Introduction to Artificial Intelligence

Course 16 :198 :440

Recitation 6:

Constraint Satisfaction Problems (CSPs)

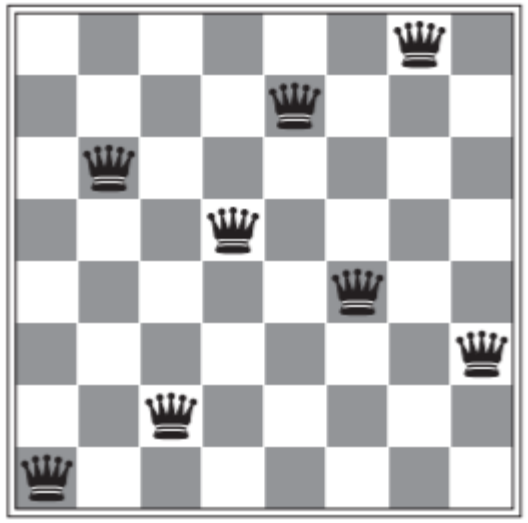
RUTGERS

Constraint Satisfaction Problems

Within the larger class of optimization problems[1], we now consider **constraint-satisfaction problems**.

These are characterized by:

- **Variables** (x_1, x_2, …, x_n)
- **Constraints** (i.e. $x_i \neq x_{i+1}$)
  (can take many forms)



Formulate this problem:

1. What are your **variables**?
2. What are your **constraints**?

[1] We have seen the 8 queens puzzle previously as an optimization problem

CSP Example – 4x4 Sudoku

4	3	2	1
2	1	4	3
1	2	3	4
3	4	1	2

SOLUTION

Sudoku rules:

Objective: Fill in a value (1,2,3,4) in each cell, subject to:

1. Each row cannot have a duplicate
2. Each column cannot have any duplicates
3. Each square (here 2x2) cannot have any duplicates

A. Define the variables:

$x_{ij} \in \{1,2,3,4\}$

$\forall i,j$ in $\{1,2,3,4\}$

There are other (i.e. implicit) ways of describing this constraint as well

B. Formulate the given constraints:

1. $x_{i1} \neq x_{i2} \neq x_{i3} \neq x_{i4} \mid \forall i$ in (1,2,3,4)
2. $x_{1j} \neq x_{2j} \neq x_{3j} \neq x_{4j} \mid \forall j$ in (1,2,3,4)
3. $(x_{11} \neq x_{12} \neq x_{21} \neq x_{22}) \cap (x_{13} \neq x_{14} \neq x_{23} \neq x_{24}) \cap \ldots$

C. **Solve!**

CSP Example – Australia

Map coloring rules:

Objective: Assign a color (R,G,B) to each region, subject to:

1. No adjacent regions (sharing a side) may have the same color

A. Define the variables:

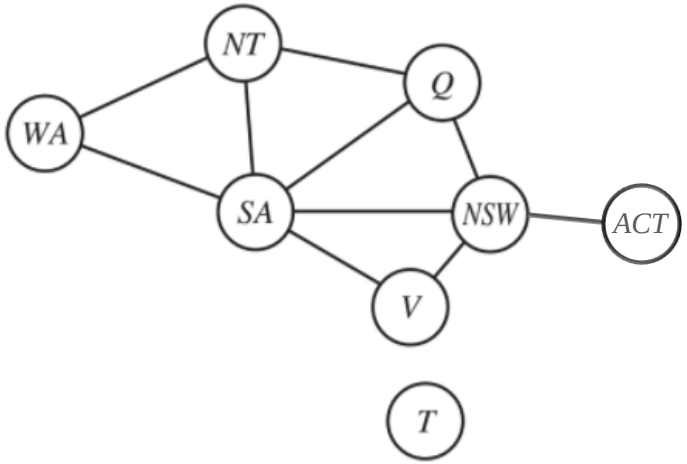
$X_i \in \{R,G,B\}$

B. Formulate the given constraints:

WA ≠ NT ≠ SA … and so on…

Or more simply: X_i *not* $\in$ *neighbors*(X_i)

C. **Solve!**



CSP Example – Australia

What happened in this search?

	WA	*NT*	*Q*	*NSW*	*V*	*SA*	*T*
Initial domains	R G B	R G B	R G B	R G B	R G B	R G B	R G B
After *WA=red*	Ⓡ	G B	R G B	R G B	R G B	G B	R G B
After *Q=green*	Ⓡ	B	Ⓖ	R B	R G B	B	R G B
After *V=blue*	Ⓡ	B	Ⓖ	R	Ⓑ		R G B

What will a DFS algorithm do next?

Consider…

1.1 Incremental Formulation

Initial state: The empty assignment in which all variables are unassigned
Successor function: A value is assigned to a variable, assuming it doesn't violate any constraints. The approach retains consistent or valid assignments throughout the problem.
Goal test: Is the assignment complete?
Path cost: 1 for every step.

An example of the incremental formulation is the 8-queens problem, when we place each queen one at a time to the left most column while making sure it's placement doesn't cause any existing queen to be attacked. The incremental formulation lends itself to a solution by classical search (uninformed and informed).

With *d* values and *n* variables, my search tree becomes **$O(d^n)$** (!)

How can we improve our search results, if we consider:
(think heuristics)

1. The variable ordering
2. The value ordering