Constraint Satisfaction Problems: Backtracking Search and Arc Consistency

Alice Gao

Lecture 5

Readings: RN 6.1 - 6.3. PM 4.1 - 4.4.

Outline

Learning Goals

By the end of the lecture, you should be able to

- Formulate a real-world problem as a constraint satisfaction problem.
- Trace the execution of the backtracking search algorithm.
- Verify whether a constraint is arc-consistent.
- Trace the execution of the AC-3 arc consistency algorithm.
- Trace the execution of the backtracking search algorithm with arc consistency.
- Trace the execution of the backtracking search algorithm with arc consistency and with heuristics for choosing variables and values.

Learning Goals

Examples of CSP Problems

Introduction to CSPs

Formulating a Problem as a CSP

Solving a CSP

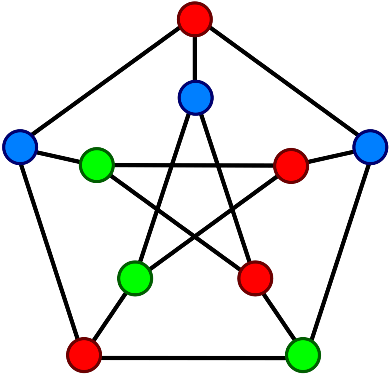
Revisiting the Learning goals

Real-World CSP Problems

- Disaster Recovery (Pascal Van Hentenryck)
  http://videolectures.net/icaps2011_van_hentenryck_disaster/
- Transportation Planning (Pascal Van Hentenryck)
  https://www.youtube.com/watch?v=SxvM0jG3qLA
- Air Traffic Control
  https://doi.org/10.1016/S1571-0661(04)80797-7
  https://doi.org/10.1017/S0269888912000215
- Factory process management
- Scheduling (courses, meetings, etc)

Example: Crossword Puzzles

Example: Graph Coloring Problem



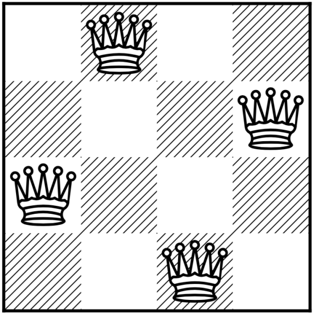
Applications:

- Designing seating plans
- Exam scheduling
- ...

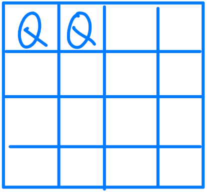
Example: Sudoku

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

Example: 4-Queens Problem



Internal Structure of States



- Search algorithms are unaware of the internal structure of states.

  treats each state as a black box.
  - generate successors.
  - test whether it's a goal.

- However, knowing a state's internal structure can help.

  search alg so far: this is not a goal state,
  let's add more queens.

  a smarter alg: this is a dead end,
  let's backtrack immediately.

can prune search tree & make search more efficient!

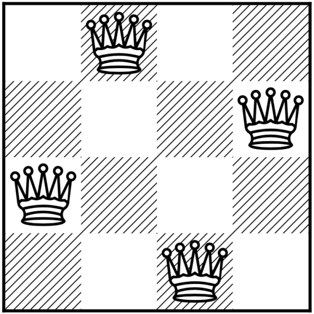
Defining a CSP

Each state contains

- A set X of variables: $\{X_1, X_2, ..., X_n\}$.
- A set D of domains: D_i is the domain for variable X_i, $\forall i$.
- A set C of constraints specifying allowable value combinations.

A solution is an assignment of values to all the variables that satisfy all the constraints.

Example: 4-Queens Problem



The 4-Queens Problem as a CSP

Variables: x_0, x_1, x_2, x_3.

x_i is the row position of the queen in column i where $i \in \{0, 1, 2, 3\}$

x_0	x_1	x_2	x_3	
				0
				1
				2
				3

(Assume that exactly one queen is in each column.)

Domains: $D_i = \{0, 1, 2, 3\}$ for each x_i.

Constraints: No two queens can be in the same row or in the same diagonal.

$$\forall i \, \forall j \, (i \neq j) \rightarrow \left((x_i \neq x_j) \wedge (|x_i - x_j| \neq |i - j|) \right)$$

for example: $(x_0 \neq x_1) \wedge (|x_0 - x_1| \neq 1)$

Backtracking Search for 4-Queens Problem

assumptions: ① place queens from left to right.
② always ensure that constraints are satisfied.

- State: one queen per column in the leftmost k columns with no pair of queens attacking each other.
- Initial state: no queens on the board. _ _ _ _
- Goal state: 4 queens on the board. No pair of queens are attacking each other. 2031, 1302.
- Successor function: add a queen to the leftmost empty column such that it is not attacked by any other existing queen.

states:

_ _ _ _

2 _ _ _

2 0 _ _

2 0 3 _

2 0 3 1

0	Q	x	x	x
1		x	x	
2		Q	x	x
3			x	x

successors:

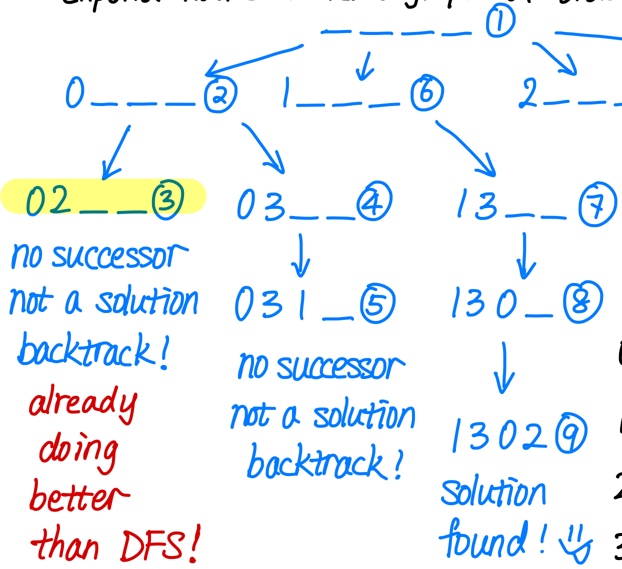
0 _ _ _

0 2 _ _

0 3 _ _

Trace Backtracking Search for 4-Queens Problem

- expand nodes in lexicographical order.



	x_0	x_1	x_2	x_3
0	X	X	Q	X
1	Q	X	X	X
2	X	X	X	Q
3	X	Q	X	X

The Idea of Arc Consistency

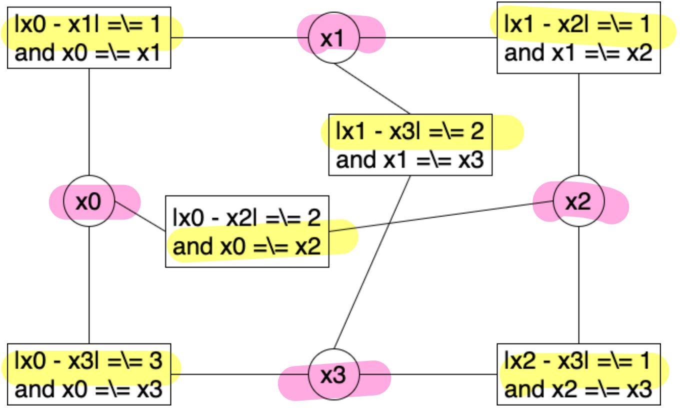
Start with $x_0 = 0$.

- $x_2 = 1$ does not lead to a solution. Why? no position for x_3.
- $x_2 = 3$ also does not lead to a solution. Why? no position for x_1.

	x_0	x_1	x_2	x_3
0	Q	X	X	X
1	X	X		
2	X	X	X	X
3	X	X	Q	X

4-Queens Constraint Network

only necessary to include binary constraints?
(1) why not unary constraints? (2) why not constraints involving more variables?



Definition of Arc Consistency

$\forall v \in D_X \ \exists w \in D_Y \ (v, w)$ satisfies $c(X, Y)$.

Definition (Arc Consistency)

An arc $\langle X, c(X, Y)\rangle$ is arc-consistent if and only if for every value v in D_X, there is a value w in D_Y such that (v, w) satisfies the constraint $c(X, Y)$.

EX1: $D_X = \{1, 2\}$ $D_Y = \{1, 2, 3\}$

EX2: $D_X = \{1, 2\}$ removing 2 from Dx does not rule out a solution! $D_Y = \{1, 2\}$?

If $\langle X, c(X,Y)\rangle$ is NOT arc-consistent, we can reduce the domain of X. This will not rule out any solution!

CQ: Definition of Arc Consistency

CQ: Consider the constraint "X is divisible by Y" between two variables X and Y. The arc $\langle X, c(X, Y) \rangle$ is arc-consistent in how many of the four scenarios below?

1. $D_X = \{10, 12\}, D_Y = \{3, 5\}$
2. $D_X = \{10, 12\}, D_Y = \{2\}$ → for different values in D_X, can choose the same value in D_Y.
3. $D_X = \{10, 12\}, D_Y = \{3\}$?
4. $D_X = \{10, 12\}, D_Y = \{3, 5, 8\}$ → fine to have a value in D_Y that we never use.

(A) 0 (B) 1 (C) 2 (D) 3 (E) 4

CQ: Is Arc-Consistency Symmetric? No!

Note: When we consider $\langle Y, c(X,Y)\rangle$, the constraint is still the same! Don't change it!

Treat $\langle X, c(X,Y)\rangle$ and $\langle Y, c(X,Y)\rangle$ as 2 separate things!

CQ: True or False:
Let c be a binary constraint between X and Y.
If $\langle X, c(X, Y)\rangle$ is arc-consistent, then $\langle Y, c(X, Y)\rangle$ is arc-consistent.

(A) True

(B) False

(C) Not enough information to tell

X — $\langle X, c(X,Y)\rangle$ — $\boxed{X \text{ is divisible by } Y}$ — $\langle Y, c(X,Y)\rangle$ — Y

$D_X = \{10, 12\}$? arc-consistent!

NOT arc-consistent! $D_Y = \{3, 5, 8\}$

No value in D_X is divisible by $8 \in D_Y$.

The AC-3 Arc Consistency Algorithm

Algorithm 1 The AC-3 Algorithm

1: put every arc in the set S. $\langle X, c(X,Y)\rangle$ $\langle Y, c(X,Y)\rangle$
2: **while** S is not empty **do**
3: select and remove $\langle X, c(X,Y)\rangle$ from S (order doesn't matter)
4: remove every value in D_X that doesn't have a value in D_Y that satisfies the constraint $c(X,Y)$ (if not A-C, reduce Dx.)
5: **if** D_X was reduced **then**
6: **if** D_X is empty **then return** false no solution exists!
7: for every $Z \neq Y$, add $\langle Z, c'(Z,X)\rangle$ to S
return true

Why do we need line 7? After reducing the domain Dx, we need to re-check every arc where X is the second variable since it may not be consistent anymore.

CQ: Effect of Removing a Value on Arc Consistency

Counterexample: X — $\boxed{X < Y}$ — Y

$D_X = \{1, 2\}$ A-C! $D_Y = \{2, 3\}$

$D_X = \{1, 2\}$ NOT A-C! $D_Y' = \{2\}$

CQ: True or False:
Let c be a binary constraint between X and Y.
If $\langle X, c(X, Y)\rangle$ is arc-consistent, then,
after removing a value from D_Y, $\langle X, c(X, Y)\rangle$ is still arc-consistent.

(A) True

(B) False

(C) Not enough information to tell

After reducing the domain of the second variable, the arc may no longer be arc-consistent.

We may need to re-visit the constraint.

CQ: Effect of Removing a Value on Arc Consistency

CQ: True or False:
Let c be a binary constraint between X and Y.
If $\langle X, c(X, Y)\rangle$ is arc-consistent, then,
after removing a value from D_X, $\langle X, c(X, Y)\rangle$ is still arc-consistent.

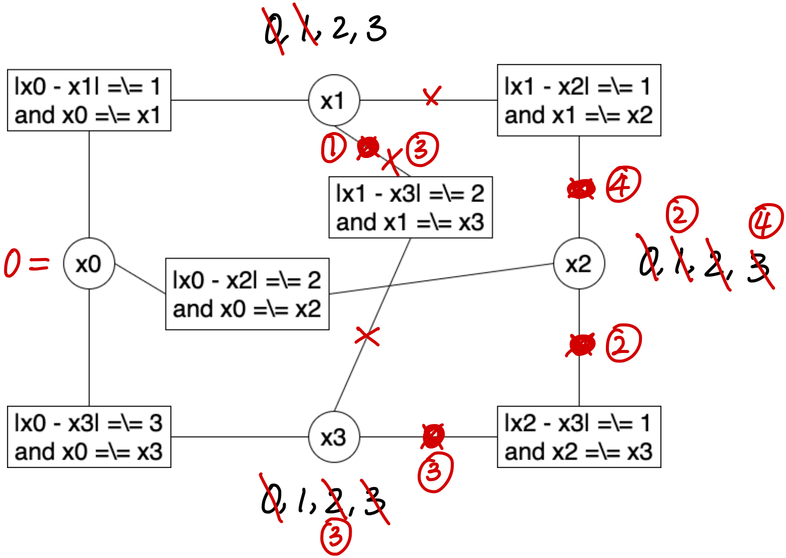
(A) True

(B) False

(C) Not enough information to tell

After reducing the domain of the first variable,
the arc is still arc-consistent.
There is no need to revisit the constraint.

Trace Arc Consistency



Properties of the AC-3 Algorithm

- Does the order in which arcs are considered matter?

  No. Any order will lead to the same solution.

- Three possible outcomes of the arc consistency algorithm:

  ① a domain is empty. No solution exists!

  ② every domain has 1 value left. A unique soln!

  ③ every domain has ≥ 1 value left and ≥ 1 domain has multiple values left. Need to do search or split domains.

- Is AC-3 guaranteed to terminate? Yes.

What is the complexity of the AC-3 algorithm?

CSP has n variables. size of each domain $\leq d$.
c binary constraints. Each arc can be added to the set d times (d values to delete). A-C checked in $O(d^2)$ time. $O(c * d * d^2) = O(cd^3)$.

Revisiting the Learning Goals

By the end of the lecture, you should be able to

- Formulate a real-world problem as a constraint satisfaction problem.
- Trace the execution of the backtracking search algorithm.
- Verify whether a constraint is arc-consistent.
- Trace the execution of the AC-3 arc consistency algorithm.
- Trace the execution of the backtracking search algorithm with arc consistency.
- Trace the execution of the backtracking search algorithm with arc consistency and with heuristics for choosing variables and values.