Constraint Satisfaction Problems

Introduction

Bernhard Nebel, **Julien Hué**, and **Stefan Wölfl**

Albert-Ludwigs-Universität Freiburg

April 23, 2012

Introduction

Constraints

What is a constraint?

1 a: the act of constraining **b:** the state of being checked, restricted, or compelled to avoid or perform some action ...
c: a constraining condition, agency, or force ...
2 a: repression of one's own feelings, behavior, or actions
b: a sense of being constrained ...

(from *Merriam-Webster's Online Dictionary*)

Usage

- In programming languages, constraints are often used to restrict the domains of variables.
- In databases, constraints can be used to specify integrity conditions.
- In mathematics, a constraint is a requirement on solutions of optimization problems.

Examples

Examples:

- Latin squares
- Eight queens problem
- Sudoku
- Map coloring problem
- Boolean satisfiability

Latin Square

Problem:

- How can one fill an $n \times n$ table with n different symbols
- ... such that each symbol occurs exactly once in each row and and each column?

$$\begin{bmatrix}1\end{bmatrix} \quad \begin{bmatrix}1 & 2\\2 & 1\end{bmatrix} \quad \begin{bmatrix}1 & 2 & 3\\2 & 3 & 1\\3 & 1 & 2\end{bmatrix} \quad \begin{bmatrix}1 & 2 & 4 & 3\\2 & 3 & 1 & 4\\3 & 4 & 2 & 1\\4 & 1 & 3 & 2\end{bmatrix}$$

There are essentially 56 different Latin squares of size 5, 9408 squares of size 6, 16.942.080 squares of size 7, 535.281.401.856 squares of size 8, ...

Latin Square

Problem:

- How can one fill an $n \times n$ table with n different symbols
- ... such that each symbol occurs exactly once in each row and and each column?

$$\begin{bmatrix}1\end{bmatrix} \quad \begin{bmatrix}1 & 2\\2 & 1\end{bmatrix} \quad \begin{bmatrix}1 & 2 & 3\\2 & 3 & 1\\3 & 1 & 2\end{bmatrix} \quad \begin{bmatrix}1 & 2 & 3 & 4\\2 & 3 & 4 & 1\\3 & 4 & 1 & 2\\4 & 1 & 2 & 3\end{bmatrix}$$

There are essentially 56 different Latin squares of size 5, 9408 squares of size 6, 16.942.080 squares of size 7, 535.281.401.856 squares of size 8, ...

Eight Queens Puzzle

Problem:

- How can one put 8 queens on a standard chess board (8×8-board)
- ...such that no queen can attack any other queen?

Solutions:

- The puzzle has 12 unique solutions (up to rotations and reflections)

- Old problem proposed in 1848.
- Various variants
  - knights (instead of queens)
  - 3D
  - n queens on an $n \times n$-board

A Solution . . .

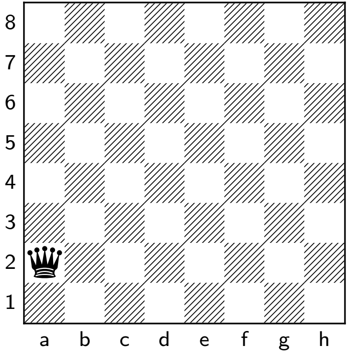

Figure: A solution of the 8-queens problem

A Solution . . .

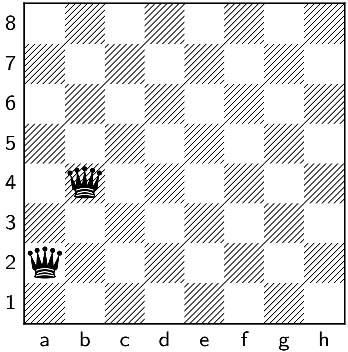

Figure: A solution of the 8-queens problem

A Solution . . .

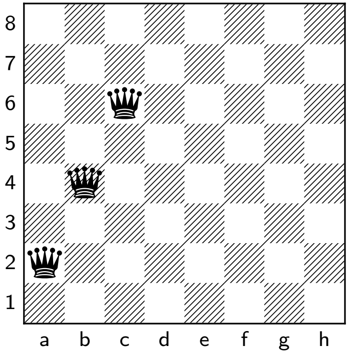

Figure: A solution of the 8-queens problem

A Solution . . .

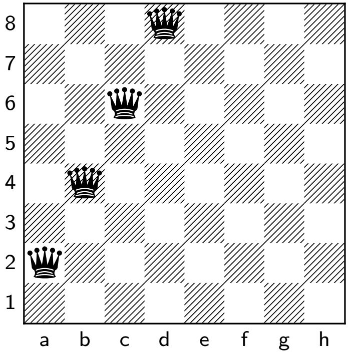

Figure: A solution of the 8-queens problem

A Solution ...

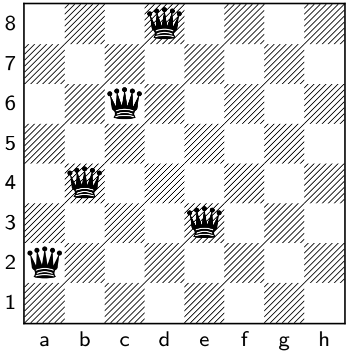

Figure: A solution of the 8-queens problem

A Solution . . .

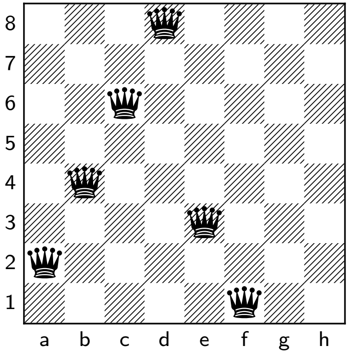

Figure: A solution of the 8-queens problem

A Solution ...

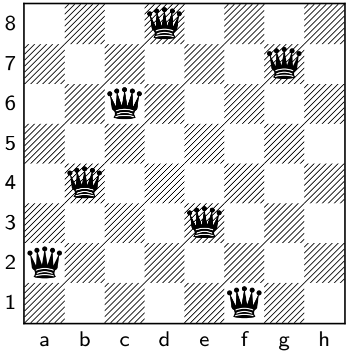

Figure: A solution of the 8-queens problem

A Solution . . .

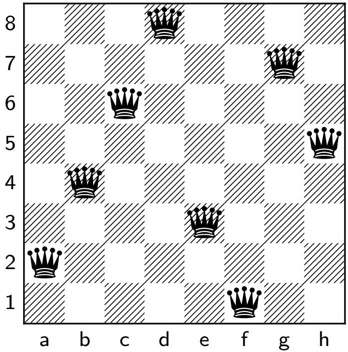

Figure: A solution of the 8-queens problem

Sudoku

Problem:

- Fill a partially completed 9×9 grid such that
- . . . each row, each column, and each of the nine 3×3 boxes contains the numbers from 1 to 9.

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

Sudoku

Problem:

- Fill a partially completed 9×9 grid such that
- . . . each row, each column, and each of the nine 3×3 boxes contains the numbers from 1 to 9.

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

Constraint Satisfaction Problem

Definition

A constraint network is defined by:

- a finite set of variables
- a (finite) domain of values for each variable
- a finite set of constraints (i.e., binary, ternary, . . . relations defined between the variables)

Problem

Is there a solution of the network, i.e., an assignment of values to the variables such that all constraints are satisfied?

k-Colorability

Problem:

- Can one color the nodes of a given graph with k colors
- ... such that all nodes connected by an edge have different colors?

Reformulated as a constraint network:

- Variables: the nodes of the graph
- Domains: "colors" $\{1, \ldots, k\}$ for each variable
- Constraints: nodes connected by an edge must have different values

This constraint network has a particular restricted form:

- only binary constraints
- domains are finite

k-Colorability

Problem:

- Can one color the nodes of a given graph with k colors
- ... such that all nodes connected by an edge have different colors?

Reformulated as a constraint network:

- Variables: the nodes of the graph
- Domains: "colors" $\{1, \ldots, k\}$ for each variable
- Constraints: nodes connected by an edge must have different values

This constraint network has a particular restricted form:

- only binary constraints
- domains are finite

k-Colorability

Problem:

- Can one color the nodes of a given graph with k colors
- ... such that all nodes connected by an edge have different colors?

Reformulated as a constraint network:

- Variables: the nodes of the graph
- Domains: "colors" $\{1, \ldots, k\}$ for each variable
- Constraints: nodes connected by an edge must have different values

This constraint network has a particular restricted form:

- only binary constraints
- domains are finite

Crossword Puzzle

Problem instance:

- Variables: empty squares in a crossword puzzle;
- Domains: letters $\{A, B, C, \ldots, Z\}$ for each variable;
- Constraints: relations defined by a given set of words that need (or are allowed) to occur in the completed puzzle.

1	2	3	4	5	6	7	8
9		10		11	12		13
14	15	16	17	18		19	20
	21	22		23	24	25	

Fill-in words: EIER, HOLZ, IE, IM, IT, NZ, ON, RAM, RE, ROLLE, ROT, ZAR, ZUHOERER

Crossword Puzzle

Problem instance:

- Variables: empty squares in a crossword puzzle;
- Domains: letters $\{A, B, C, \ldots, Z\}$ for each variable;
- Constraints: relations defined by a given set of words that need (or are allowed) to occur in the completed puzzle.

Z	U	H	O	E	R	E	R
A		O		I	E		A
R	O	L	L	E		I	M
	N	Z		R	O	T	

Fill-in words: EIER, HOLZ, IE, IM, IT, NZ, ON, RAM, RE, ROLLE, ROT, ZAR, ZUHOERER

Boolean Satisfiability

Problem instance (Boolean constraint network):

- Variables: (propositional) variables;
- Domains: truth values $\{0, 1\}$ for each variable;
- Constraints: defined by a propositional formula in these variables.

Example: $(x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee x_4)$

SAT as a constraint satisfaction problem:

Given an arbitrary Boolean constraint network, is the network solvable?

Real World Applications

In practice, not only constraint satisfaction, but constraint optimization is required.

Seminar topic assignment

- Given n students who want to participate in a seminar; m topics are available to be worked on by students; each topic can be worked on by at most one student, and each student has preferences which topics s/he would like to work on;
- . . . how to assign topics to students?

Real World Applications

CSP/COP techniques can be used in

- *civil engineering* (design of power plants, water and energy supply, transportation and traffic infrastructure)
- *mechanical engineering* (design of machines, robots, vehicles)
- digital circuit *verification*
- automated timetabling
- air traffic control
- finance

Computational Complexity

Theorem

It is NP-hard to decide solvability of CSPs.

Since k-colorability (SAT, 3SAT) is NP-complete, solvability of CSPs in general must be NP-hard.

Question: Is CSP solvability *in* NP?

Computational Complexity

Theorem

It is NP-hard to decide solvability of CSPs.

Since k-colorability (SAT, 3SAT) is NP-complete, solvability of CSPs in general must be NP-hard.

Question: Is CSP solvability *in* NP?

Computational Complexity

Theorem

It is NP-hard to decide solvability of CSPs.

Since k-colorability (SAT, 3SAT) is NP-complete, solvability of CSPs in general must be NP-hard.

Question: Is CSP solvability *in* NP?

Solving CSPs

- Enumeration of all assignments and testing

⇝ . . . too costly

- Backtracking search

⇝ numerous different strategies, often "dead" search paths are explored extensively

- Constraint propagation: elimination of obviously impossible values
- Interleaving backtracking and constraint propagation: constraint propagation at each generated search node
- Many other search methods, e. g., local/stochastic search, etc.

Solving CSPs

- Enumeration of all assignments and testing

⇝ . . . too costly

- Backtracking search

⇝ numerous different strategies, often “dead” search paths are explored extensively

- Constraint propagation: elimination of obviously impossible values
- Interleaving backtracking and constraint propagation: constraint propagation at each generated search node
- Many other search methods, e. g., local/stochastic search, etc.

Solving CSPs

- Enumeration of all assignments and testing

⇝ . . . too costly

- Backtracking search

⇝ numerous different strategies, often "dead" search paths are explored extensively

- Constraint propagation: elimination of obviously impossible values
- Interleaving backtracking and constraint propagation: constraint propagation at each generated search node
- Many other search methods, e. g., local/stochastic search, etc.

Solving CSPs

- Enumeration of all assignments and testing

$\rightsquigarrow$ ... too costly

- Backtracking search

$\rightsquigarrow$ numerous different strategies, often “dead” search paths are explored extensively

- Constraint propagation: elimination of obviously impossible values
- Interleaving backtracking and constraint propagation: constraint propagation at each generated search node
- Many other search methods, e. g., local/stochastic search, etc.

Solving CSPs

- Enumeration of all assignments and testing

⇝ . . . too costly

- Backtracking search

⇝ numerous different strategies, often “dead” search paths are explored extensively

- Constraint propagation: elimination of obviously impossible values
- Interleaving backtracking and constraint propagation: constraint propagation at each generated search node
- Many other search methods, e. g., local/stochastic search, etc.

Contents I

- *Introduction and mathematical background*
  - Sets, relations, graphs
  - Constraint networks and satisfiability
  - Binary constraint networks
  - Simple solution methods (backtracking, etc.)
- *Inference-based methods*
  - Arc and path consistency
  - k-consistency and global consistency
- *Search methods*
  - Backtracking
  - Backjumping
  - Comparing different methods
  - Stochastic local search

Contents I

- *Introduction and mathematical background*
  - Sets, relations, graphs
  - Constraint networks and satisfiability
  - Binary constraint networks
  - Simple solution methods (backtracking, etc.)
- *Inference-based methods*
  - Arc and path consistency
  - k-consistency and global consistency
- *Search methods*
  - Backtracking
  - Backjumping
  - Comparing different methods
  - Stochastic local search

Contents I

- *Introduction and mathematical background*
  - Sets, relations, graphs
  - Constraint networks and satisfiability
  - Binary constraint networks
  - Simple solution methods (backtracking, etc.)
- *Inference-based methods*
  - Arc and path consistency
  - k-consistency and global consistency
- *Search methods*
  - Backtracking
  - Backjumping
  - Comparing different methods
  - Stochastic local search

Contents II

- *Global constraints*
- *Constraint optimization*
- *Selected advanced topics*
  - Expressiveness vs complexity of constraint formalisms
  - Qualitative constraint networks

Contents II

- *Global constraints*
- *Constraint optimization*
- *Selected advanced topics*
  - Expressiveness vs complexity of constraint formalisms
  - Qualitative constraint networks

Contents II

- *Global constraints*
- *Constraint optimization*
- *Selected advanced topics*
  - Expressiveness vs complexity of constraint formalisms
  - Qualitative constraint networks

Organization

Lectures: Where, When, Web Page

Where

Bld. 101, Room 00-036

When

Monday, 16:15–18:00
Wednesday, 16:15–17:00 (+ exercises: 17:15–18:00)

No lectures

- 14-05-2012
- 16-05-2012
- 28-05-2012 (Pentecost break)
- 30-05-2012 (Pentecost break)

Web Page

`http://www.informatik.uni-freiburg.de/~ki/teaching/ss12/csp/`

Lecturers

Prof. Bernhard Nebel

Bld. 52, Room 00-029

Consultation: Wednesday, 14-15

Phone: 0761/203-8221

Email: nebel@informatik.uni-freiburg.de

Dr. Julien Hué

Bld. 52, Room 00-041

Phone: 0761/203-8234

Email: hue@informatik.uni-freiburg.de

Dr. Stefan Wölfl

Bld. 52, Room 00-043

Phone: 0761/203-8228

Email: woelfl@informatik.uni-freiburg.de

Exercises

Where

Bld. 101, Room 00-036

When

Wednesday, 17:15–18:00

Who

Matthias Westphal
Bld. 52, Room 00-041
Phone: 0761/203-8227
Email: westpham@informatik.uni-freiburg.de

Course Prerequisites & Goals

Goals

- Acquiring skills in constraint processing
- Understanding the principles behind different solving techniques
- Being able to read and understand research literature in the area of constraint satisfaction
- Being able to complete a project (thesis) in this research area

Prerequisites

- Basic knowledge in the area of AI
- Basic knowledge in formal logic
- Basic knowledge in theoretical computer science

Exercises

Exercise assignments

- handed out on Wednesdays
- due on Wednesday in the following week (before the lecture)
- may be solved in groups of **two** students
- 50 % of reachable points are required for exam admission

Programming project

Implement a CSP solver . . .

- Implementation tasks are specified on a regular basis (depending on the progress of the lecture)
- May be worked on in groups of **two** students
- Programming language
- Implementation should compile on a standard Linux computer (Ubuntu 11.10)
- We provide git repositories for source code
- Working solver is prerequisite for exam admission
- Will do a competition between solvers at the end of the lecture

Examination

Credit points

- 6 ECTS points

Exams

- (Oral or written) exam in September 2012

Acknowledgements

Lecture is based on slidesets of previous CSP lectures:

- Malte Helmert and Stefan Wölfl (summer term 2007)
- Bernhard Nebel and Stefan Wölfl (winter term 2009/10)

Special thanks to: Matthias Westphal, Robert Mattmüller, Gabriele Röger, Manuel Bodirsky

Literature

- Rina Dechter:
  Constraint Processing,
  Morgan Kaufmann, 2003.
- Francesca Rossi, Peter van Beek, and Toby Walsh:
  Handbook of Constraint Programming,
  Elsevier, 2006.
- Wikipedia contributors:
  Wikipedia, The Free Encyclopedia,
  `http://en.wikipedia.org/`
- Wolfram Research:
  Wolfram MathWorld,
  `http://mathworld.wolfram.com/`

- Further readings will be given during the lecture.