.5
  - 1 or 2 students; enrollment due until Sep 22
  - <u>Students do not have to be enrolled in the same practical shift</u>
- 50% Exam (Nov 6) >= 8

  (Repeat exam - optional – Jan 30)

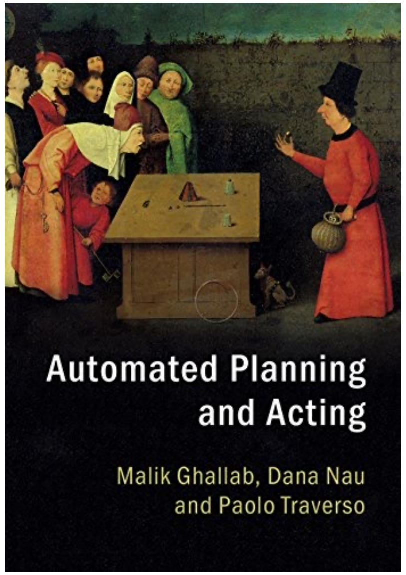
# Course Dynamics: project

- Project published: today!
- Fenix group registration due: 22/09 (opens today)
- Project due: 20/10

# Course Material




- Bibliography
  - Constraint Processing: Rina Dechter 2003 Elsevier Morgan Kaufmann
  - *Principles of Constraint Programming: Krzysztof Apt 2003 Cambridge University Press*
  - Automated Planning and Acting: Malik Ghallab, Dana Nau and Paolo Traverso 2016 Cambridge University Press
  - *Automated Planning – theory and practice: Malik Ghallab, Dana Nau and Paolo Traverso 2004 Elsevier*
- Slides of the lectures
- Exercises to be <u>prepared at home</u> and jointly discussed in the class
- Abundant material on the web!!!



# What is Search and Planning about???

You have
3 minutes!

# Dictionary definitions


- Search [verb]: try to find something by looking or otherwise seeking carefully and thoroughly

- Plan [noun]: a detailed proposal for doing or achieving something

# Search: in the context of AI…

- Search algorithms: related with problem solving
  - **Problem-Solving Methods in Artificial Intelligence by Nils Nilson** (1971)

    - Nils John Nilsson was an American computer scientist (d.2019). He was one of the founding researchers in the discipline of artificial intelligence. He was the first Kumagai Professor of Engineering in computer science at Stanford University from 1991 until his retirement.
    - https://stacks.stanford.edu/file/druid:xw061vq8842/xw061vq8842.pdf
  - Traditional problems (e.g. 8-queen, 15-puzzle, TSP) well studied also in other domains
    - Most often in Operations Research
  - ***Search*** *in AI is the process of navigating from a* ***starting state*** *to a* ***goal state*** *by* ***transitioning*** *through* ***intermediate states*** **(https://towardsdatascience.com/ai-search-algorithms-every-data-scientist-should-know-ed0968a43a7a)**

# Planning: in the context of AI…

- AI Planning is **a field of Artificial Intelligence which explores the process of using autonomous techniques to solve planning and scheduling problems**. A planning problem is one in which we have some initial starting state, which we wish to transform into a desired goal state through the application of a set of actions.
  - https://planning.wiki/guide/whatis/aip#:~:text=AI%20Planning%20is%20a%20field,of%20a%20set%20of%20actions.

# Search vs Planning

- Planning is the process of computing several steps of a problem-solving procedure before executing any of them
- This problem can be solved by search
- The main difference between search and planning is the representation of states

| | Search | Planning |
|---|---|---|
| **States** | data structures | Logical sentences |
| **Actions** | code | Preconditions/outcomes |
| **Goal** | code | Logical sentence (conjunction) |
| **Plan** | Sequence from $S_0$ | Constraints on actions |

- https://intellipaat.com/community/2632/what-is-the-difference-between-search-and-planning

# Research Communities

- Conferences
  - CP International Conference on Principles and Practice of Constraint Programming
  - ICAPS International Conference on Automated Planning and Scheduling
  - And others… e.g. IJCAI, AAAI, ECAI

- Journals
  - Constraints, an International Journal
  - Journal of Scheduling
  - And others...

    e.g. Artificial Intelligence Journal, Journal of Artificial Intelligence Research

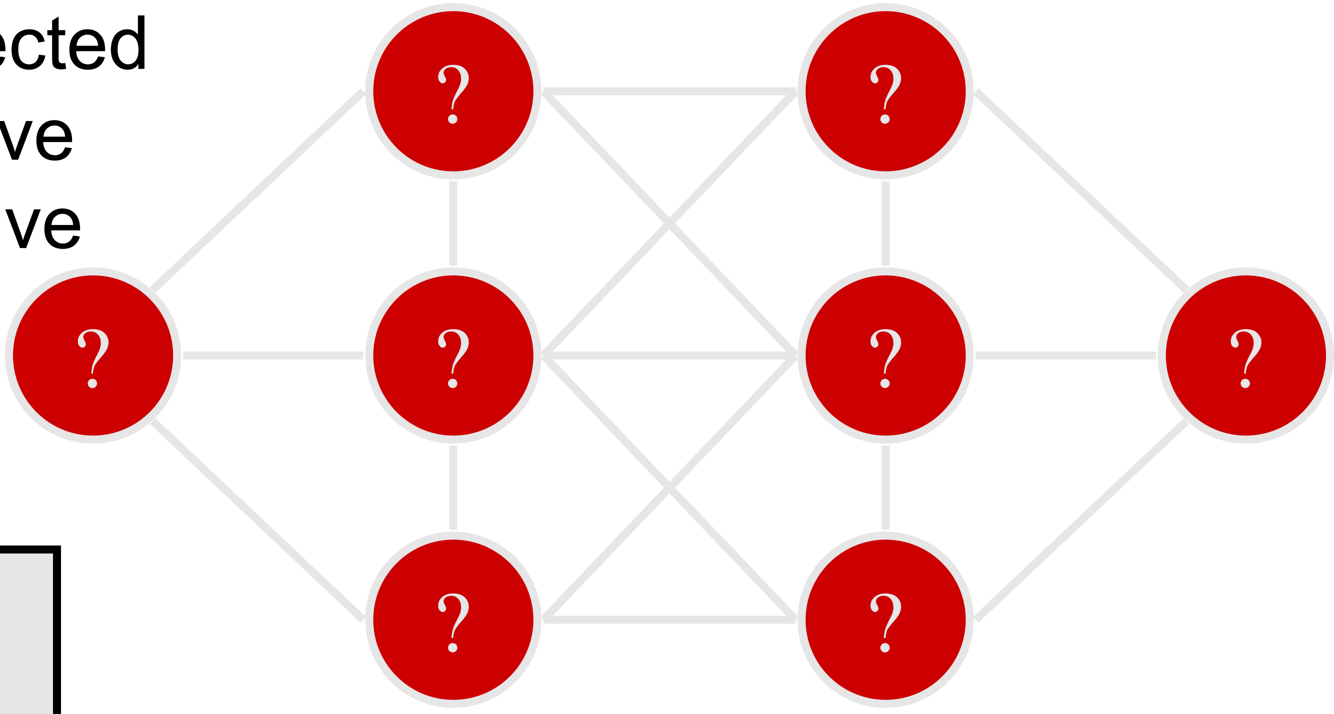
# Problem Solving with SEARCH

- In this course, we encode search problems as Constraint Satisfaction Problems (CSPs)
  - Ever heard about CSPs in the past?
- Next slides borrowed from Patrick Prosser (UGlasgow)
  - with help from Toby Walsh, Chris Beck, Barbara Smith, Peter van Beek, Edward Tsang, ...

# A Puzzle

- Place numbers 1 through 8 on nodes
  - Each number appears exactly once

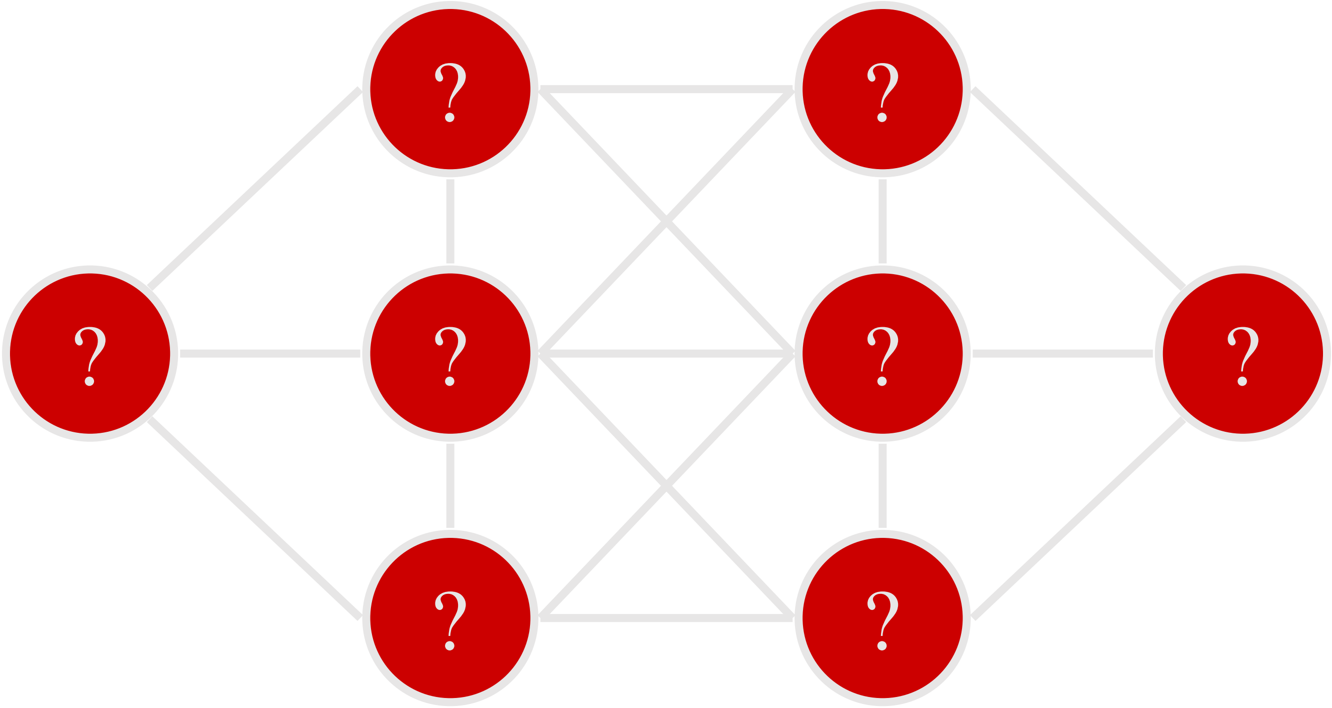
– No connected nodes have consecutive numbers



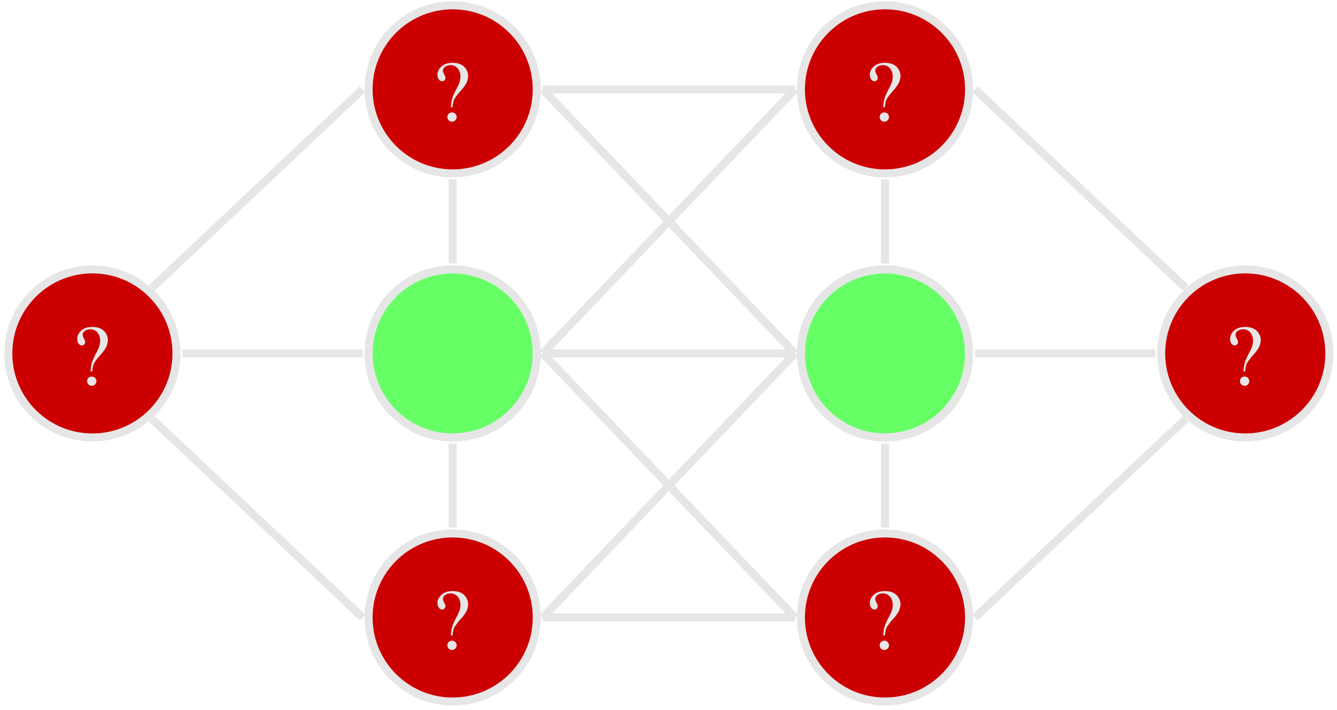

You have
8 minutes!

# Heuristic Search

Which nodes are hardest to number?

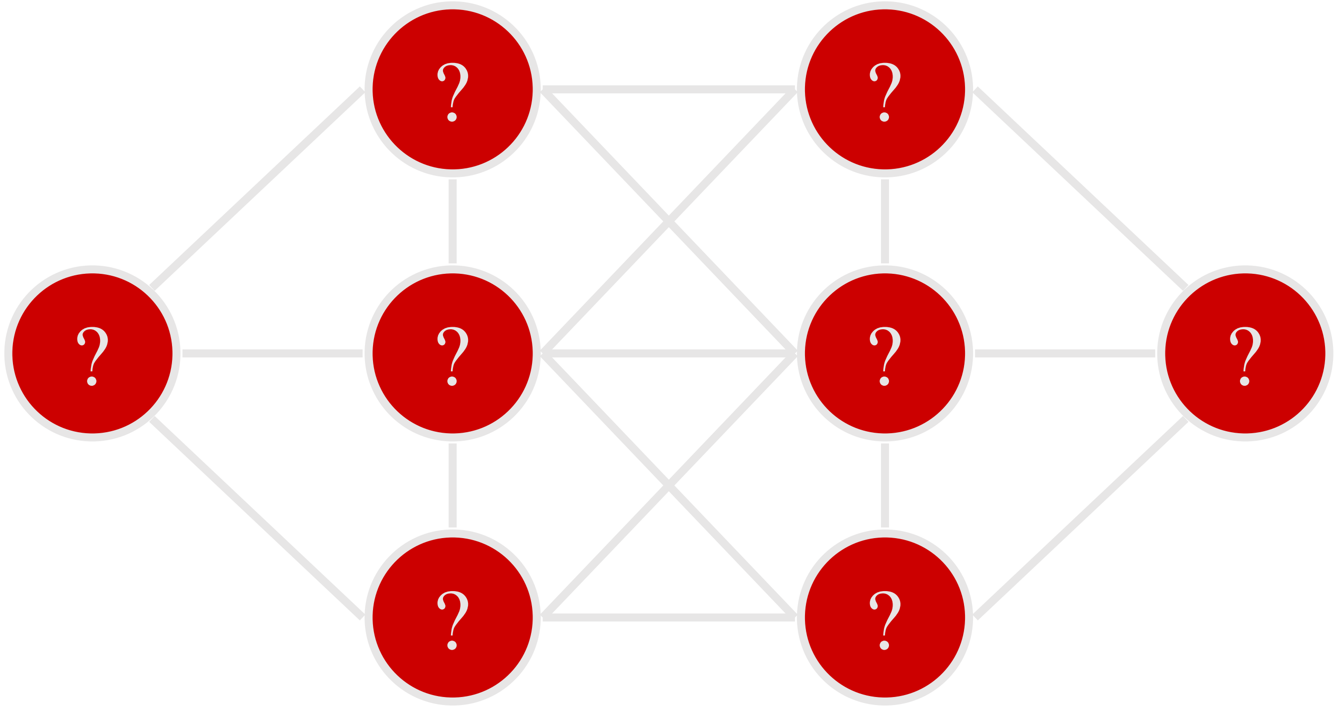


# Heuristic Search



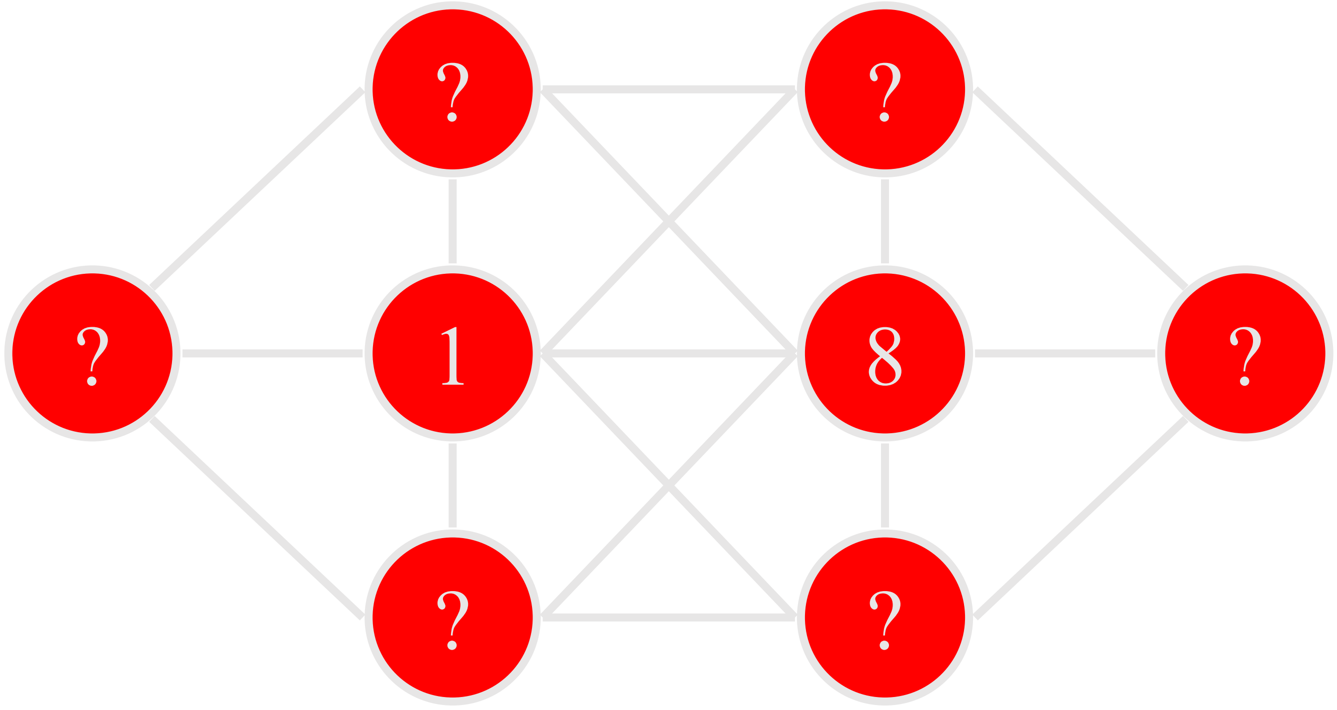
# Heuristic Search

Which are the least constraining values to use?



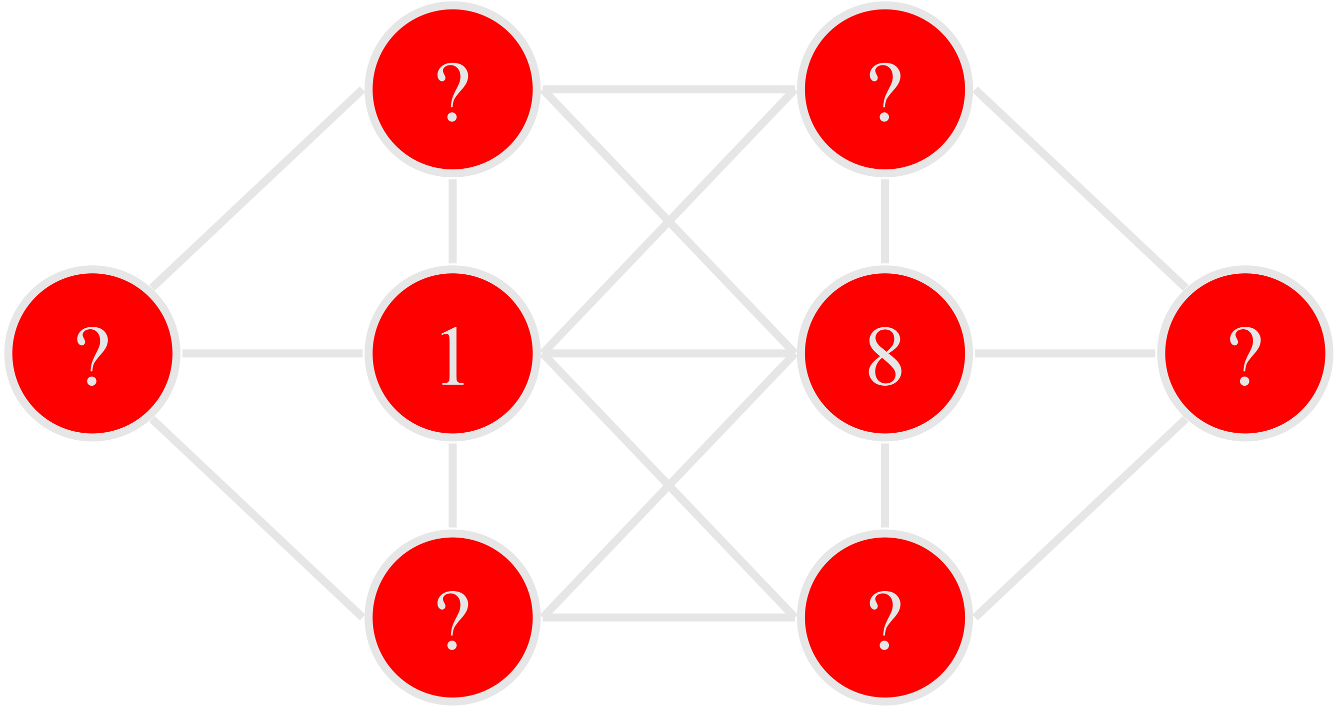
# Heuristic Search

Values 1 and 8



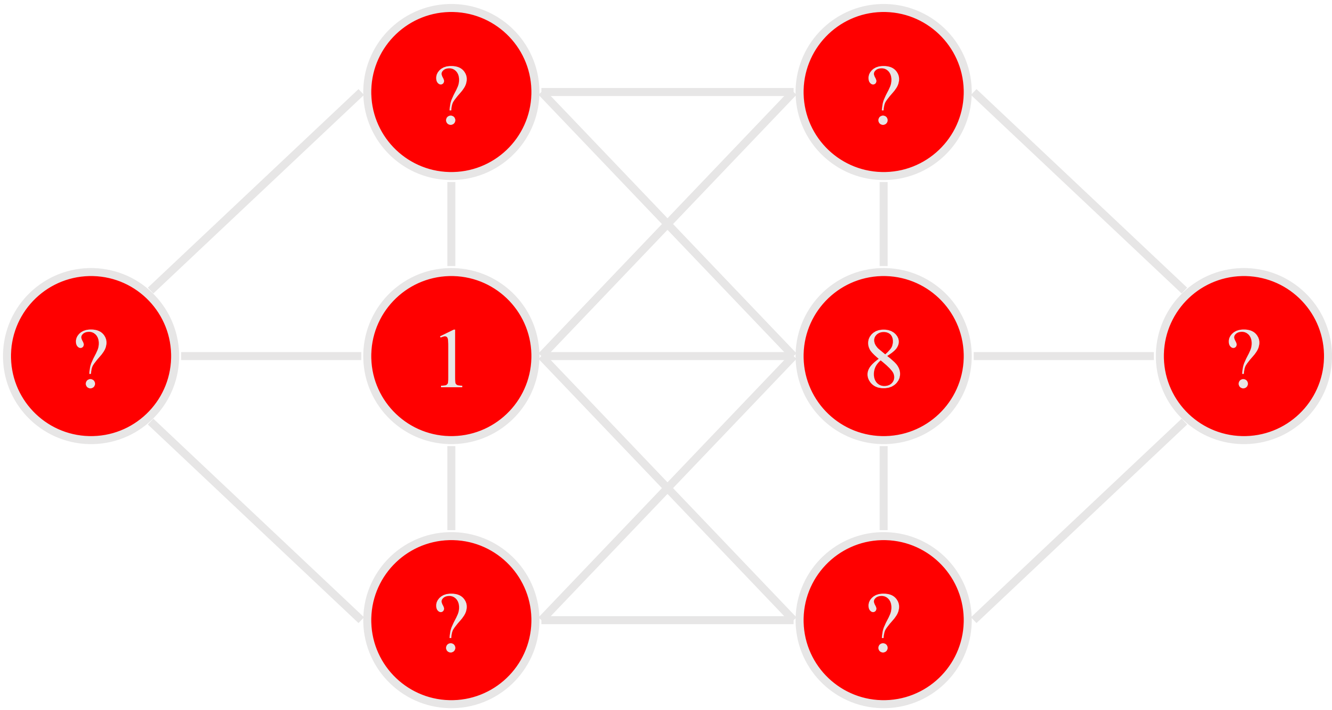
# Heuristic Search

Values 1 and 8



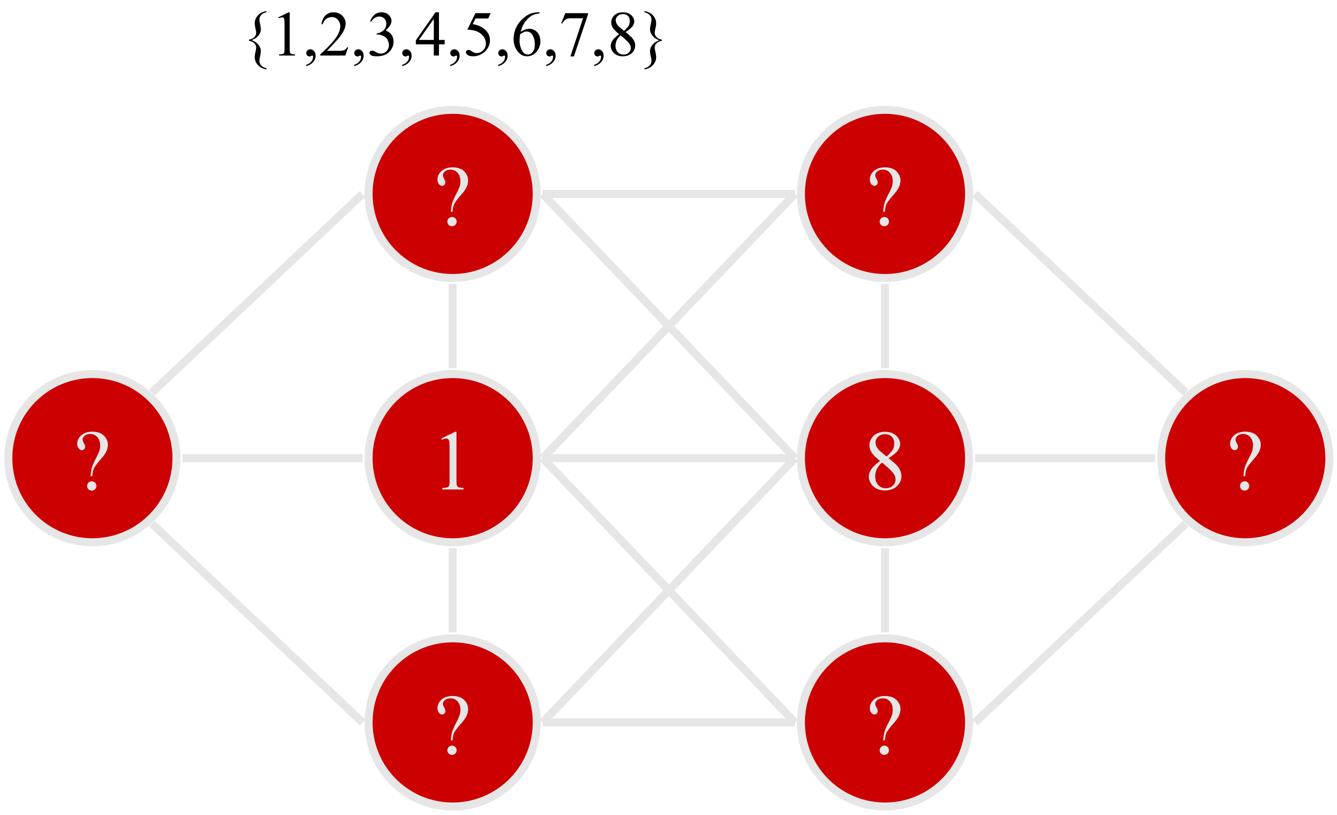

Symmetry means we don't need to consider: 8 1

# Inference/propagation

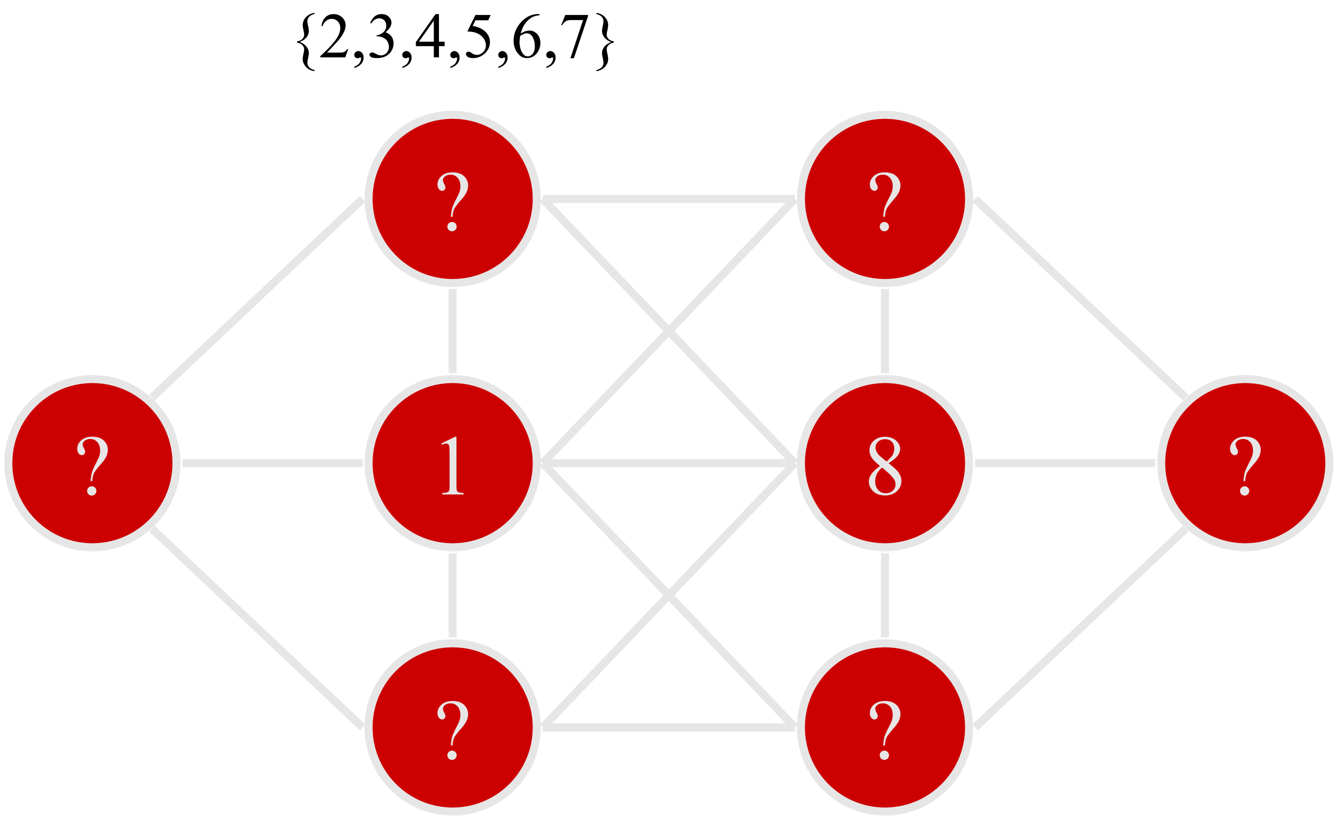



We can now eliminate many values for other nodes

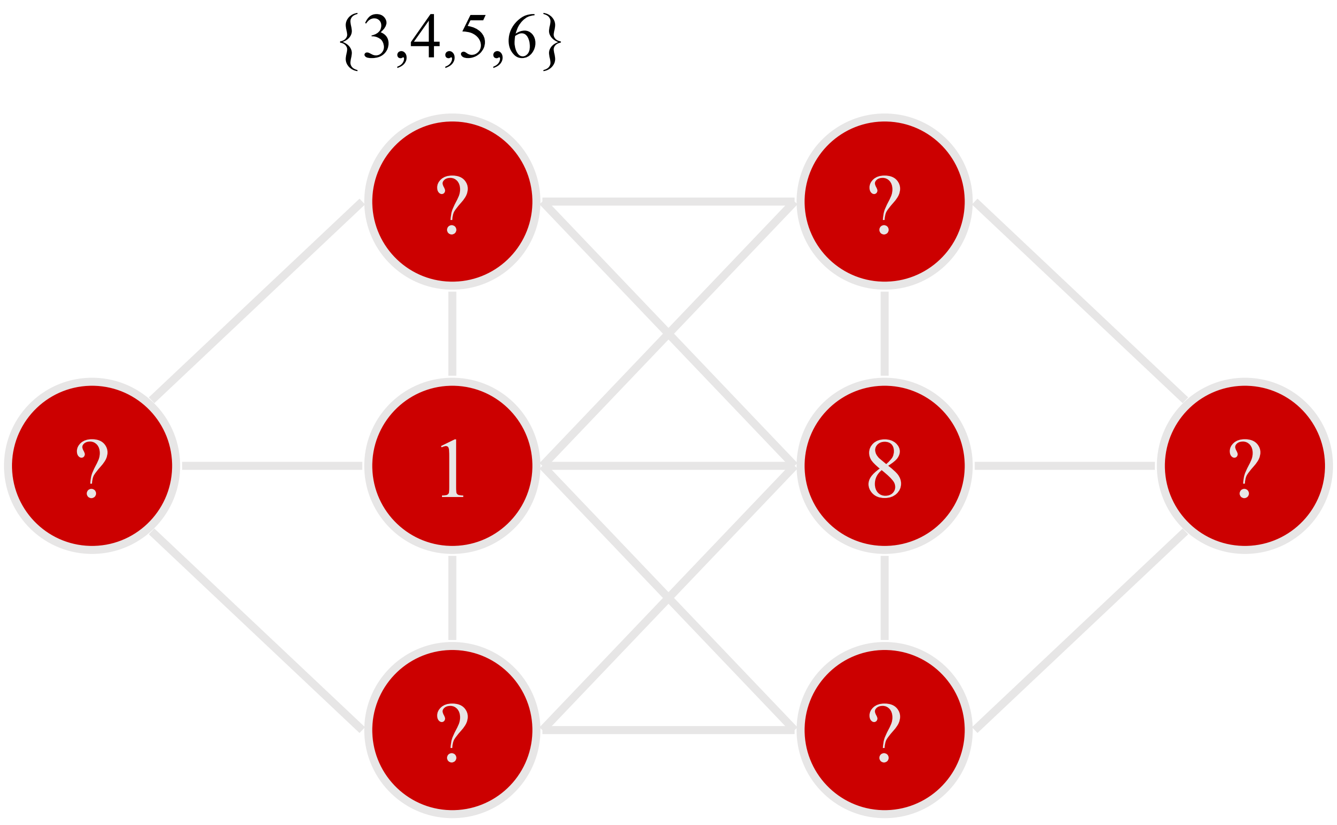
# Inference/propagation



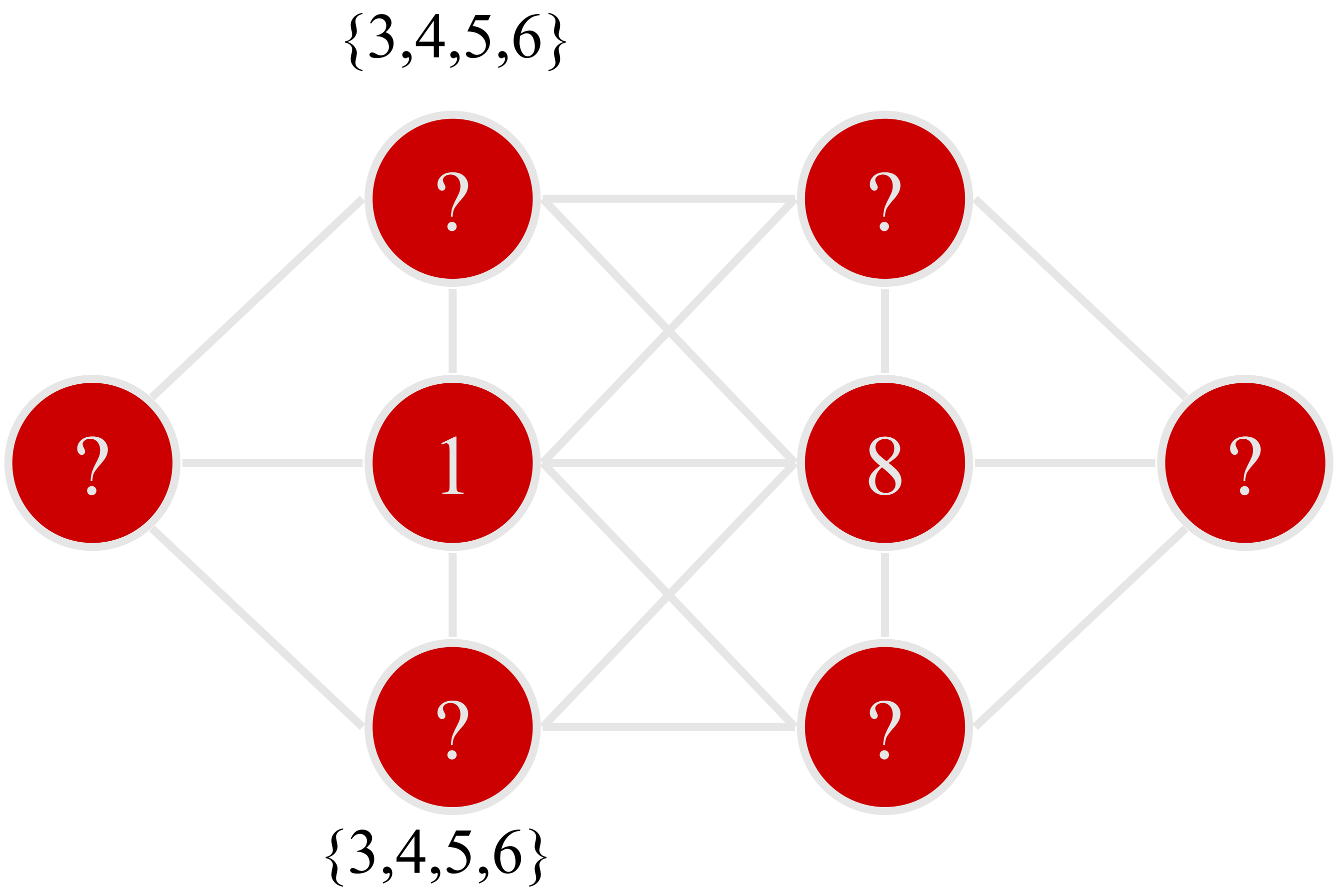
# Inference/propagation



# Inference/propagation

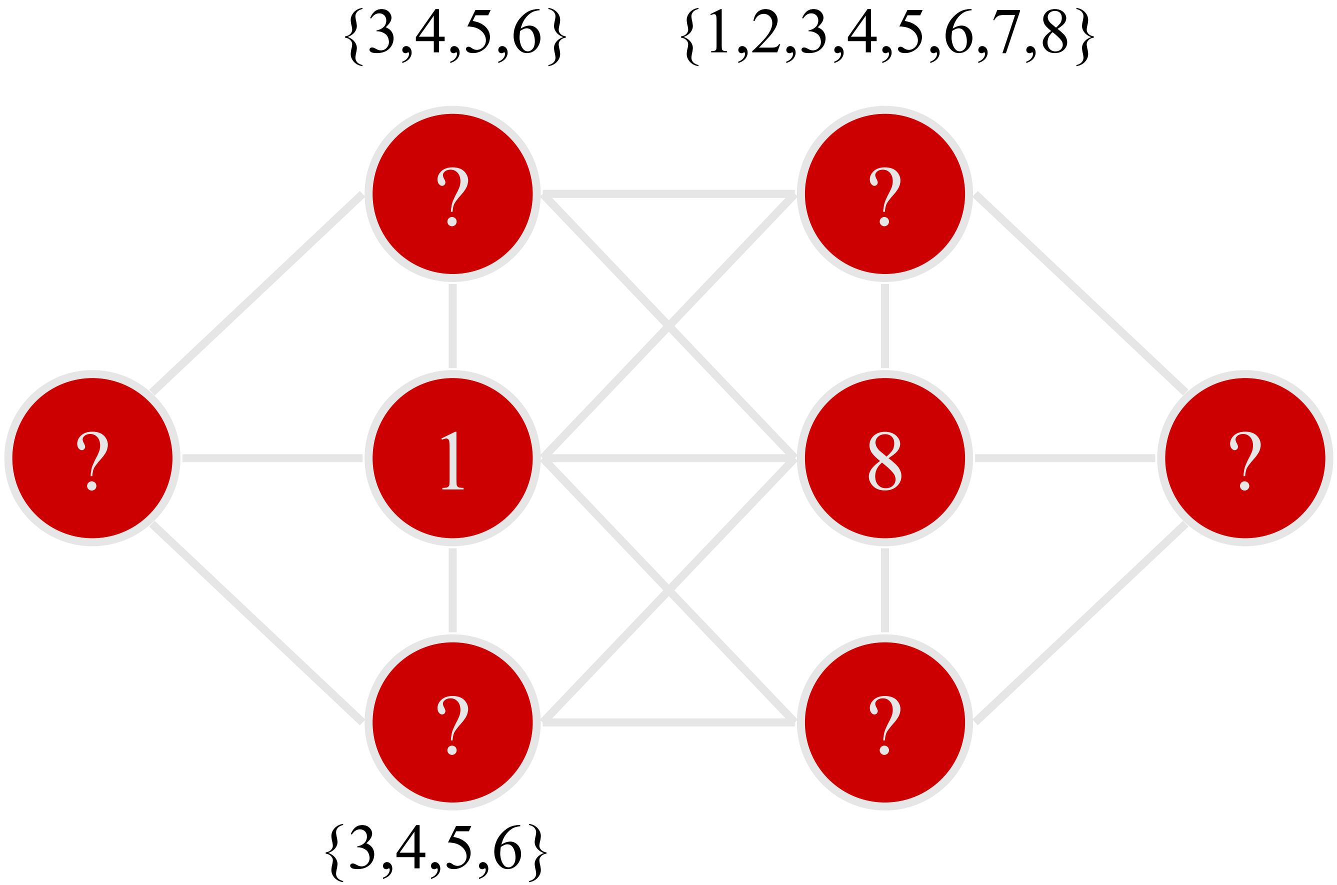


# Inference/propagation

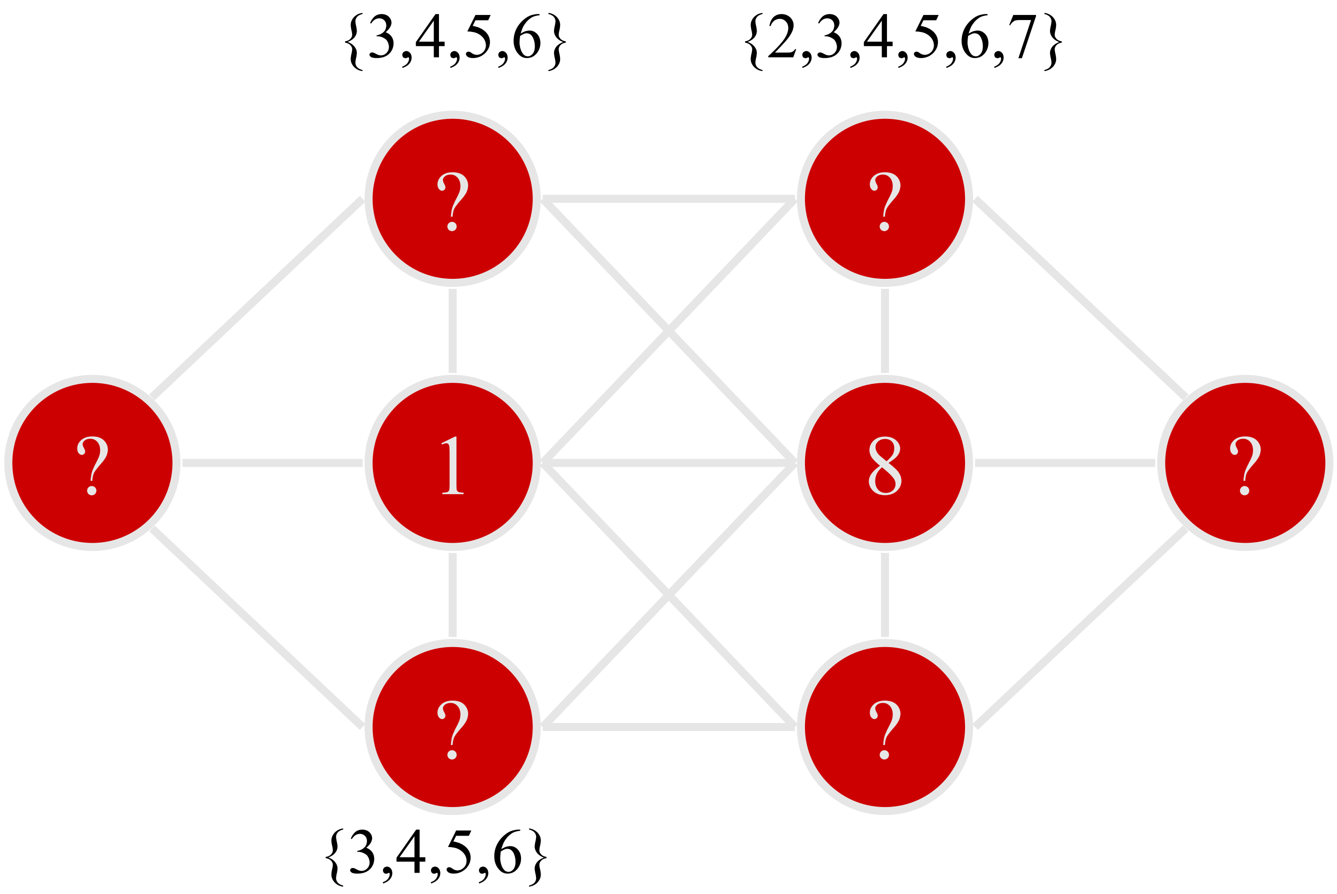



By symmetry

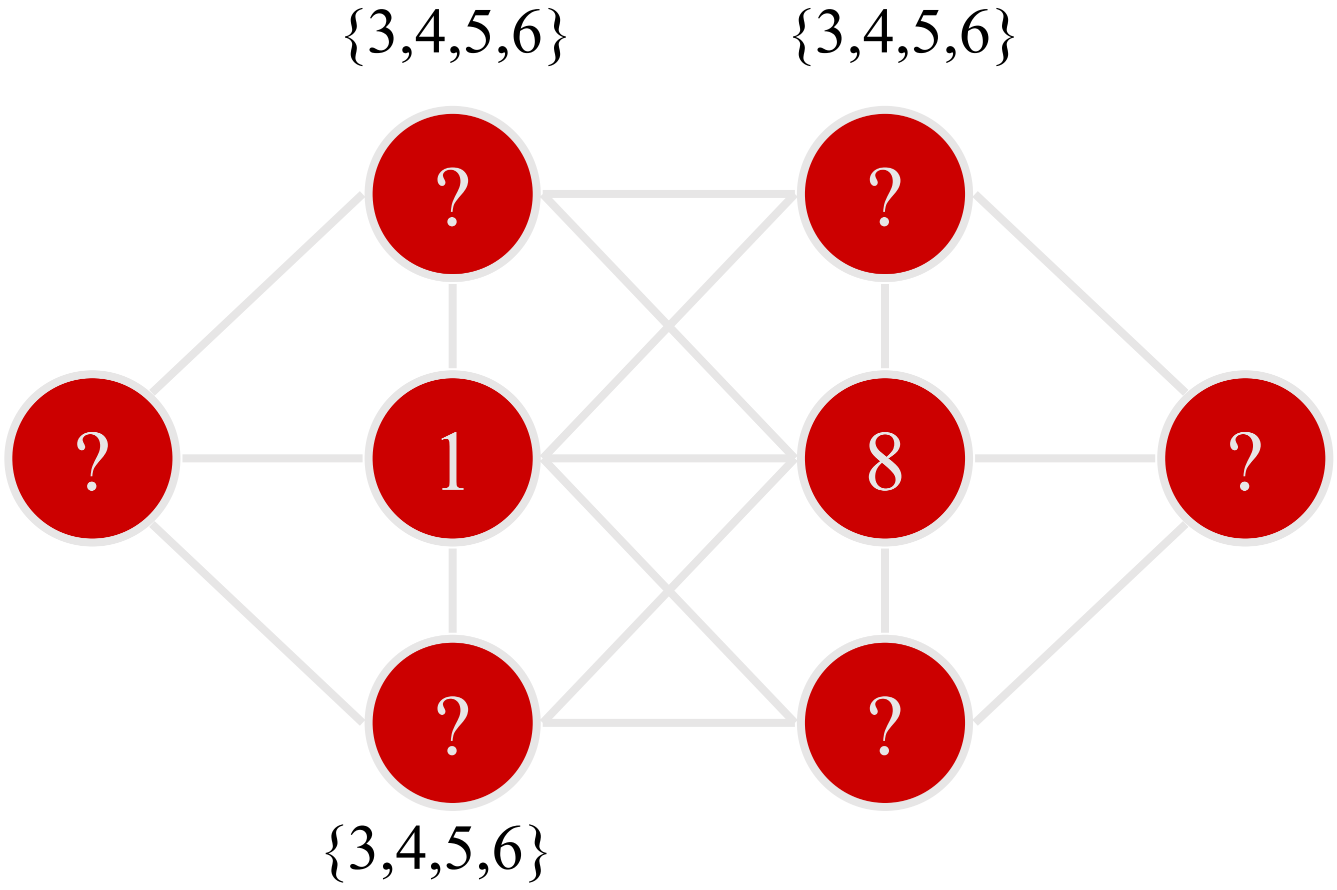
# Inference/propagation



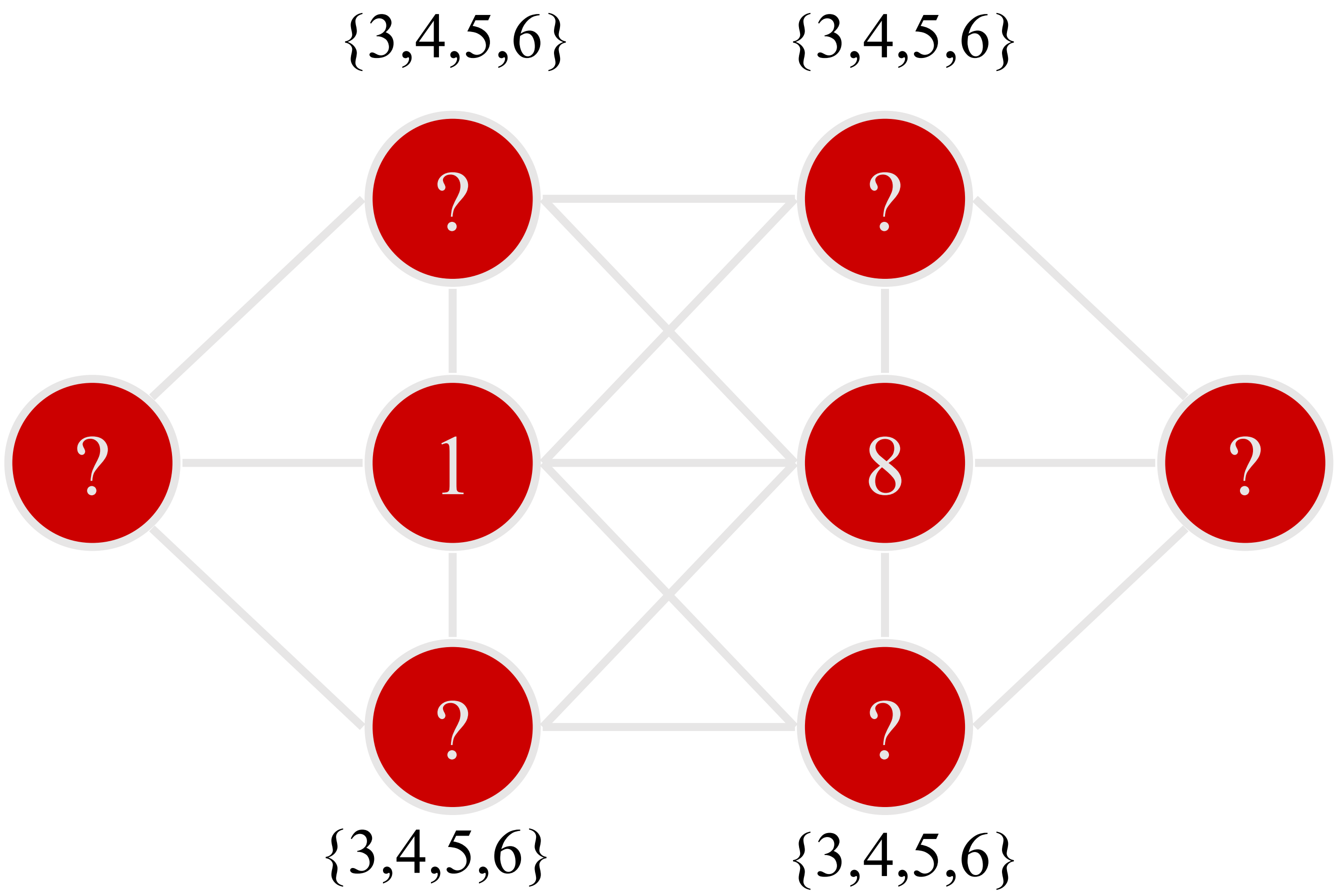
# Inference/propagation



# Inference/propagation

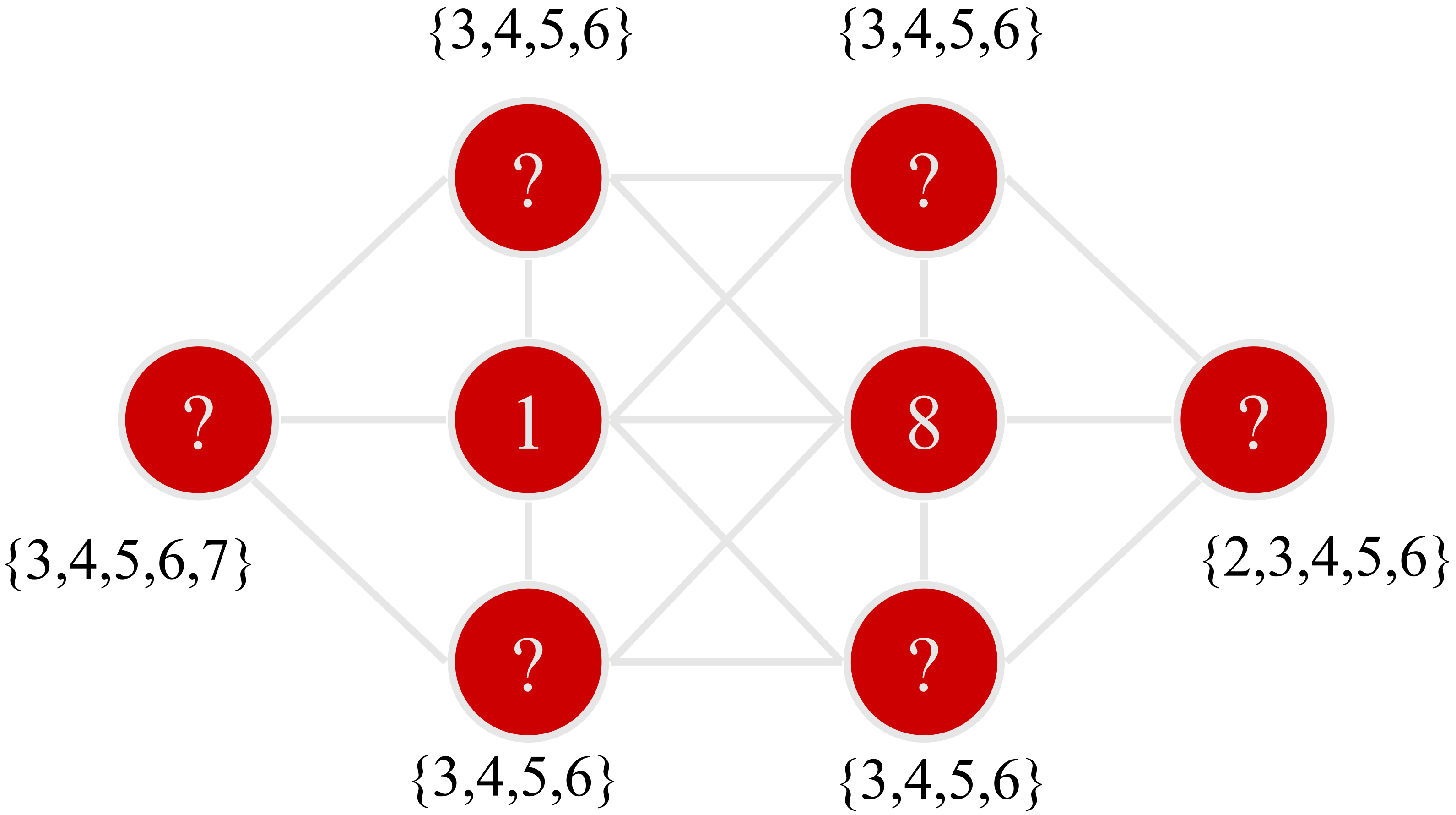


# Inference/propagation

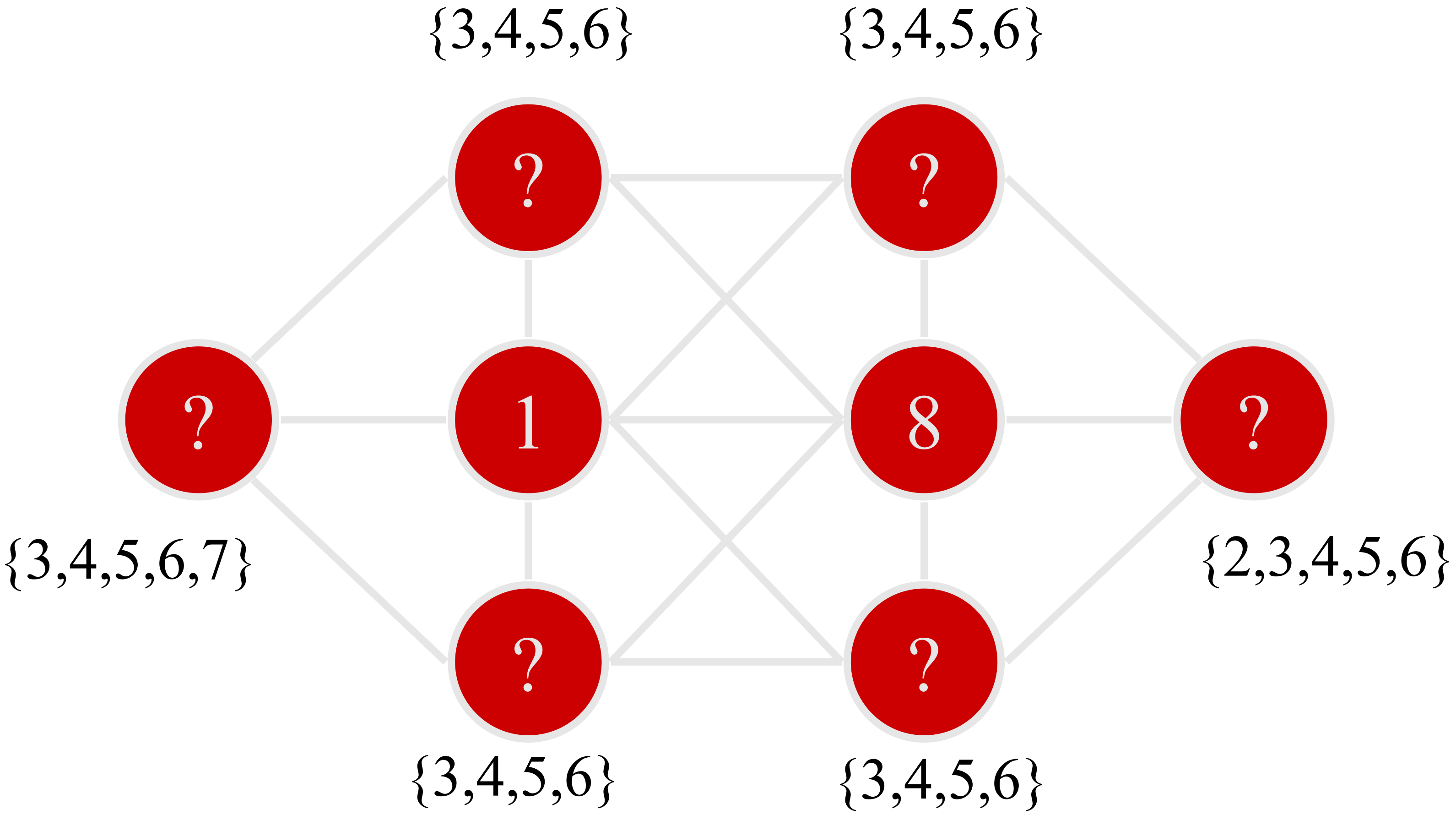



By symmetry

# Inference/propagation

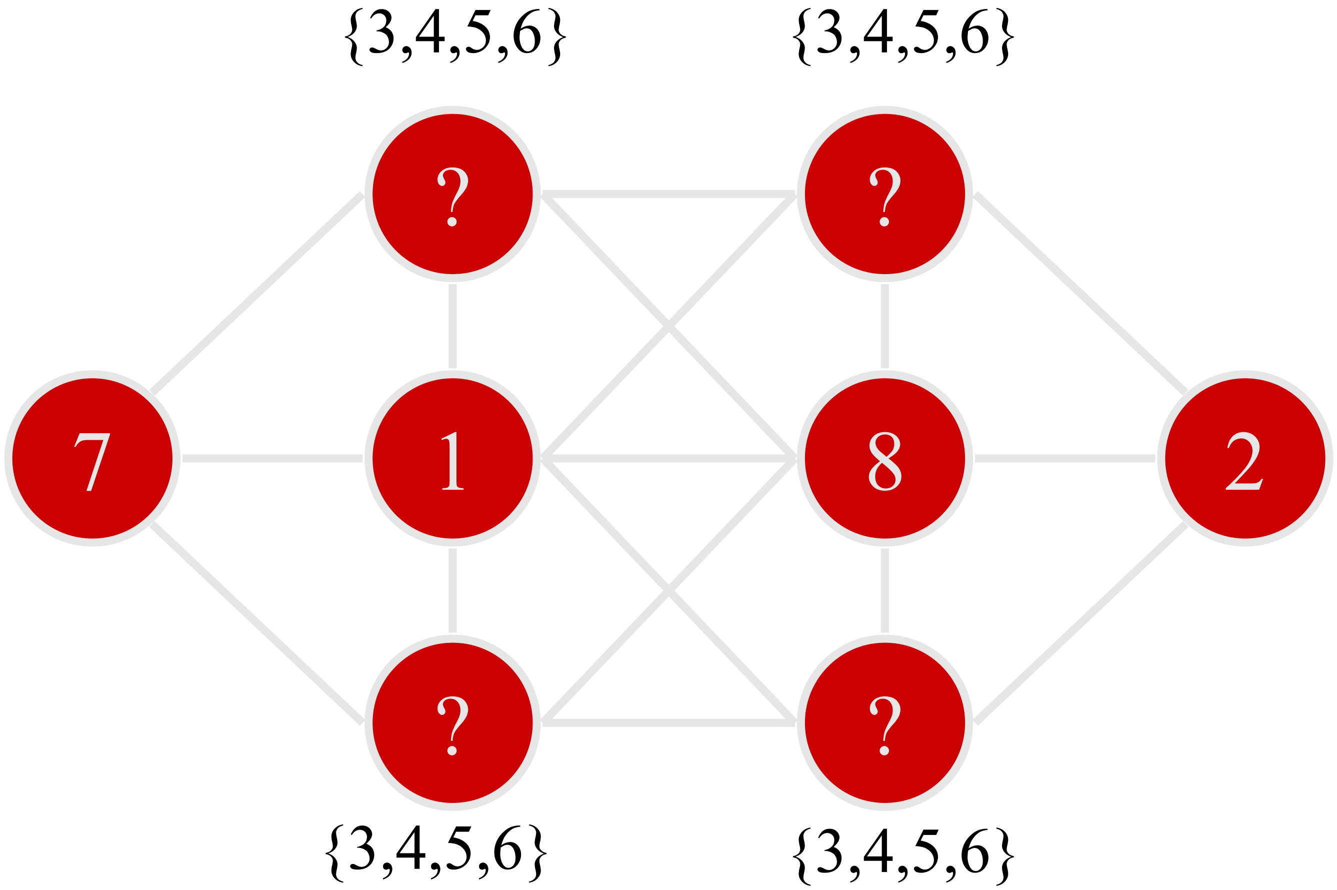


# Inference/propagation



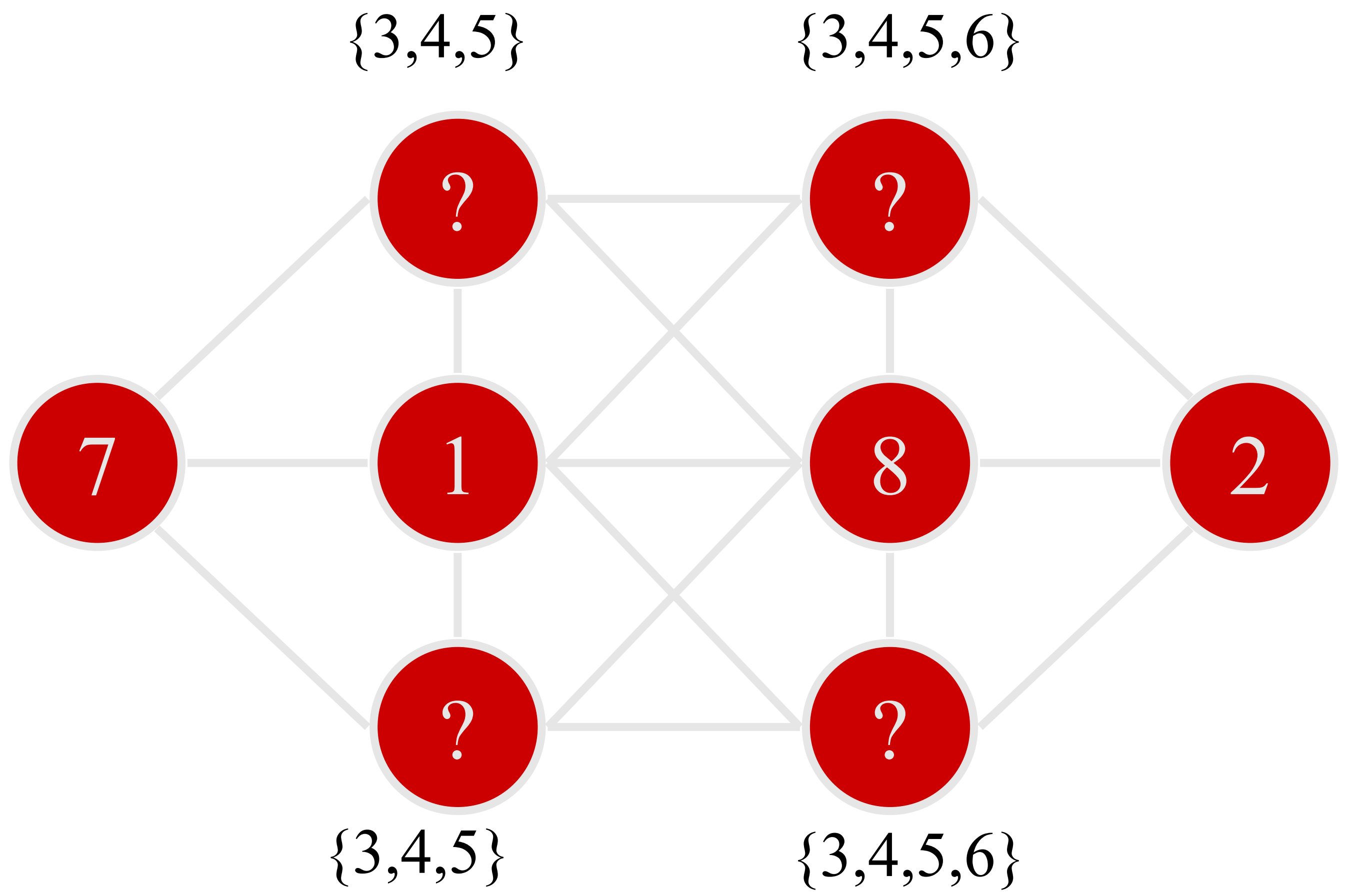

Value 2 and 7 are left in just one variable domain each

# Inference/propagation



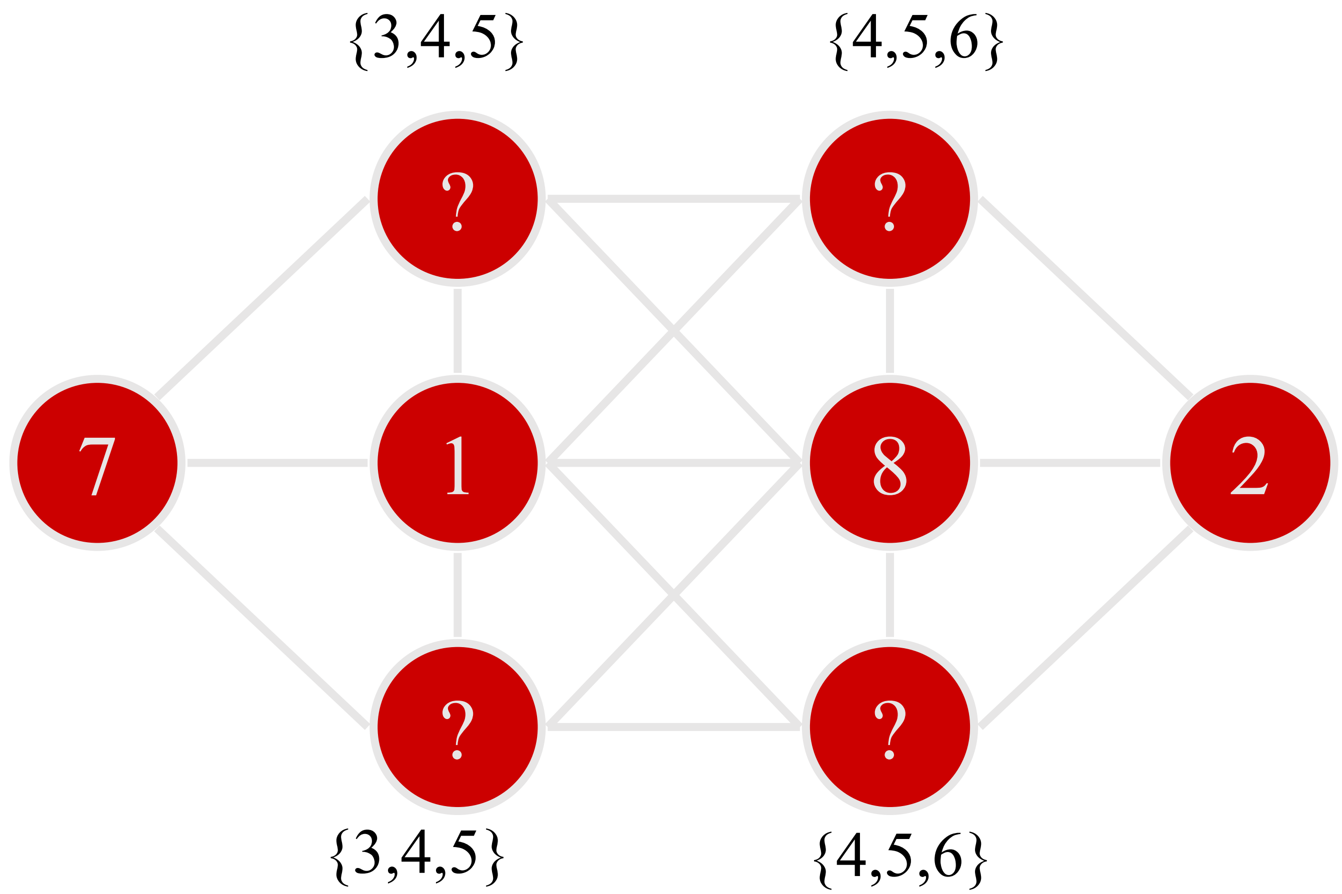

And propagate …

# Inference/propagation



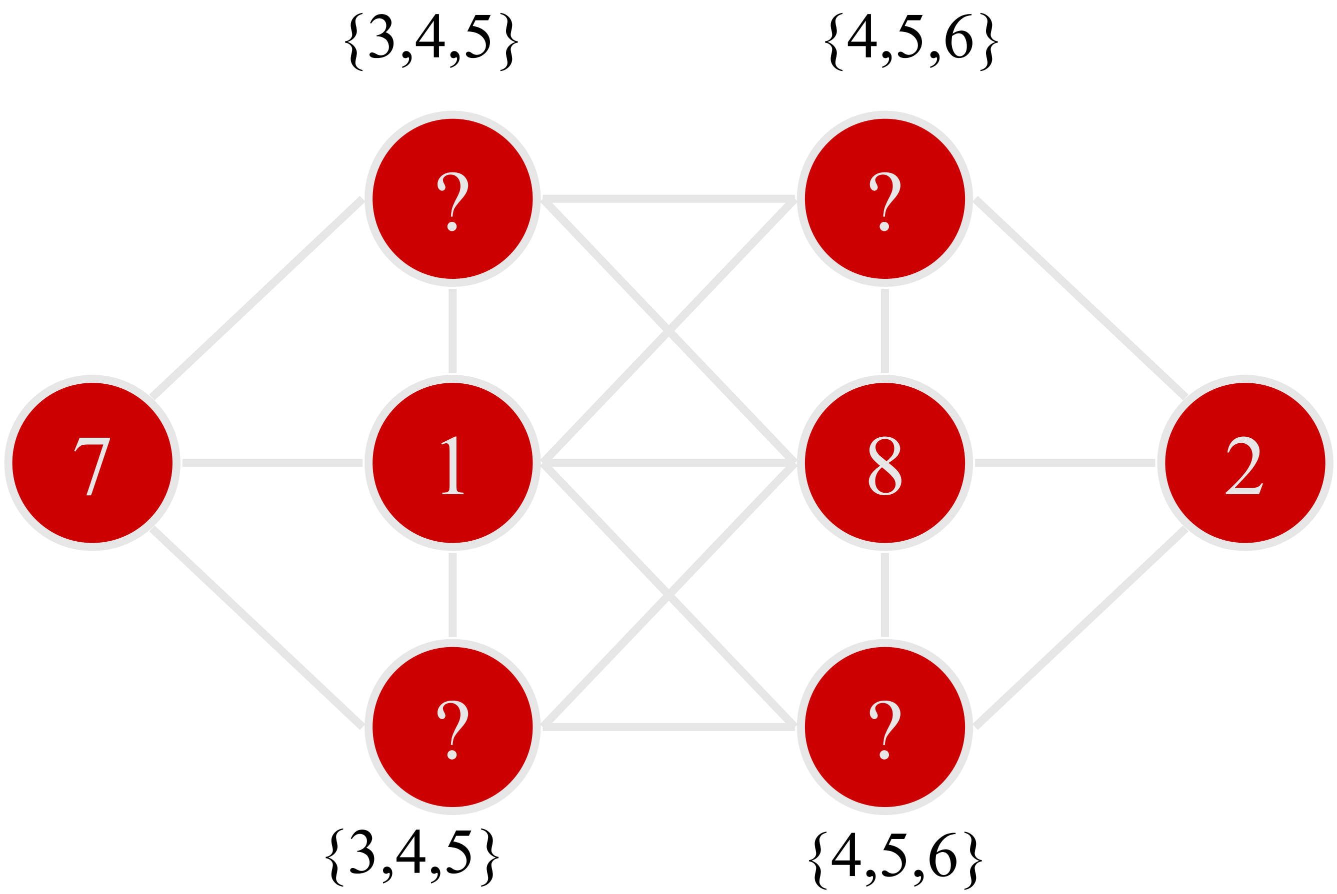

And propagate …

# Inference/propagation



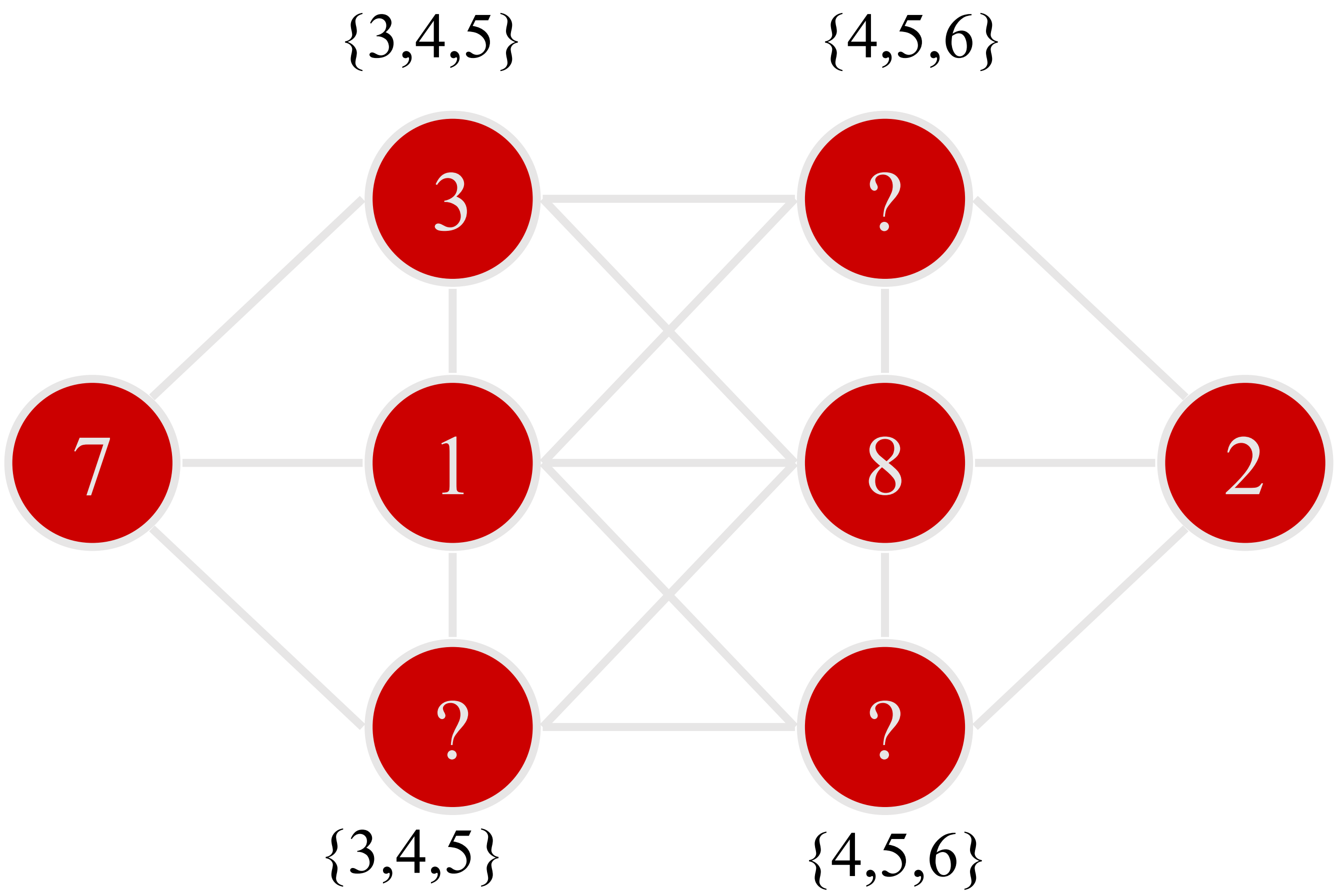

And propagate …

# Inference/propagation



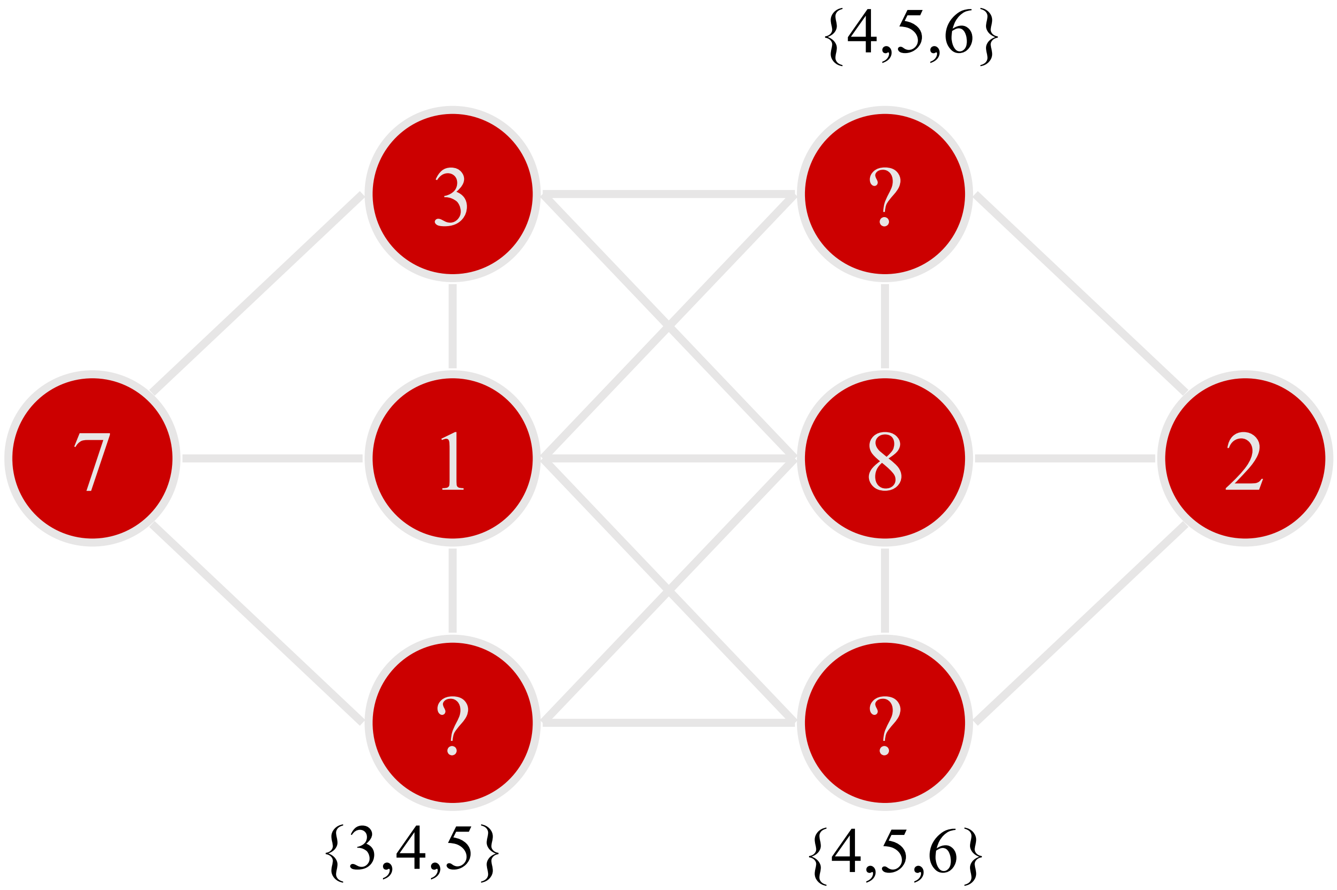

Guess a value, but be prepared to backtrack …

# Inference/propagation



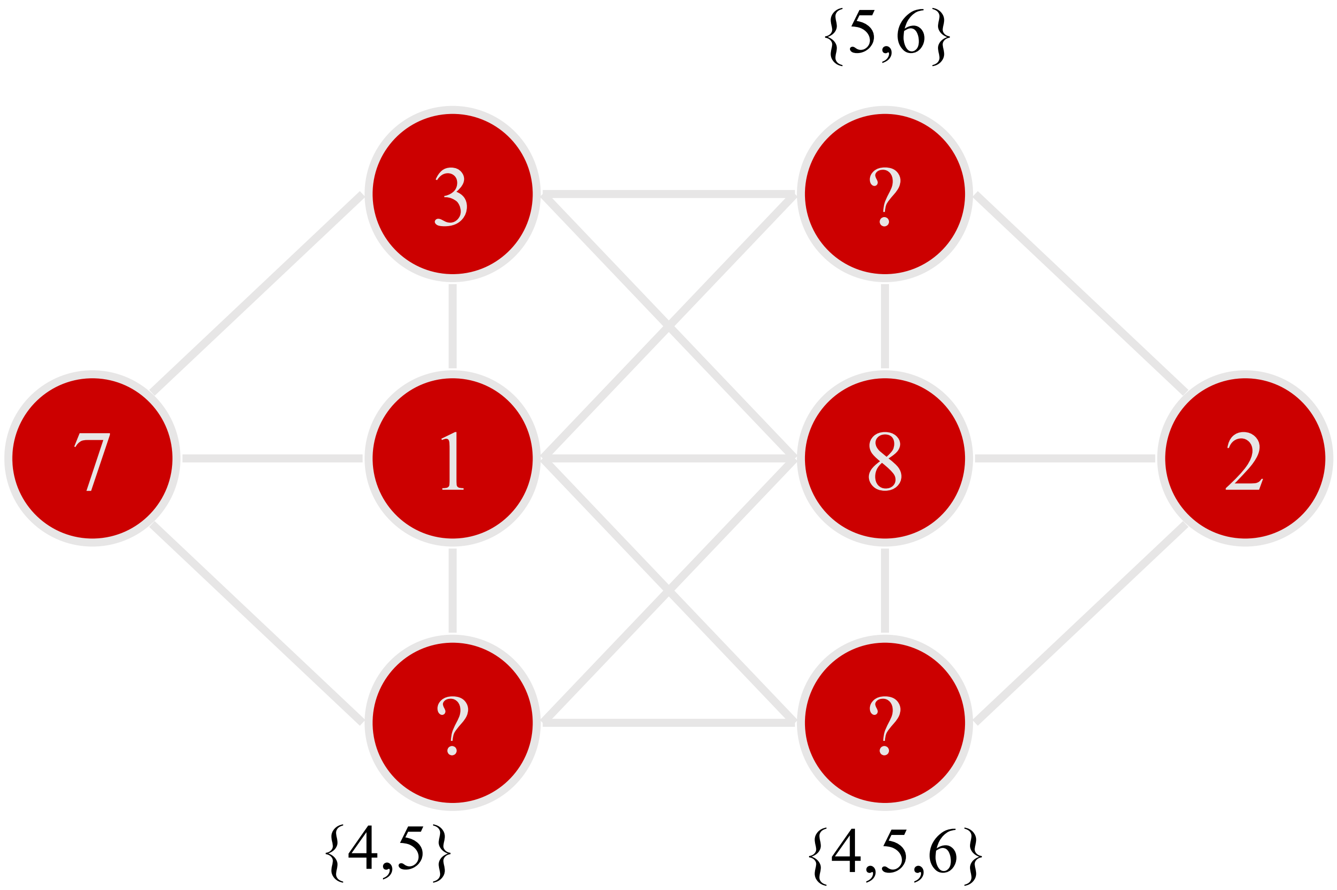

Guess a value, but be prepared to backtrack …

# Inference/propagation



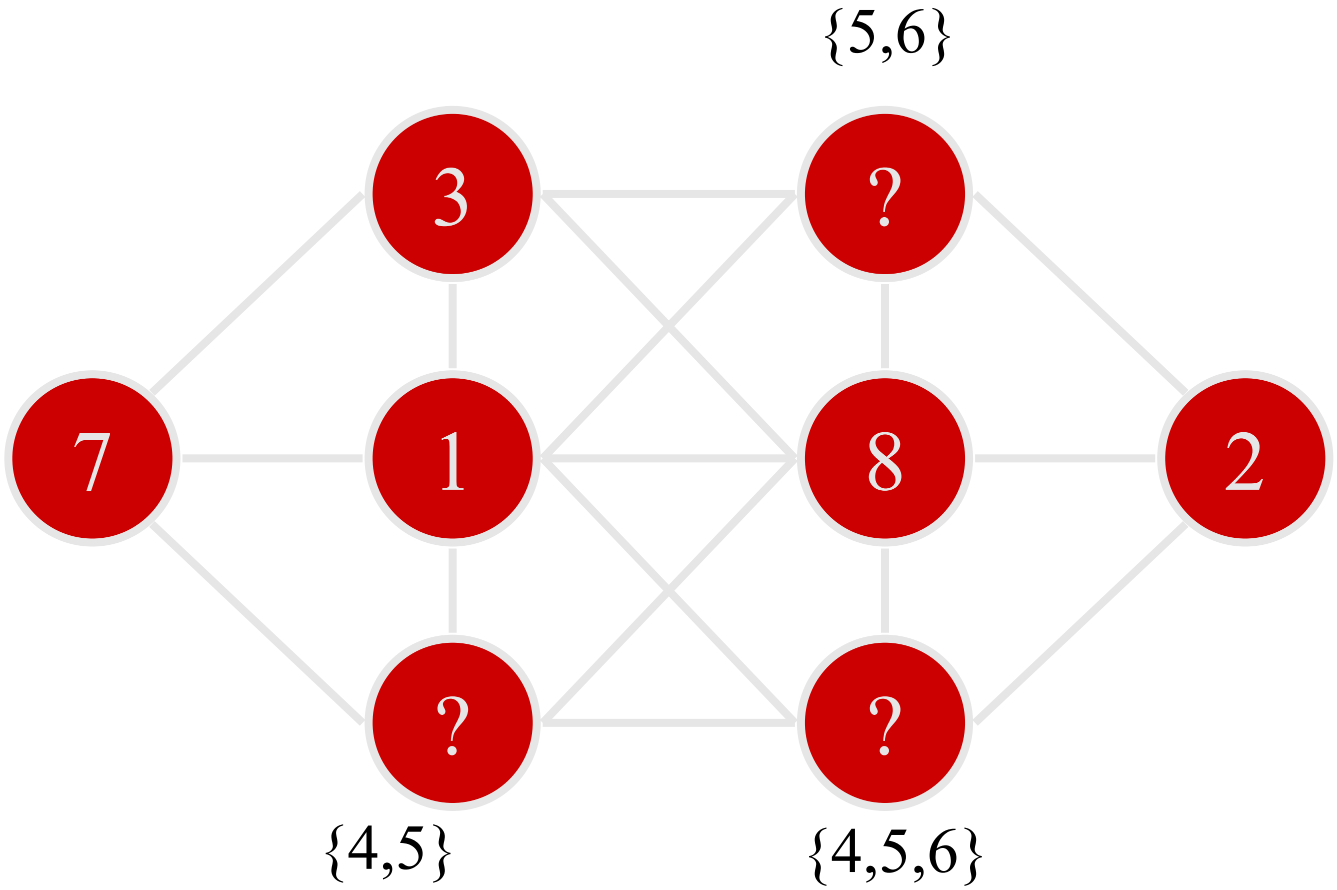

And propagate …

# Inference/propagation



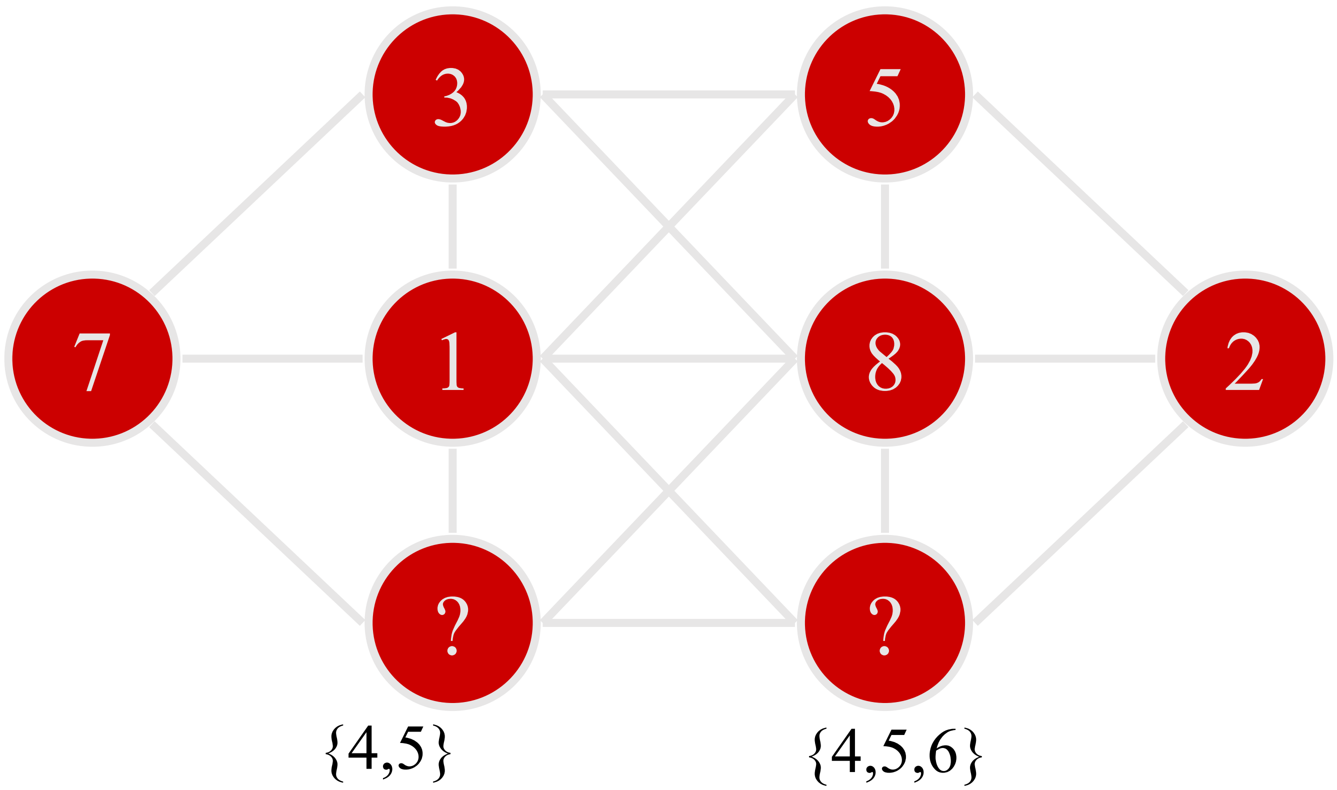

And propagate …

# Inference/propagation



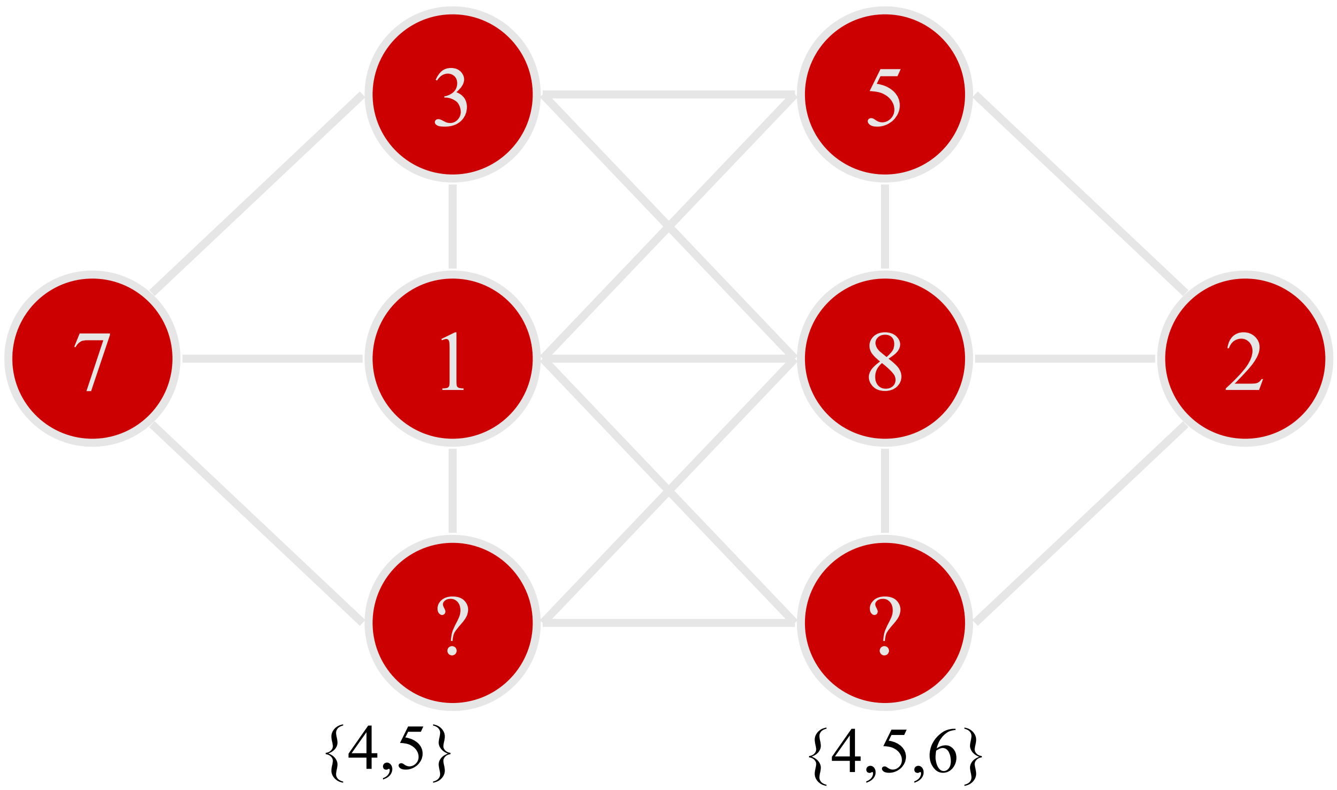

Guess another value …

# Inference/propagation



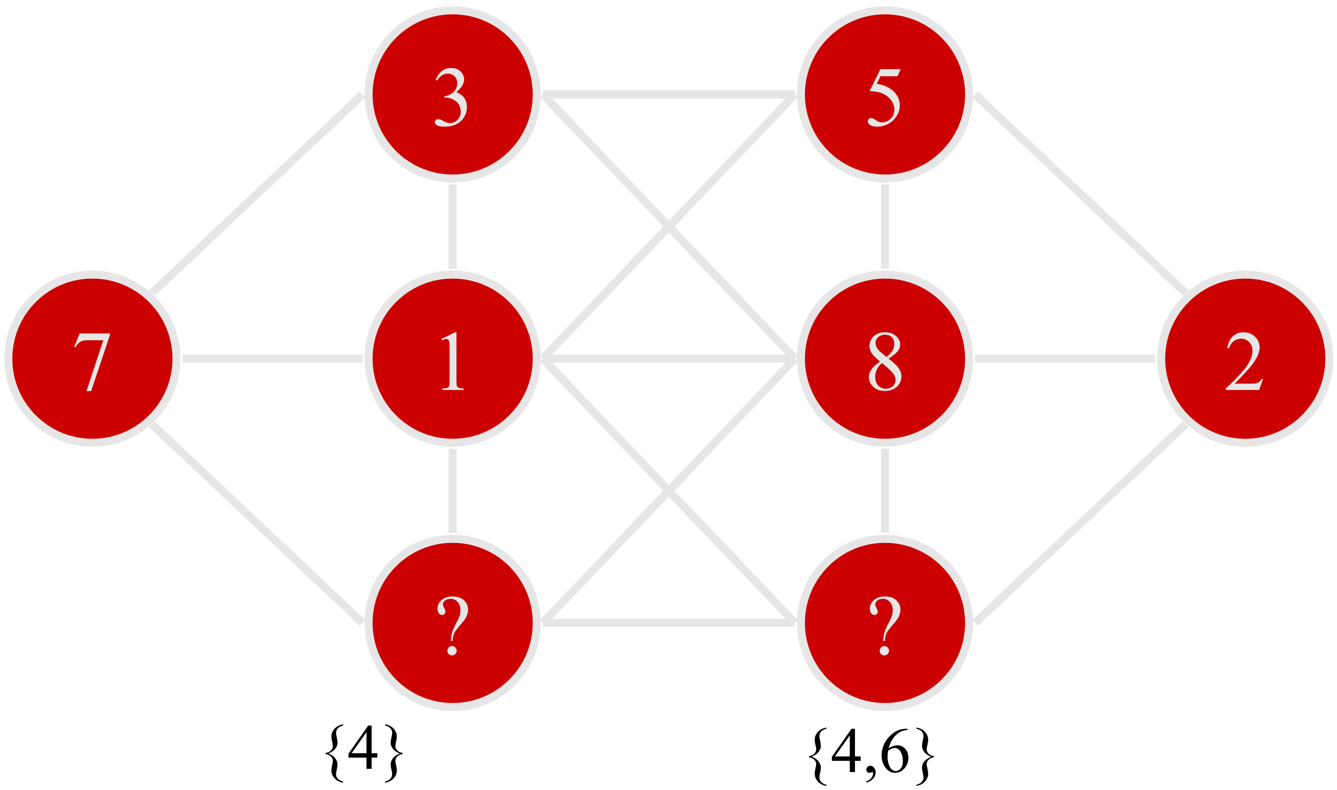

Guess another value …

# Inference/propagation



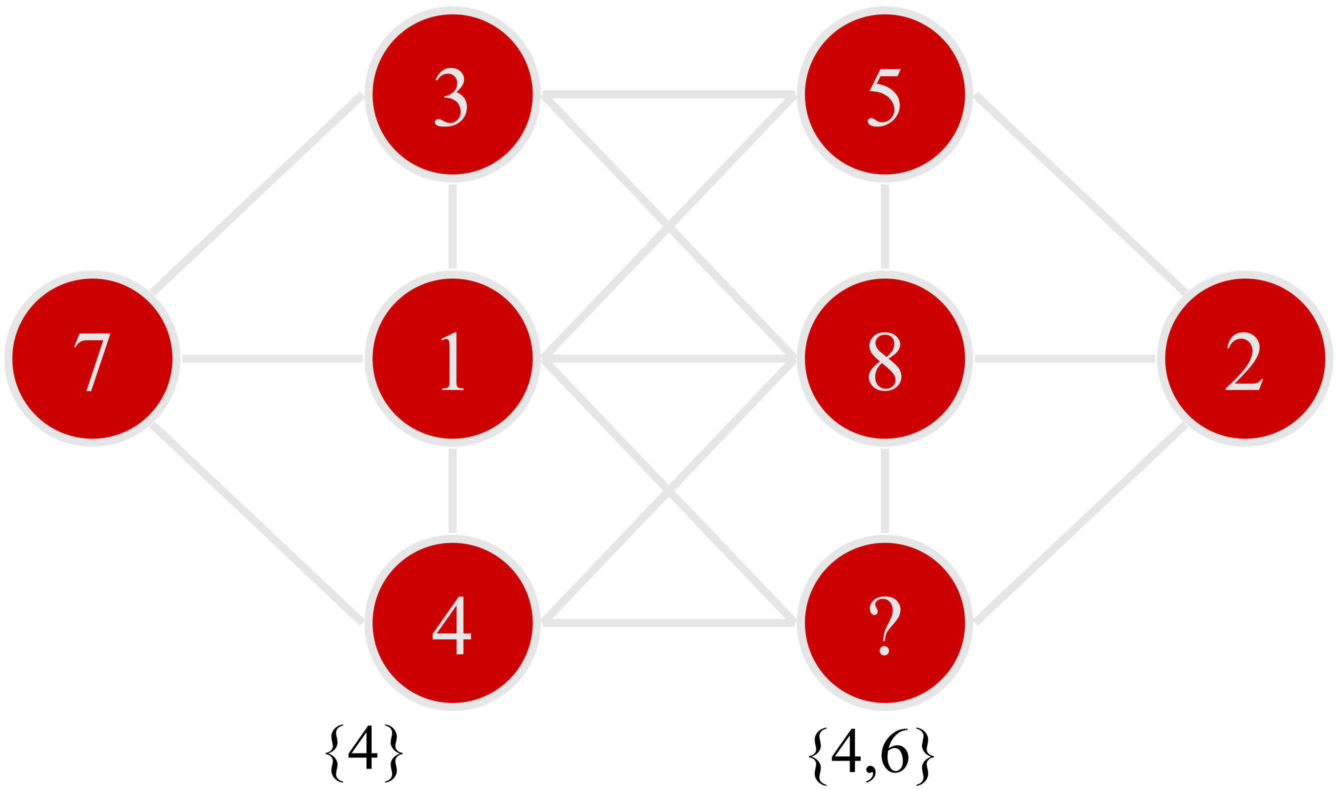

And propagate …

# Inference/propagation



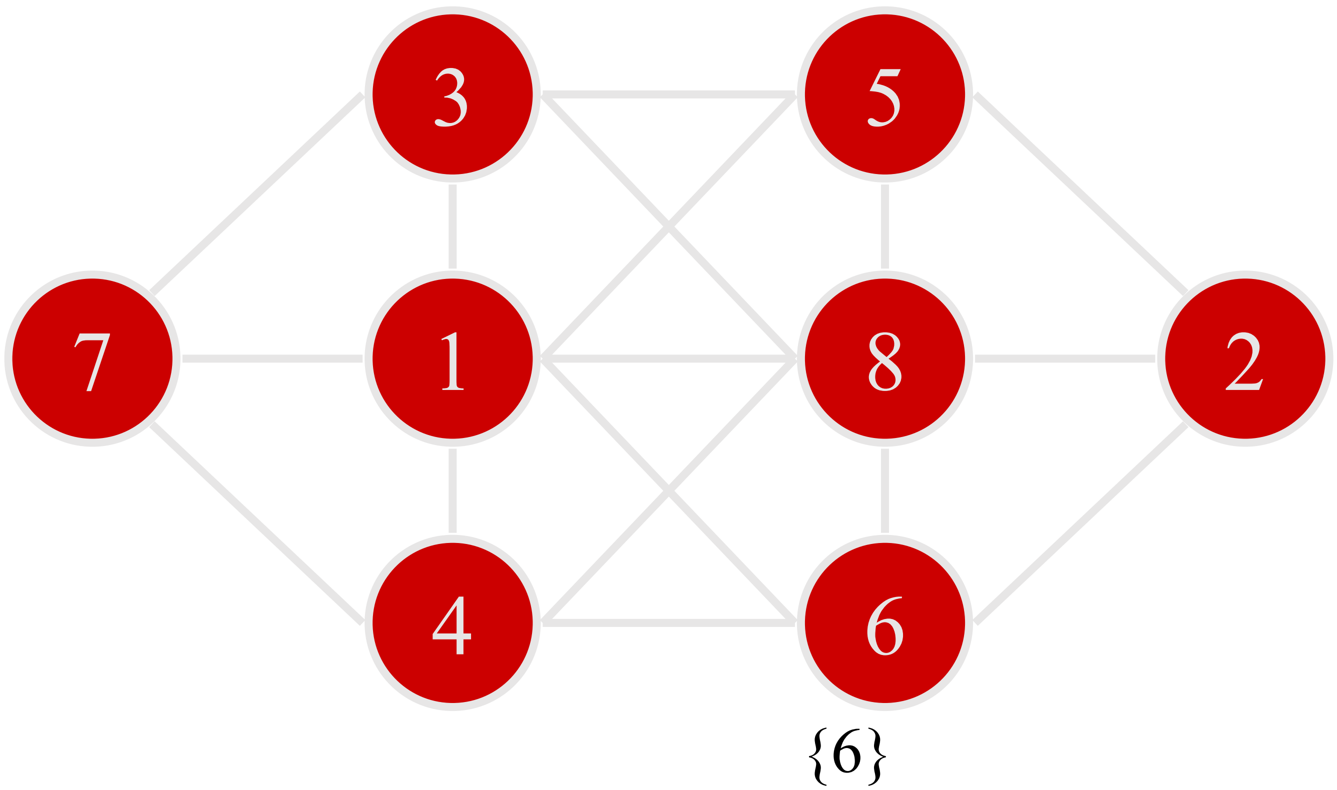

And propagate …

# Inference/propagation

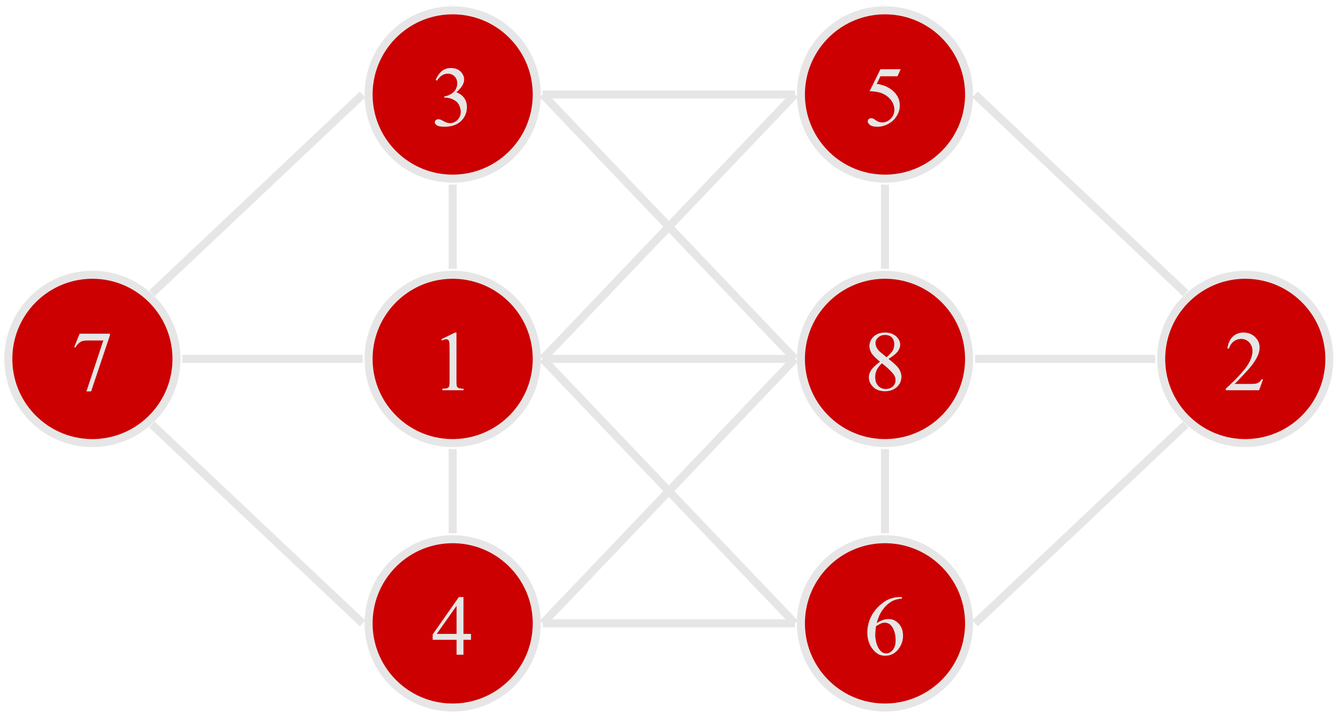


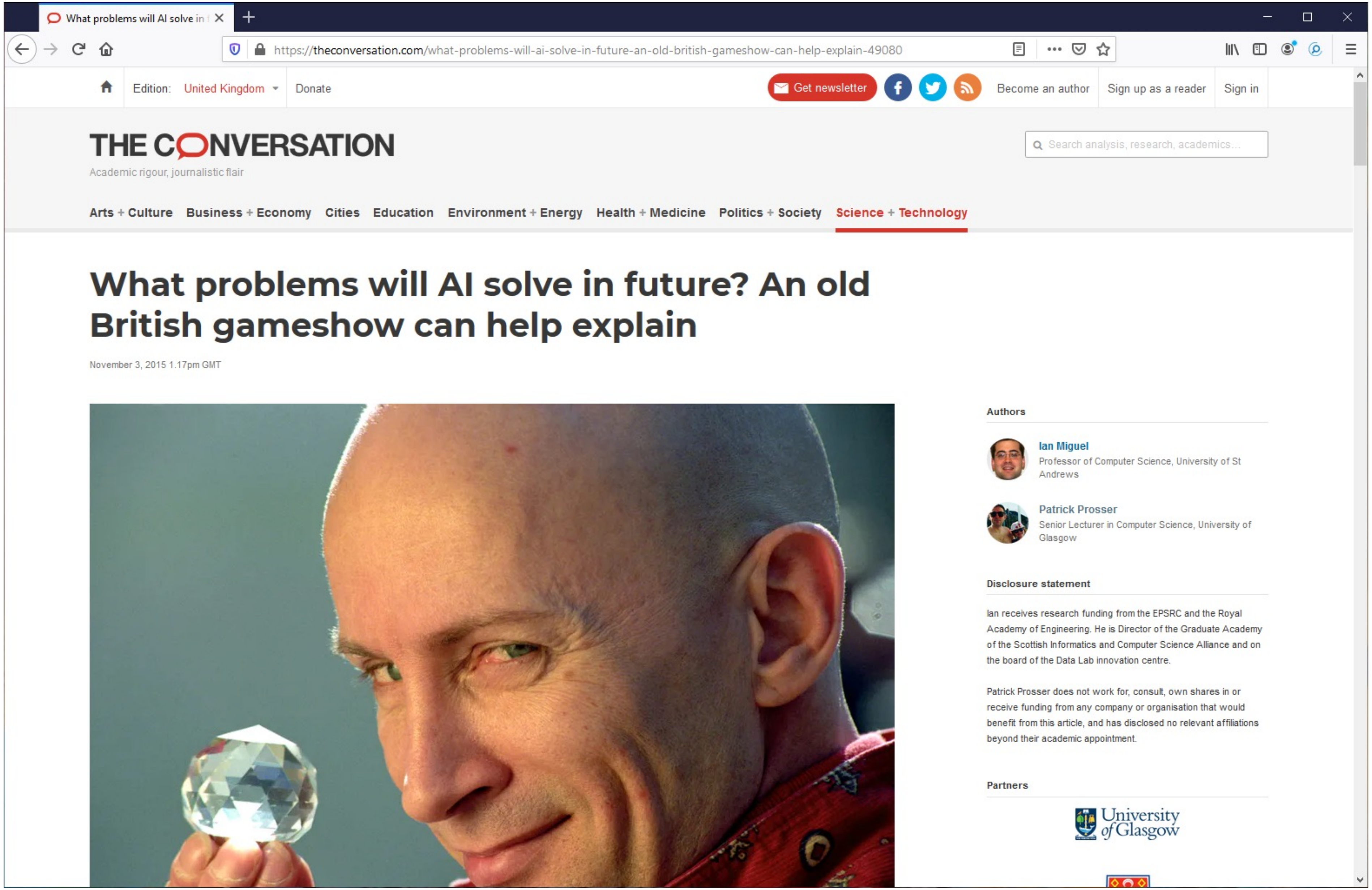

One node has only a single value left …

# Inference/propagation



# Solution





What problems will AI solve in future? An old British gameshow can help explain

https://theconversation.com/what-problems-will-ai-solve-in-future-an-old-british-gameshow-can-help-explain-49080

Edition: United Kingdom Donate

Get newsletter Become an author Sign up as a reader Sign in

THE CONVERSATION

Academic rigour, journalistic flair

Search analysis, research, academics...

Arts + Culture Business + Economy Cities Education Environment + Energy Health + Medicine Politics + Society Science + Technology

# What problems will AI solve in future? An old British gameshow can help explain

November 3, 2015 1.17pm GMT

**Authors**


Ian Miguel
Professor of Computer Science, University of St Andrews

Patrick Prosser
Senior Lecturer in Computer Science, University of Glasgow


**Disclosure statement**


Ian receives research funding from the EPSRC and the Royal Academy of Engineering. He is Director of the Graduate Academy of the Scottish Informatics and Computer Science Alliance and on the board of the Data Lab innovation centre.

Patrick Prosser does not work for, consult, own shares in or receive funding from any company or organisation that would benefit from this article, and has disclosed no relevant affiliations beyond their academic appointment.

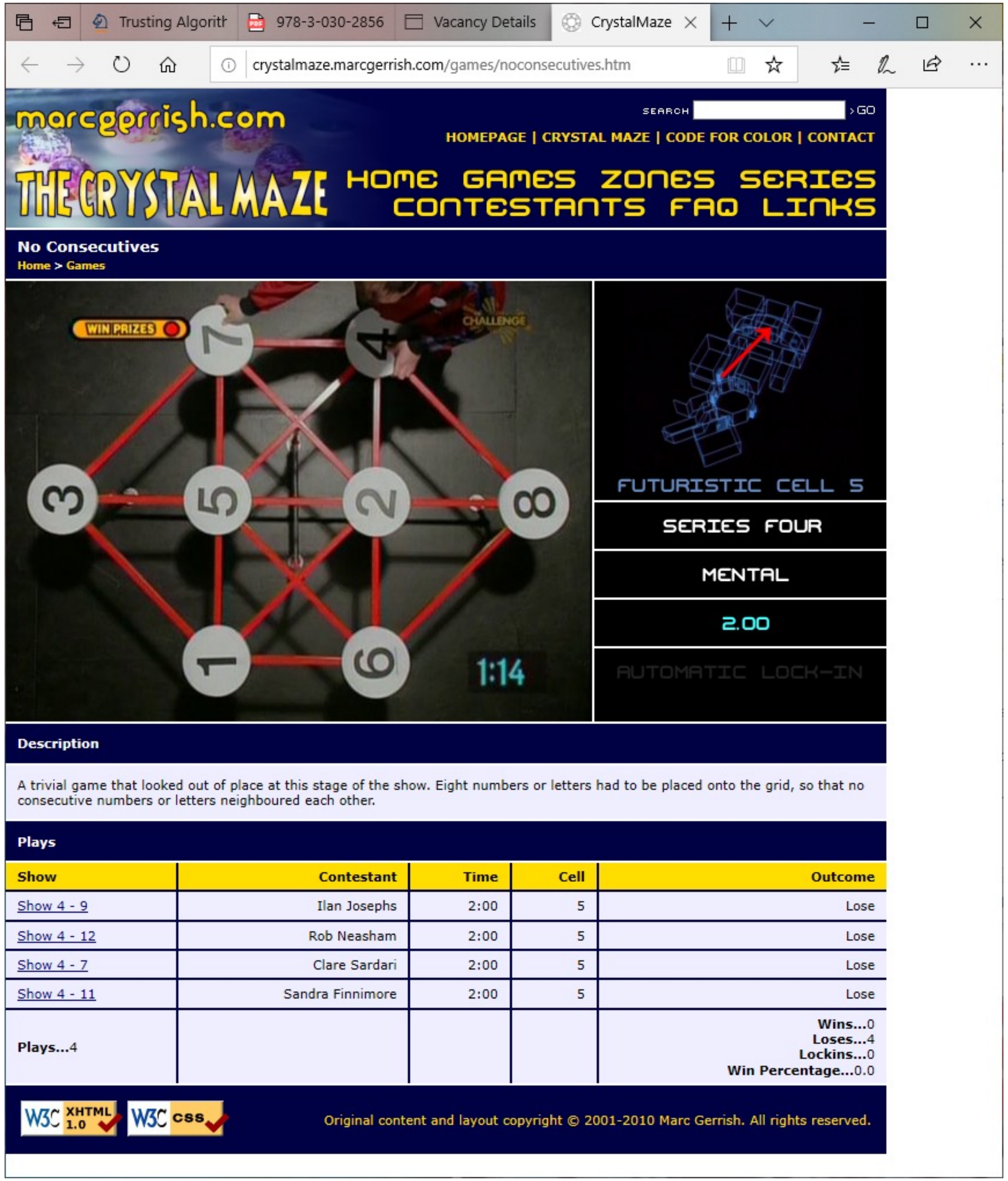

**Partners**

University of Glasgow

https://theconversation.com/what-problems-will-ai-solve-in-future-an-old-british-gameshow-can-help-explain-49080

The Crystal Maze was a popular UK television show from the early 1990s

This puzzle was never solved in the two-minute time frame


crystalmaze.marcgerrish.com/games/noconsecutives.htm

marcgerrish.com

HOMEPAGE | CRYSTAL MAZE | CODE FOR COLOR | CONTACT

THE CRYSTAL MAZE

HOME GAMES ZONES SERIES CONTESTANTS FAQ LINKS

**No Consecutives**

Home > Games

FUTURISTIC CELL 5

SERIES FOUR

MENTAL

2.00

AUTOMATIC LOCK-IN

**Description**

A trivial game that looked out of place at this stage of the show. Eight numbers or letters had to be placed onto the grid, so that no consecutive numbers or letters neighboured each other.

**Plays**

| Show | Contestant | Time | Cell | Outcome |
|---|---|---|---|---|
| Show 4 - 9 | Ilan Josephs | 2:00 | 5 | Lose |
| Show 4 - 12 | Rob Neasham | 2:00 | 5 | Lose |
| Show 4 - 7 | Clare Sardari | 2:00 | 5 | Lose |
| Show 4 - 11 | Sandra Finnimore | 2:00 | 5 | Lose |
| Plays...4 | | | | Wins...0<br>Loses...4<br>Lockins...0<br>Win Percentage...0.0 |

# The Core of Constraint Computation

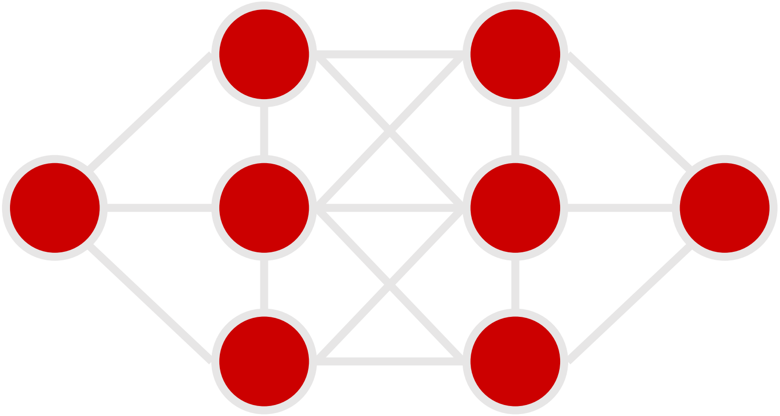
- Modelling
  - Deciding on variables/domains/constraints
- Heuristic Search
- Inference/Propagation
- Symmetry
- Backtracking

# A Commercial Reality

- First-tier software vendors use CP technology

# Hardness

- The puzzle is actually a hard problem
  - NP-complete



# Constraint programming

- Model problem by specifying constraints on acceptable solutions
    - define variables and domains
    - post constraints on these variables
- Solve model
    - choose algorithm
        - incremental assignment / backtracking search
        - complete assignments / stochastic search
    - design heuristics

# Constraint satisfaction

- Constraint satisfaction problem (CSP) is a triple <V,D,C> where:
  - V is set of variables
  - Each X in V has set of values, $D_X$
    - Usually assume finite domain
    - {true,false}, {red,blue,green}, [0..10], …
  - C is set of constraints

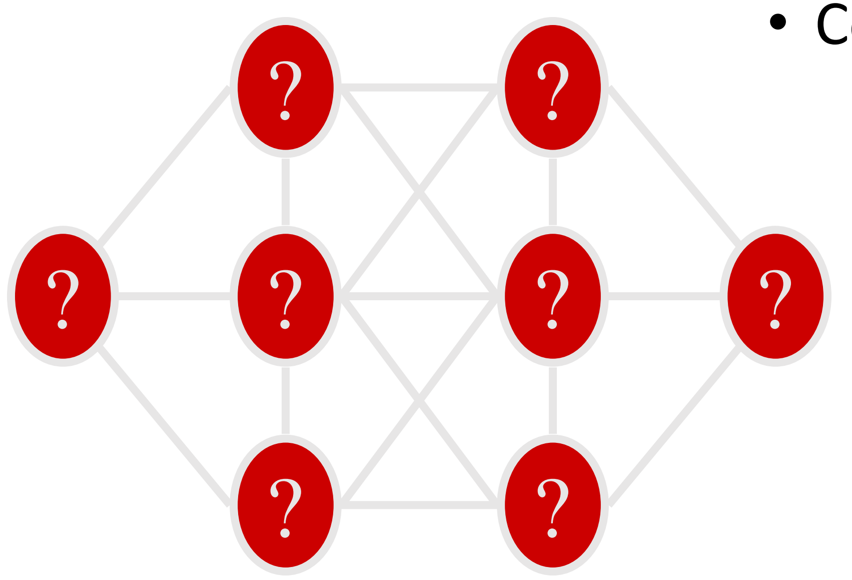
Goal: find assignment of values to variables to satisfy all the constraints

How complex?

Assume

- n variables
- each with a domian size of m
- how many states might we consider?

# Example CSP




- Variable, $v_i$ for each node
- Domain of {1, …, 8}
- Constraints
  - All values used

  allDifferent($v_1$ $v_2$ $v_3$ $v_4$ $v_5$ $v_6$ $v_7$ $v_8$)

  - No consecutive numbers for adjoining nodes

  $|v_1 - v_2| > 1$
  $|v_1 - v_3| > 1$
  …

# Constraints

- Constraints are tuples <S,R> where
  - S is the scope, [X1,X2, … Xm]
    - list of variables to which constraint applies
  - R is relation specifying allowed values (goods)
    - Subset of D_X1 x D_X2 x … x D_Xm
    - May be specified intensionally or extensionally

# Constraints

- Extensional specification
  - List of goods (or for tight constraints, nogoods)
- Intensional specification
  - X1 =/= X2
  - 5*X1 + 6*X2 < X3
  - alldifferent([X1,X2,X3,X4]), …

more examples?

Do you know any constraint satisfaction problems?

To a man with a hammer, everything looks like a nail.

Sudoku - Wikipedia, the free encyclopedia - Windows Internet Explorer

http://en.wikipedia.org/wiki/Sudoku

File Edit View Favorites Tools Help

Page Tools



# Sudoku

From Wikipedia, the free encyclopedia

*Not to be confused with Sodoku.*

**Sudoku** (数独 *sūdoku*, listen (help·info)) (English pronunciation: /suːˈdoʊkuː/) is a logic-based,[1][2] combinatorial[3] number-placement puzzle. The objective is to fill a 9×9 grid so that each column, each row, and each of the nine 3×3 boxes (also called blocks or regions) contains the digits from 1 to 9 only one time each. The puzzle setter provides a partially completed grid. Completed puzzles are usually a type of Latin square with an additional constraint on the contents of individual regions.

Sudoku was popularized in 1986 by the Japanese puzzle company Nikoli, under the name Sudoku, meaning *single number*.[4] It became an international hit in 2005.[5]

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

A Sudoku puzzle...

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

**Contents** [hide]

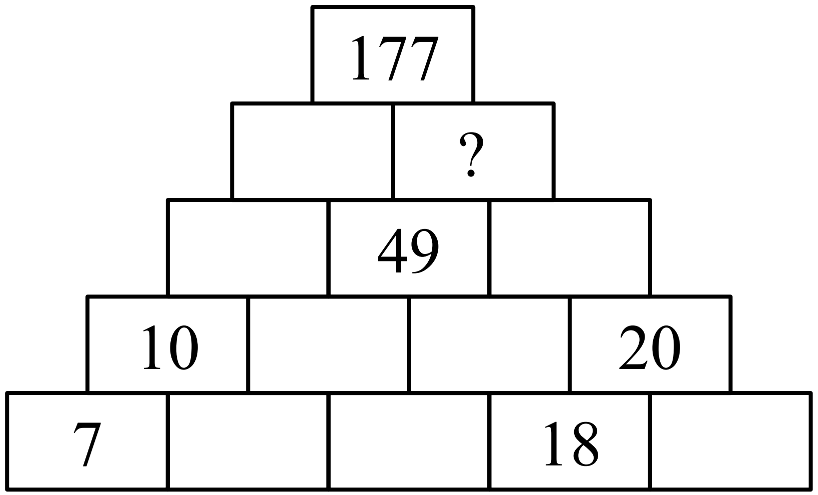


## History [edit]

Number puzzles first appeared in newspapers in the late 19th century, when French puzzle setters began experimenting with removing numbers from magic squares. *Le Siècle*, a Paris-based daily, published a partially completed 9×9 magic square with 3×3 sub-squares on November 19, 1892.[6] It was not a Sudoku because it contained double-digit numbers and required arithmetic rather than logic to solve, but it shared key characteristics: each row, column and sub-square added up to the same number.

start | slides | 3 Internet Explo... | Microsoft PowerP... | 10:42

# Scotsman 4/12/2003




In the pyramid above, two adjacent bricks added together give the value of the brick above. Find the value for the brick marked ?

# Exam timetabling

# An Example, Exam Timetabling

- Someone timetables the exams
- We have a number of courses to examine
    - how many?
        - Dept has 36
        - Faculty?
        - University?
- There are constraints
    - if a student S takes courses Cx and Cy
        - Cx and Cy cannot be at same time!
    - If Cy and Cz have no students in common
        - they can go in room R1 if there is space
    - Temporal and resource constraints

# An Example, Exam Timetabling

- Represent as graph colouring
  - vertices are courses
  - colours are time
  - vertices have weight (room requirements)
  - edge connects vertices of diff colour
- How complex is this
  - if we have n vertices and k times
    - an n-digit number to the base k?
- How would you solve this
  - backtracking search?
  - Greedy?
  - Something else?

## An Example, Exam Timetabling

- How does the person solve this?
- Is that person intelligent?
- Is there always a solution?
- If there isn't, do we want to know why?
    - Do you think they can work out "why"?

# Crossword puzzle generation

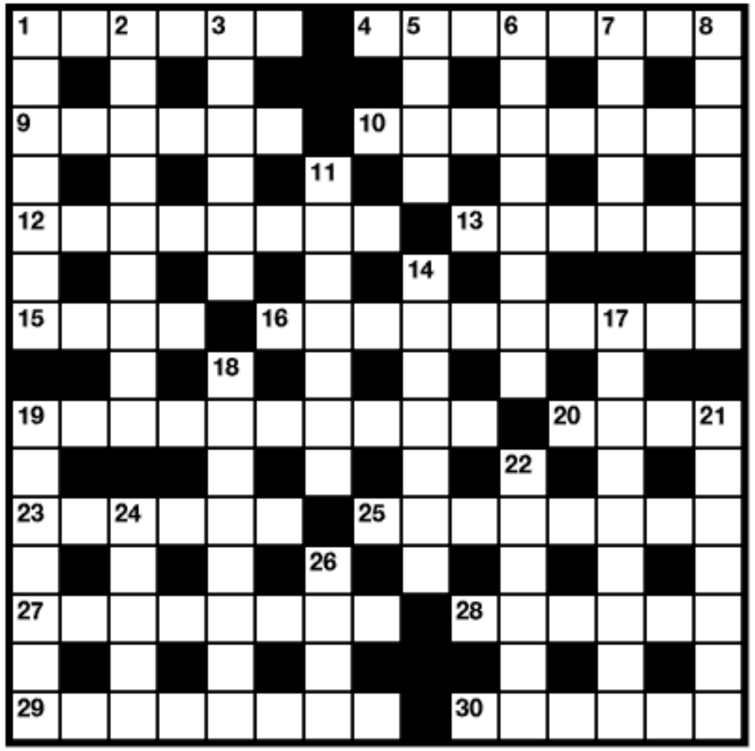
Printer Friendly Format - - Windows Internet Explorer

http://www.heraldandtimeslabs.com/dev/crosswords/print-crossword.php

File Edit View Favorites Tools Help

Page Tools

heraldscotland



Monday, 28 Sep 2009

**Across**

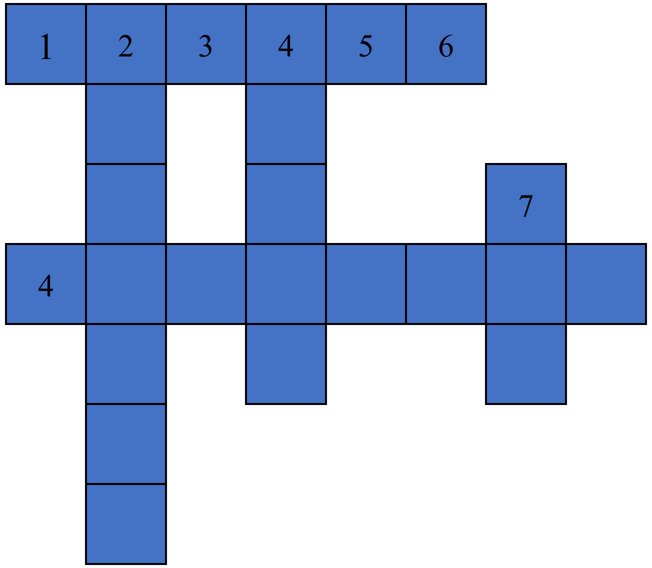
**1** Mass-produced legislation? [3,3]

**Down**

**1** Not subject put on in advance [7]

start slides 3 Internet Explo... Microsoft PowerP... 10:39

## An example

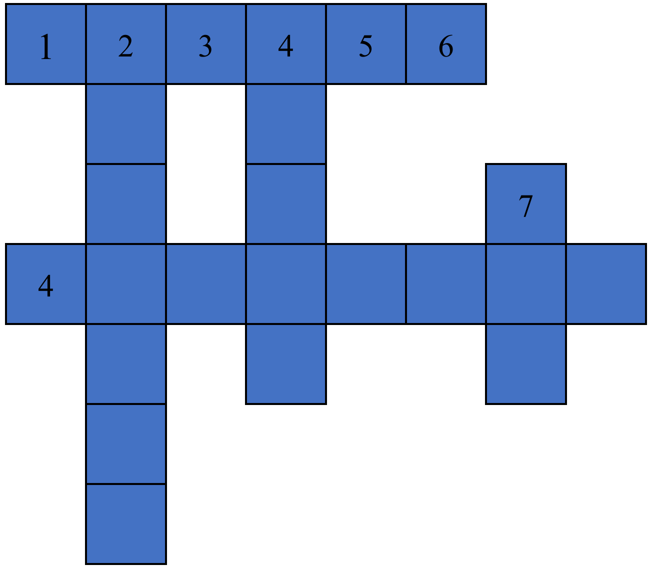


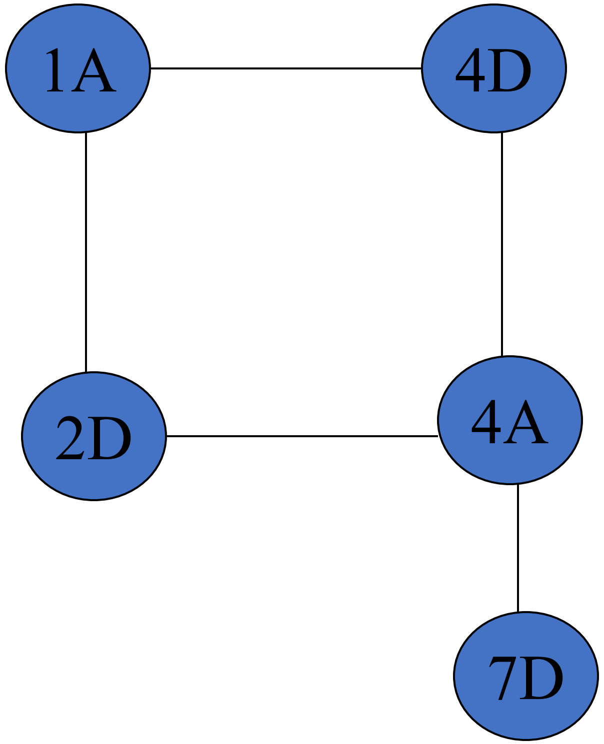

Make a crossword puzzle!

Given the above grid and a dictionary, fill it.

Then go get the clues (not my problem)

# An example

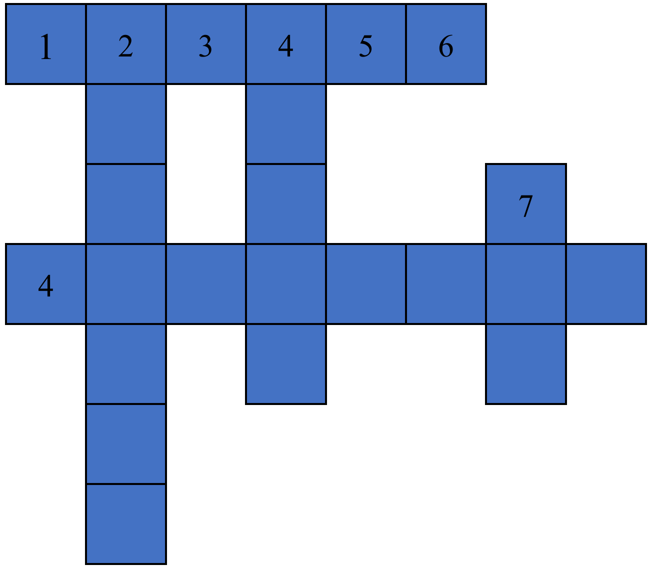


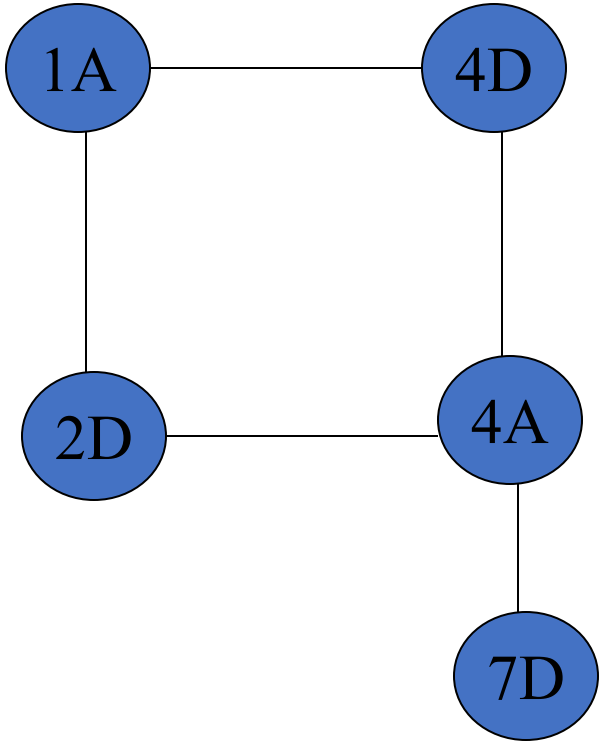




| | |
|---|---|
| 1A | 1 across |
| 4D | 4 down |
| 2D | 2 down |
| 4A | 4 across |
| 7D | 7 down |

Variables

# An example




1A-4D: 4th of 1A equals 1st of 4D

1A-2D: 2nd of 1A equals 1st of 2D

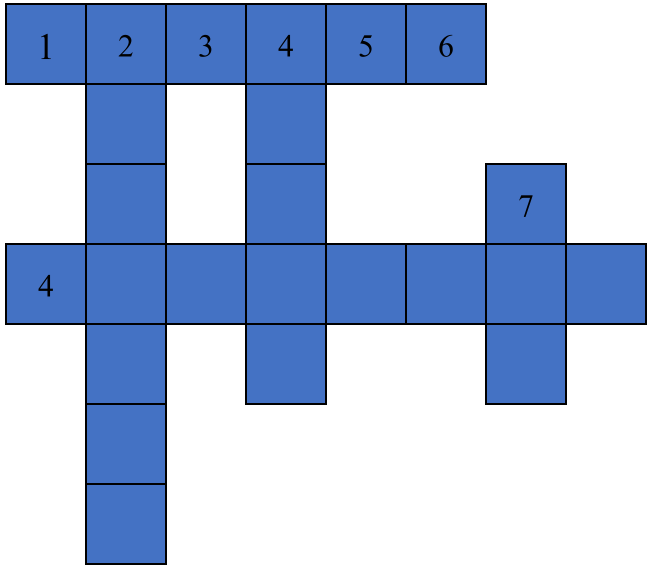
2D-4A: 4th of 2D equals 2nd of4D

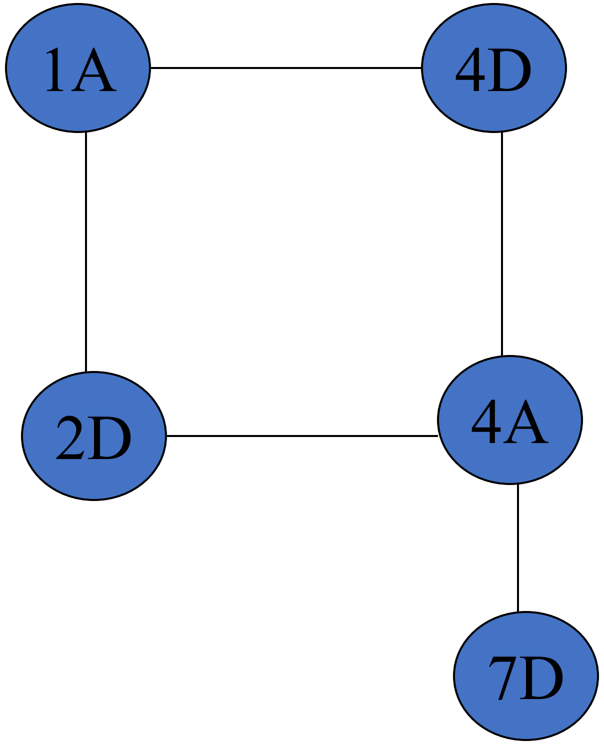
4D-4A: 4th of 4A equals 4th of 4D

4A-7D: 7th of 4A equals 2nd of 7D

Constraints

# An example

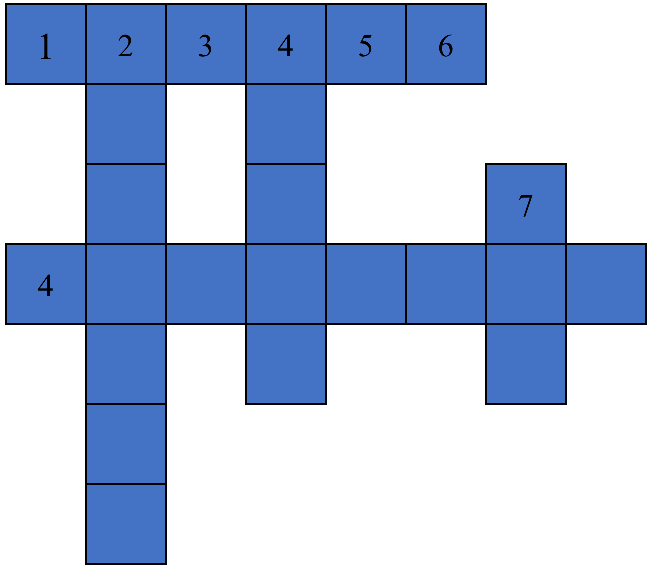


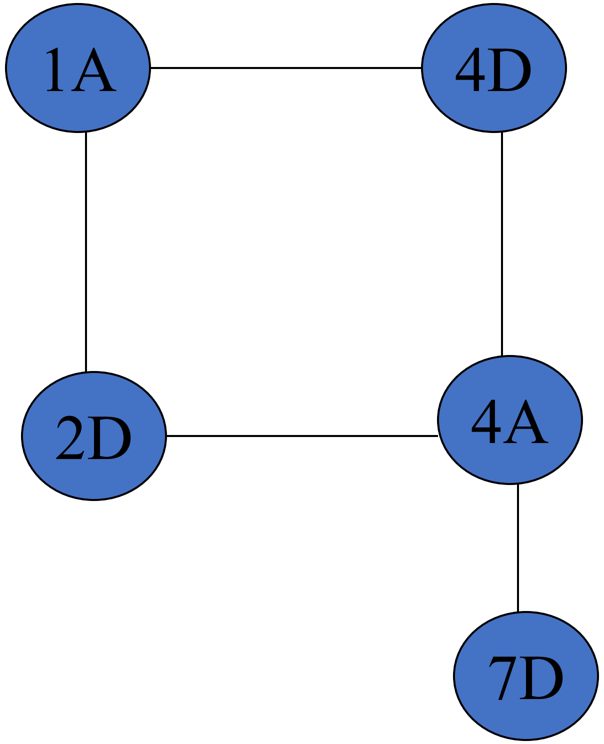




1A: any 6 letter word
4A: any 8 letter word
4D: any 5 letter word
2D: any 7 letter word
7D: any 3 letter word

Domains (also unary constraints!)

# An example

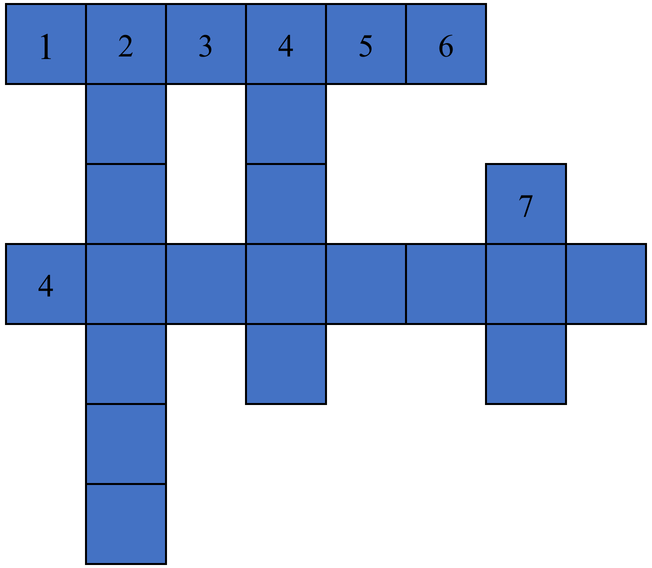


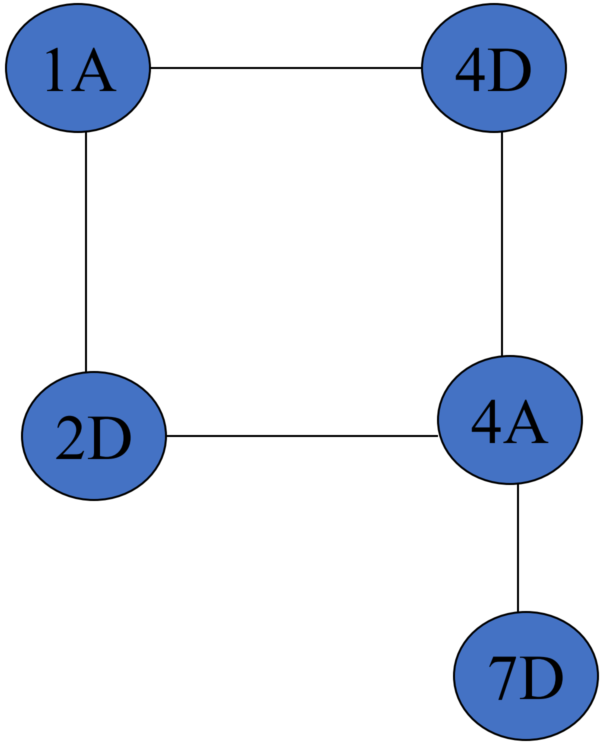




Find an assignment of values to variables, from their domains, such that the constraints are satisfied (or show that no assignment exists)

A CSP!

# An example




Choose a variable

Assign it a value

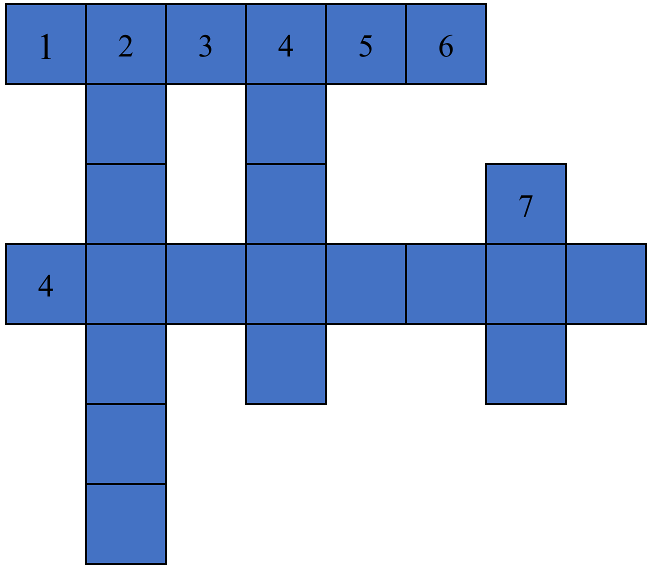
Check compatibility

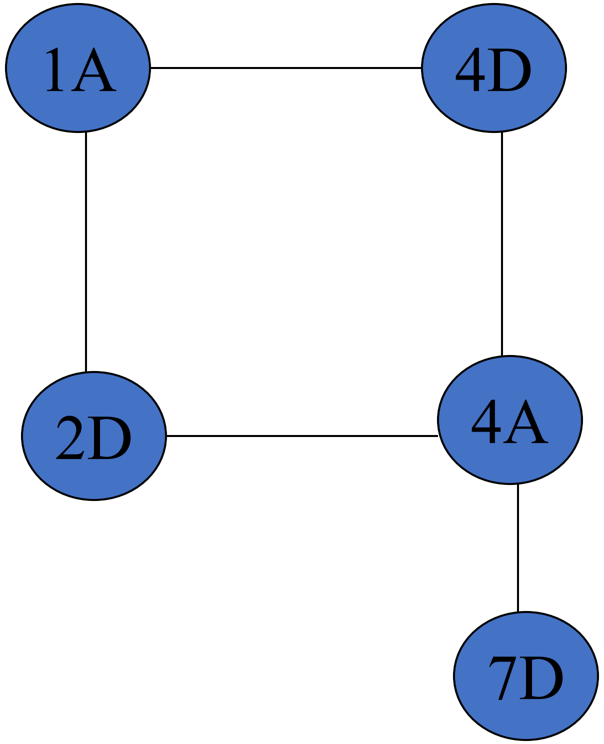
If not compatible try a new value

If no values remain re-assign previous variable

Good old fashioned backtracking!

# Questions?




What variable should I choose?

What value should I choose?

What reasoning can I do when making an assignment?

What reasoning can I do on a dead end?

Decisions, decisions!

# Problems of interest to CP

These are some of the problems that have been tackled by CP

- factory scheduling (JSSP)
- vehicle routing (VRP)
- packing problems (NumPart and BinPack)
- timetabling (exams, lectures, trains)
- configuration and design (hardware)
- workforce management (call centres, etc)
- car sequencing (assembly line scheduling)
- supertree construction (bioinformatics)
- network design (telecoms problem)
- gate arrival (at airports)
- logistics (Desert Storm an example)
- aircraft maintenance schedules
- aircraft crew scheduling (commercial airlines)
- air cover for naval fleet

# What will be covered in course

- the technology behind constraint programming (CP)
- CP in MiniZinc
- modelling and solving problems
- the state of the art

# MiniZinc: a CSP solver

- On the web https://www.minizinc.org/
  - MiniZinc is a free and open-source **constraint modeling language**
- MiniZinc Handbook https://www.minizinc.org/doc-2.6.4/en/index.html
- Coursera's Basic Modeling for Discrete Optimization and Advanced Modeling for Discrete Optimization courses for an in-depth introduction to constraint modeling using MiniZinc.
- Download from https://www.minizinc.org/software.html

# Crystal Maze with MiniZinc


That's all Folks!