Artificial Intelligence

6. CSP

Shashi Prabh

School of Engineering and Applied Science
Ahmedabad University

Contents

Goal: use factored representation of agents to solve problems.

Topics

- Constraint Satisfaction Problem
- Constraint Propagation
- Backtracking Search
- Local Search

Constraint Satisfaction Problems (CSP)

- We consider factored representation of states
  - A state is a set of variables
- A problem solution is an assignment of values to the state variables where all the constraints on the variables are satisfied

Constraint Satisfaction Problems (CSP)

- Why CSP?
  - CSP is a natural formulation in many problems
    - Scheduling, planning, resource allocation, temporal models, control etc.
  - Significant reduction of search space, availability of fast solvers
  - Insight into the problem structure can be used for search speed-up
    - Some intractable atomic search-space problems can be quickly solved as CSP formulation
  - Actions and transition model can be deduced from the formulation

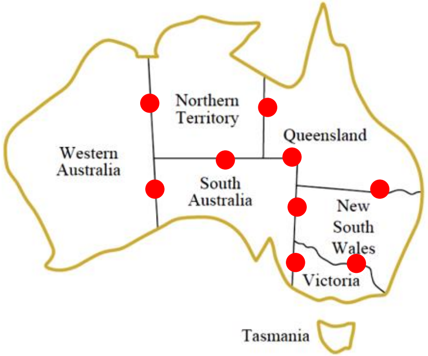
Constraint Satisfaction Problems (CSP)

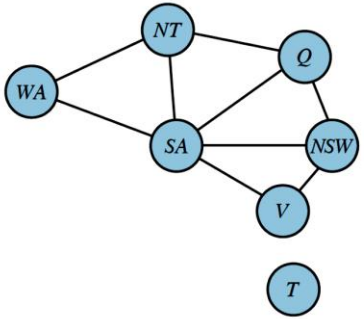
- A CSP consists of three components (X, D, C):
- Variables $X = \{ x_1, x_2, \ldots, x_n\}$
- Domains $D = \{D_1, D_2, \ldots, D_n\}$
- Constraints $C = \{c_1, c_2, \ldots, c_m\}$
  - Domain D_i consists of the set of allowable values $\{v_1, \ldots, v_k\}$ for each x_i
    - {T, F} for a Boolean variable
  - Constraint c_j consists of a pair ⟨scope, relation⟩
    - $\langle(x_1, x_2), x_1 \neq x_2\rangle$ or just $x_1 \neq x_2$

Constraint Satisfaction Problems (CSP)

- Goal: Assign values to the variables from their respective domains such that all the constraints are satisfied
  - An assignment that does not violate any constraint is called consistent or legal assignment
  - A solution to a CSP is a complete and consistent assignment
  - Solving a CSP is NP-complete in general
- Types of constraints:
  - Unary, binary, global
  - $\langle (x_1), x_1 \neq 4 \rangle$, $\langle (x_1, x_2), x_1 \neq x_2 \rangle$, AllDiff, AtMost, AtLeast

Map coloring





- X = {W, N, S, Q, NSW, V, T}
- D = {r, g, b}
- C = { W ≠ N, S ≠ N, Q ≠ N, W ≠ S, S ≠ Q, etc}
  - W ≠ N means {(r, g), (r, g), (g, r), (g, b), (b, r), (b, g)}
  - Note the reduced search space due to the constraints: 2^5 vs 3^5
  - Can you find one solution?
- In a CSP constraint graph, two variables are connected by an edge if there is a constraint that involves both

Job-Shop Scheduling – Car Assembly

- X is the set of tasks

$$\{Axle_F, Axle_B, Wheel_{RF}, Wheel_{LF}, Wheel_{RB}, Wheel_{LB}, Nuts_{RF}, Nuts_{LF}, Nuts_{RB}, Nuts_{LB}, Cap_{RF}, Cap_{LF}, Cap_{RB}, Cap_{LB}, Inspect\}$$

- Values are the start times of tasks: $D_i = \{0, 1, \ldots, 30\}$
- Constraints: precedence constraints, completion times
  - If T_i preceeds T_j, $T_i + D_i \leq T_j$

$$Axle_F + 10 \leq Wheel_{RF}; \quad Axle_F + 10 \leq Wheel_{LF};$$
$$Axle_B + 10 \leq Wheel_{RB}; \quad Axle_B + 10 \leq Wheel_{LB}.$$

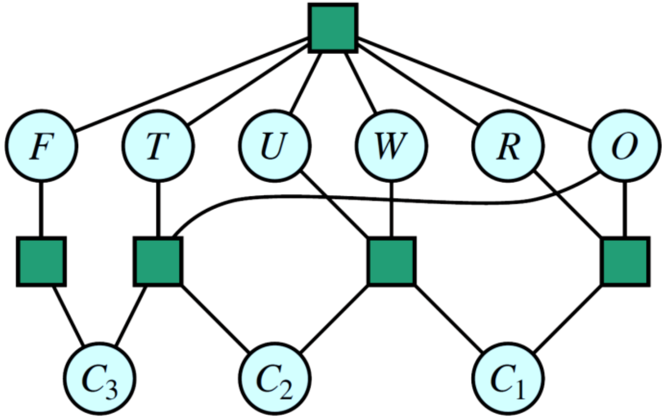
  - Disjunctions: Axle installations must not overlap in time

$$(Axle_F + 10 \leq Axle_B) \quad \textbf{or} \quad (Axle_B + 10 \leq Axle_F)$$

  - Exercise: CSP formulation of 8-Queens problem

Cryptarithmetic Puzzles

$$\begin{array}{cccc} & T & W & O \\ + & T & W & O \\ \hline F & O & U & R \end{array}$$



Constraint hypergraph

- Constraints: AllDiff (F, T, U, W, R, O), F ≠ 0 and

$$O + O = R + 10 \cdot C_1$$
$$C_1 + W + W = U + 10 \cdot C_2$$
$$C_2 + T + T = O + 10 \cdot C_3$$
$$C_3 = F\,,$$

1. In the context of a Constraint Satisfaction Problem (CSP), what are the three main components?
   A. Variables, Functions, and Relations
   B. Nodes, Arcs, and Paths
   C. States, Transitions, and Goals
   D. Variables, Domains, and Constraints

2. Which of the following is an example of a binary constraint?
   A. The sum of variables A, B, and C must be less than 10.
   B. The value of variable A must be different from the value of variable B.
   C. The colors of all bordering regions must be different.
   D. The value of variable A must be even.

1. In the context of a Constraint Satisfaction Problem (CSP), what are the three main components?
   A. Variables, Functions, and Relations
   B. Nodes, Arcs, and Paths
   C. States, Transitions, and Goals
   D. Variables, Domains, and Constraints

2. Which of the following is an example of a binary constraint?
   A. The sum of variables A, B, and C must be less than 10.
   B. The value of variable A must be different from the value of variable B.
   C. The colors of all bordering regions must be different.
   D. The value of variable A must be even.

3. In the N-queens problem, what do the variables, domains, and constraints represent?
   A. Variables are the queens, domains are the squares on the board, and constraints are that no two queens can attack each other.
   B. Variables are the rows and columns, domains are whether a queen is present, and constraints are that a queen must be in every row.
   C. Variables are the rows, domains are the columns, and constraints are that queens cannot be in the same row.
   D. Variables are the columns, domains are the rows, and constraints are that no two queens can share a row, column, or diagonal.

4. Which of the following problems can be formulated as a CSP?
   A. Generating a creative story given a set of keywords.
   B. Calculating the final grade for a student.
   C. Determining a weekly schedule for classes without any conflicts.
   D. Finding the shortest path between two cities on a map.

1. In the N-queens problem, what do the variables, domains, and constraints represent?
   A. Variables are the queens, domains are the squares on the board, and constraints are that no two queens can attack each other.
   B. Variables are the rows and columns, domains are whether a queen is present, and constraints are that a queen must be in every row.
   C. Variables are the rows, domains are the columns, and constraints are that queens cannot be in the same row.
   D. Variables are the columns, domains are the rows, and constraints are that no two queens can share a row, column, or diagonal.

2. Which of the following problems can be formulated as a CSP?
   A. Generating a creative story given a set of keywords.
   B. Calculating the final grade for a student.
   C. Determining a weekly schedule for classes without any conflicts.
   D. Finding the shortest path between two cities on a map.

Constraint Propagation

- A CSP algorithm can generate successors as new assignments
- Constraint propagation is an alternative where using constraints the number of legal values is reduced
- Used along-with search and/or as a preprocessing step

Constraint Propagation

- Local consistency shrinks the search space by eliminating the inconsistent assignments
- Types of local consistency
  - Node consistency
  - Arc consistency
    - Path and K-Consistency
- Global constraints, bounds propagation

Node Consistency

- A node in the constraint graph is node-consistent if all the values in the variable's domain satisfy the variable's unary constraints.
- Example: consider a unary constraint SA ≠ {green}
  - The variable SA with initial domain {red, green, blue} can be made node consistent by eliminating green from its domain, leaving SA with the reduced domain {red, blue}.
- A graph is node-consistent if every variable in the graph is node-consistent.
  - One can just eliminate domain values inconsistent with unary constraints.

Arc Consistency

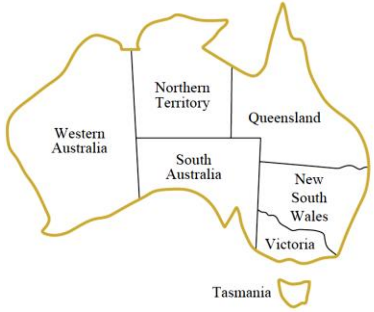
- A variable is arc-consistent if for every value in its domain, there is some value in the domains of all the variables connected by a binary constraint.
  - Example: consider the constraint $Y = X^2$, $D_X = \mathbb{N}$, $D_Y = \{ 1, 4, 9\}$
  - X is made arc-consistent with Y by restricting $D_X = \{ 1, 2, 3\}$
  - However, arc-consistency is ineffective in the map coloring example
- **AC-3** is a widely used arc-consistency algorithm

AC-3 (Mackworth, 1977)

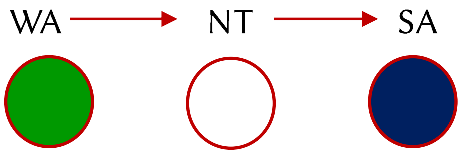
```
function AC-3(csp) returns false if an inconsistency is found and true otherwise
   queue ← a queue of arcs, initially all the arcs in csp

   while queue is not empty do
      (Xi, Xj) ← POP(queue)
      if REVISE(csp, Xi, Xj) then
         if size of Di = 0 then return false
         for each Xk in Xi.NEIGHBORS - {Xj} do
            add (Xk, Xi) to queue
   return true

function REVISE(csp, Xi, Xj) returns true iff we revise the domain of Xi
   revised ← false
   for each x in Di do
      if no value y in Dj allows (x,y) to satisfy the constraint between Xi and Xj then
         delete x from Di
         revised ← true
   return revised
```

- Initially, each binary constraint inserts two arcs
- X_i is being made consistent with X_j
- O (c d^3) worst-case complexity

Path Consistency



- AC does not help with map coloring
    - Does not object to 2-coloring the map
- $\{X_i, X_j\}$ is path-consistent with respect to a third variable X_m if, for every assignment $\{X_i = a, X_j = b\}$ consistent with the constraints on $\{X_i, X_j\}$, there is an assignment to X_m that satisfies the constraints on $\{X_i, X_m\}$ and $\{X_m, X_j\}$.
    - Refers to the overall consistency of the path $X_i \rightarrow X_m \rightarrow X_j$
    - Can infer that no valid 2-coloring of the Australia map exists



K-Consistency

- A CSP is k-consistent if, for any set of k−1 variables and for any consistent assignment to those variables, a consistent value can always be assigned to any k^{th} variable
  - 1-consistency says that, given the empty set, we can make any set of one variable consistent: this is what we called node consistency
  - 2-consistency is the same as arc consistency
  - 3-consistency (binary constraints) is the same as path consistency

K-Consistency

- A CSP is strongly k-consistent if it is k-consistent and is also (k–1)-consistent, (k–2)-consistent, . . ., 1-consistent
- Why?
  - Can design a greedy algorithm
  - CSP is NP-complete
    - K-consistency requires exponential time and space

Global constraints

- A global constraint involves an arbitrary number of variables. It is more efficient to handle these by special-purpose algorithms
  - AllDiff: if m variables are involved in an AllDiff constraint, and if n possible distinct values altogether are available, then the constraint cannot be satisfied if $m > n$
  - Atmost: resource constraint
    - Example: no more than 10 personnel are scheduled in total
    - We can detect an inconsistency simply by checking the sum of the minimum values of the current domains

Global constraints

- **Bounds propagation:** For problems with large integer domains it is usually not efficient to represent the domain of each variable as a large set of integers.
  - Domains can be represented by upper and lower bounds and managed by bounds propagation

Global constraints

- Example:
  - Consider two flights, F1 and F2, for which the planes have capacities 165 and 385, respectively
  - The initial domains for the numbers of passengers are then D1 = [0, 165] and D2 = [0, 385]
  - The additional constraint that the two flights together must carry 450 people can be handled by propagating bounds constraints as D1 = [65, 165] and D2 = [285, 385]

Sudoku

	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	8		6	4		
D			8	1		2	9		
E	7								8
F			6	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1		3		

	1	2	3	4	5	6	7	8	9
A	4	8	3	9	2	1	6	5	7
B	9	6	7	3	4	5	8	2	1
C	2	5	1	8	7	6	4	9	3
D	5	4	8	1	3	2	9	7	6
E	7	2	9	5	6	4	1	3	8
F	1	3	6	7	9	8	2	4	5
G	3	7	2	6	8	9	5	1	4
H	8	1	4	2	5	3	7	6	9
I	6	9	5	4	1	7	3	8	2

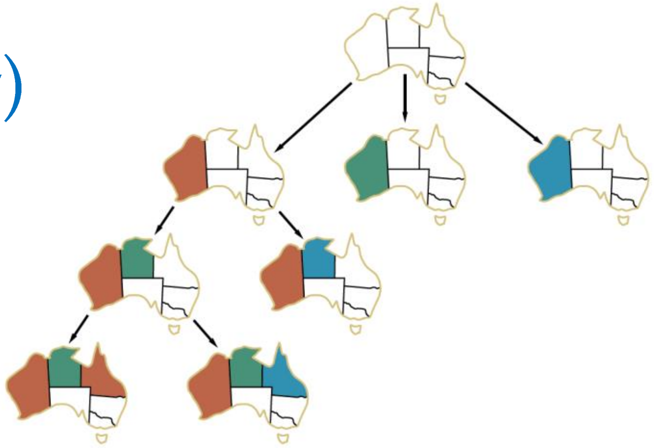
Exercise: Write CSP formulation!

Backtracking Search

- DFS at heart but with significant differences
- Search for solution is needed when after constraint propagation there exist variables with ≥ 1 values
- For a CSP with n variables of domain size d, a pure DFS results in a search tree with $n!\, d^n$ leaf nodes at depth n
  - The branching factor at the top would be nd, at the next level (n-1) d and so on
- The order of assignments does not matter
  - There are only d^n possible assignments!

Backtracking Search

- Backtracking search progresses via a recursive call
- An unassigned variable is (repeatedly) chosen, a value is assigned and the search progresses to another variable
  - If the search succeeds, the solution is returned
  - If the search fails, the assignment is restored to the previous state, and the next value is tried



Backtracking Search

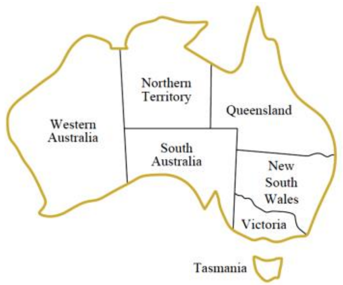
```
function BACKTRACKING-SEARCH(csp) returns a solution or failure
  return BACKTRACK(csp, { })

function BACKTRACK(csp, assignment) returns a solution or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(csp, assignment)
  for each value in ORDER-DOMAIN-VALUES(csp, var, assignment) do
      if value is consistent with assignment then
         add {var = value} to assignment
         inferences ← INFERENCE(csp, var, assignment)
         if inferences ≠ failure then
            add inferences to csp
            result ← BACKTRACK(csp, assignment)
            if result ≠ failure then return result
            remove inferences from csp
         remove {var = value} from assignment
  return failure
```

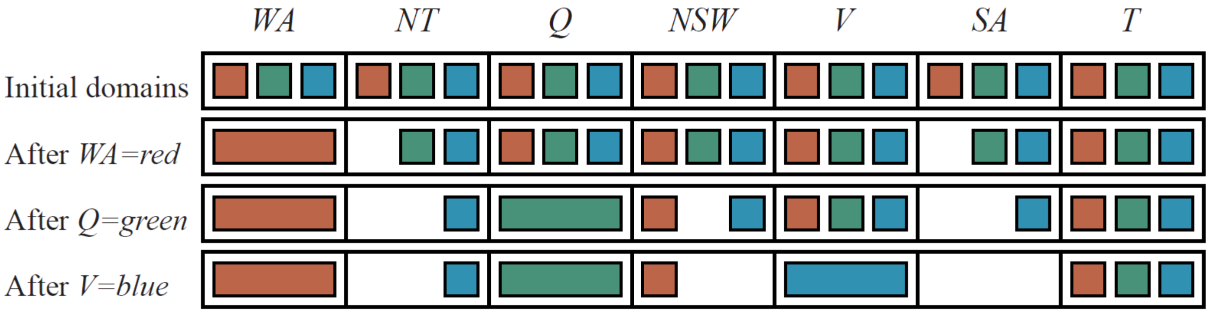
Improving Backtracking Search

- Backtracking search can be improved using domain-independent heuristics that take advantage of the factored representation of states
- Variable and value ordering heuristics
  - Minimum-remaining-values heuristic (MRV)
    - Start with F in crypatrithmetic puzzle
  - Degree heuristic – largest first
    - Start with SA in Australia map
  - Least constraining value first heuristic (LCV)
    - Values that rule out the fewest choices first

Forward Checking

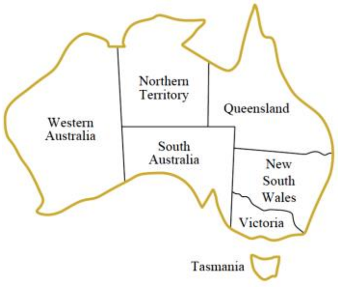
- Forward Checking: Check for arc consistancy upon a variable assignment
  - Upon assignment to X, make each unassigned variable Y that is connected to X by a constraint, arc-consistent with X
    - After assigning V =blue, forward checking finds domain of SA empty
      - ⟹Backtrack

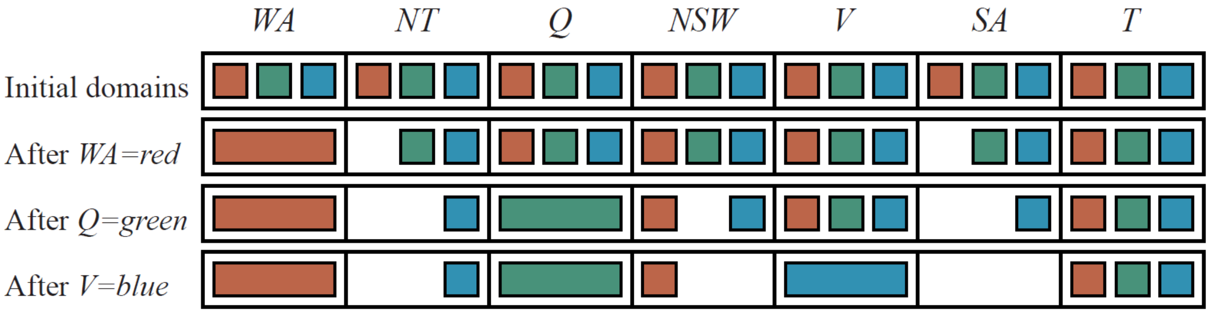




Interleaved Search and Inference

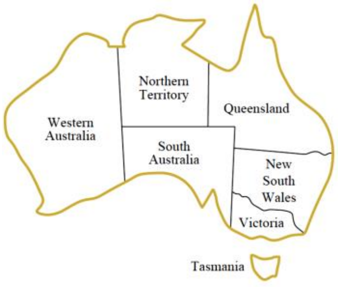
- Combining MRV with forward checking is more effective
  - After assigning {WA=red} NT and SA have two values.
  - MRV would have chosen one of them first leading to a solution
- Forward checking incrementally computes the information that the MRV heuristic needs…

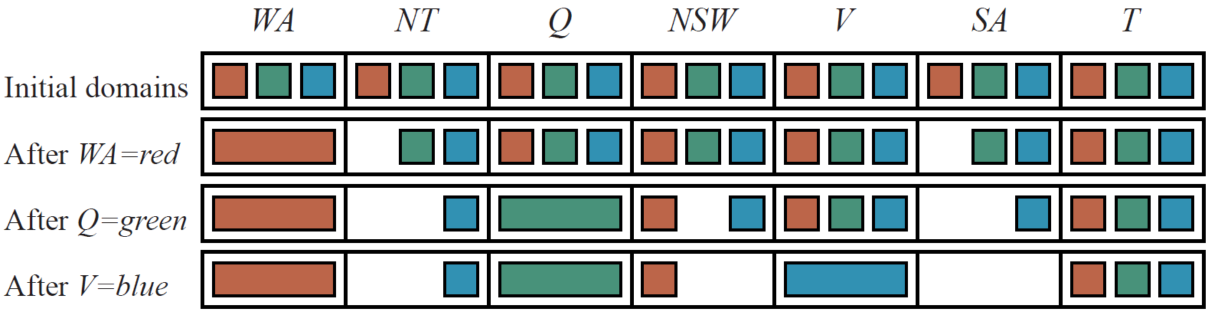




Interleaved Search and Inference

- Forward checking doesn't detect all inconsistencies since it does not look ahead far enough
  - In the Q=green row, both NT and SA are left with blue as their only possible value, which is an inconsistency, since they are neighbors.





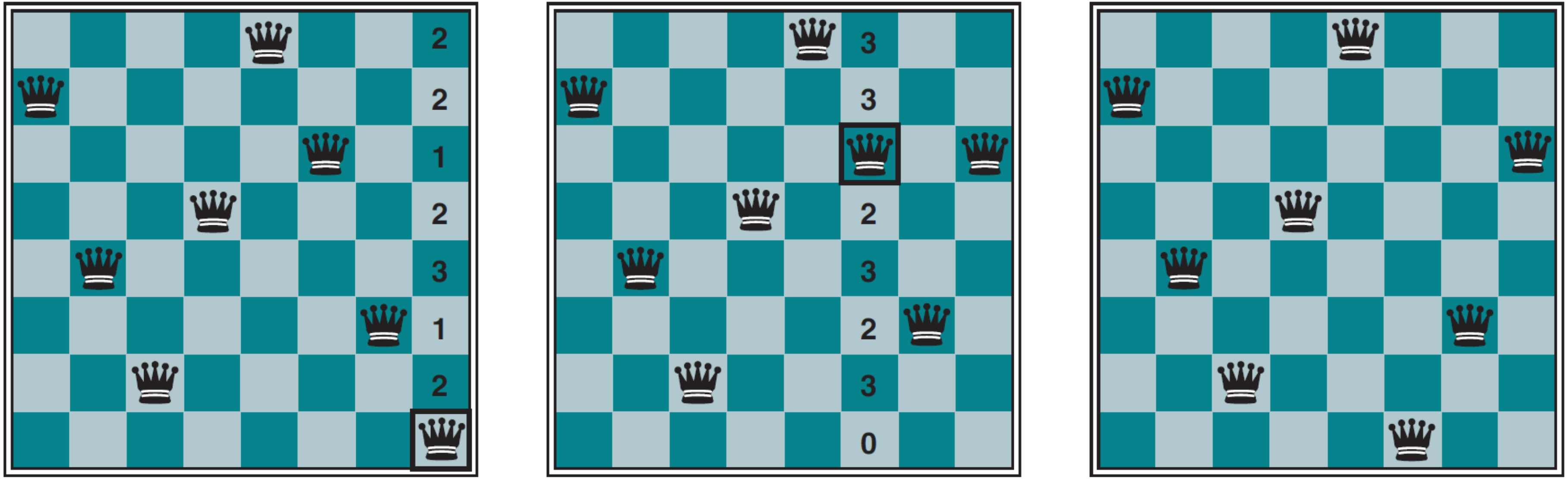
Maintaining Arc Consistency (MAC)

- After a variable X_i is assigned a value, inference calls AC-3
  - Instead of a queue of all the arcs, it starts with only the arcs (X_j, X_i) for all X_j that are unassigned and are neighbors of X_i.
  - If any variable's domain is reduced to the empty set, the call to AC-3 fails which triggers backtracking immediately.
- We can see that MAC is strictly more powerful than forward checking.
  - Unlike MAC, forward checking does not recursively propagate constraints.

Local Search

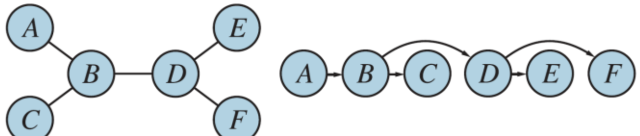
- Start with a possibly conflicting but complete assignment
- Pick a conflicted variable randomly and change its value
- Use Min-Conflicts heuristic to find a new value
  - Select the value that results in the smallest number of conflicts
  - Quite effective. Can solve a million variable N-Queens in 50 or so steps!
- Local search is great in online settings since repairing is usually much faster than solving from scratch

Local Search



Tree Structured CSP

- CSP is NP-complete in general BUT any tree structured CSP can be solved in linear time in the number of CSP variables
- Directional Arc Consistency (DAC)
  - Given an ordering of variables X_1, X_2, …, X_n, a CSP is DAC iff every X_i is arc-consistent with X_j where $j > i$
- Linear time algorithm: topological sort the variables, make the tree DAC and traverse from root down the leaves picking any remaining values.



(a) (b)

Quiz

1. What is the primary purpose of arc consistency (AC-3) in a CSP solver?
   A. To identify a single correct value for each variable.
   B. To remove values from a variable's domain that cannot possibly be part of a consistent solution.
   C. To add new constraints to the problem to make it easier to solve.

2. When a search algorithm for a CSP finds a partial assignment that violates a constraint, what does it do?
   A. It changes the domain of the conflicted variable to resolve the issue.
   B. It adds a new variable to the problem.
   C. It backtracks to the most recent variable with remaining values in its domain.

Quiz

1. What is the primary purpose of arc consistency (AC-3) in a CSP solver?
   A. To identify a single correct value for each variable.
   B. To remove values from a variable's domain that cannot possibly be part of a consistent solution.
   C. To add new constraints to the problem to make it easier to solve.

2. When a search algorithm for a CSP finds a partial assignment that violates a constraint, what does it do?
   A. It changes the domain of the conflicted variable to resolve the issue.
   B. It adds a new variable to the problem.
   C. It backtracks to the most recent variable with remaining values in its domain.

Quiz

3. What is the purpose of the least constraining value heuristic when solving a CSP?
   A. To choose the value that has the fewest constraints associated with it.
   B. To choose the value that is most likely to lead to a solution.
   C. To choose the value that prunes the smallest number of values from the domains of neighboring variables.

4. What is the key difference between backtracking search and local search for CSPs?
   A. Backtracking uses a single variable, while local search considers multiple variables at once.
   B. Backtracking is a DFS, while local search starts with a full assignment and improves it.
   C. Backtracking is incomplete, while local search is complete.

5. What is the min-conflicts heuristic used for in local search for CSPs?
   A. To select the variable that is in a conflict.
   B. To choose which variable to assign a value to next.
   C. To choose a value for a conflicted variable that results in the minimum number of conflicts with other variables.

Quiz

3. What is the purpose of the least constraining value heuristic when solving a CSP?
   A. To choose the value that has the fewest constraints associated with it.
   B. To choose the value that is most likely to lead to a solution.
   C. To choose the value that prunes the smallest number of values from the domains of neighboring variables.

3. What is the key difference between backtracking search and local search for CSPs?
   A. Backtracking uses a single variable, while local search considers multiple variables at once.
   B. Backtracking is a DFS, while local search starts with a full assignment and improves it.
   C. Backtracking is incomplete, while local search is complete.

4. What is the min-conflicts heuristic used for in local search for CSPs?
   A. To select the variable that is in a conflict.
   B. To choose which variable to assign a value to next.
   C. To choose a value for a conflicted variable that results in the minimum number of conflicts with other variables.

Summary

- A CSP is defined by three components:
  A. A set of variables
  B. A domain of possible values for each variable
  C. A set of constraints that specify which combinations of values are allowed.
- Constraints can be classified by the number of variables they involve.
  - A unary constraint affects a single variable, a binary constraint involves two variables, and a global constraint affects more than two.
- Many real-world problems can be effectively formulated as CSPs

Summary

- Backtracking Search is a core algorithm for solving CSPs.
  - It works by performing a depth-first search on the variables. The algorithm incrementally assigns a value to one variable at a time, and if a variable assignment leads to a constraint violation, it backtracks to the previous variable and tries a different value.
- Heuristics are used to improve backtracking search
  - Variable Ordering: The Minimum Remaining Values heuristic selects the variable with the fewest legal values remaining.
  - Value Ordering: The least constraining value heuristic selects the value that rules out the fewest choices for neighboring variables.

Summary

- Constraint Propagation is a technique used to prune the search space. It works by using constraints to reduce the domain of variables.
  - Arc consistency (AC-3) is a popular algorithm that removes values from a variable's domain that have no consistent value in a neighboring variable's domain.
- Forward Checking checks for conflicts with unassigned variables immediately after a variable is assigned.
  - Prunes the domains of neighboring unassigned variables, preventing future dead ends.

Summary

- Local Search algorithms start with a complete but possibly invalid assignment and iteratively improve it by making small changes.
  - Min-Conflicts Heuristic guides the iterative repair process in local search.
    - For a variable is in conflict, it chooses a new value for that variable that minimizes the number of remaining conflicts with other variables.
- Completeness vs. Efficiency: Backtracking search is a complete algorithm, and local search algorithms, while often faster, are typically incomplete.
  - Local search may get stuck in a local optimum and fail to find a solution.

- Reading: Chapter 6
- Assignments: PS 4, csp.ipynb
- Next: Logical Agents, Chapter 7