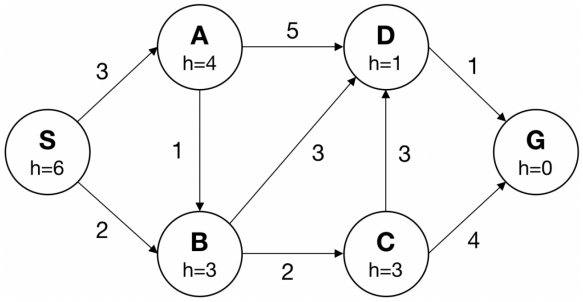
Due: 03/21/2023 23:59

Foundations of Artificial Intelligence: Homework 1

Instructor: Shang-Tse Chen & Yun-Nung Chen

Problem 1 (14 points)



In the above graph, the start and goal states are S and G, respectively. Write down the order of state expansion and the final path returned by each of the graph search (as oppose to tree search) algorithms below. You can assume ties are resolved alphabetically.

a) Depth-first search.

States Expanded: S, A, B, C, D, G

Path Returned: S-A-B-C-D-G

b) Breadth-first search.

States Expanded: S, A, B, D, C, G (or S, A, B, C, D, G)

Path Returned: S-A-D-G

c) Uniform cost search.

States Expanded: S, B, A, C, D, G

Path Returned: S-B-D-G

d) Greedy search with the heuristic h shown on the graph.

States Expanded: S, B, D, G

Path Returned: S-B-D-G

e) A^* search with the same heuristic.

States Expanded: S, B, D, G

Path Returned: S-B-D-G

f) Is the heuristic in the graph admissible? If not, can you make it admissible by changing the cost of an edge? Yes.

g) Is the heuristic in the graph consistent? If not, can you make it consistent by changing the cost of an edge? No. Change the cost of path S-B to 3.

Problem 2 (10 points)

Suppose the heuristic overestimates the shortest path from any state to the goal by a factor of at most ϵ, where $\epsilon > 1$. Prove that the cost of the path found by A* tree search is at most ϵ times the cost of the optimal path.

Proof. Let $h(n)$ be the heuristic that overestimates.
Let $g(n)$ be the true cost from start to n.
Let $t(n)$ be the true cost from n to goal.

If all the nodes in the path found by A^* tree search are also in the optimal path, then we are done.
If not, let n be a node in the path found by A^* tree search but not in the optimal path.
$\exists n_1$ in the optimal path s.t.
Cost(Path found by A^* tree search)
$= g(n) + t(n)$
$\leq g(n) + h(n)$ (h(n) overestimates)
$\leq g(n_1) + h(n_1)$ (lower priority on heuristic and not selected by A*)
$\leq g(n_1) + \epsilon * t(n_1)$ (h(n) overestimates at most ϵ)
$\leq \epsilon * g(n_1) + \epsilon * t(n_1)$ ($\epsilon > 1$)
$= \epsilon * (g(n_1) + t(n_1))$
$= \epsilon *$ (optimal cost)
The found path is at most ϵ times the cost of the optimal path.

□

Problem 3 (10 points)

You are scheduling for 5 classes on the same day taught by 3 instructors. Of course, each instructor can only teach one class at a time.

The classes are:

- Class 1 - Intro to Programming: 8:00-9:00am
- Class 2 - Intro to Artificial Intelligence: 8:30-9:30am
- Class 3 - Natural Language Processing: 9:00-10:00am
- Class 4 - Computer Vision: 9:00-10:00am
- Class 5 - Machine Learning: 10:30-11:30am

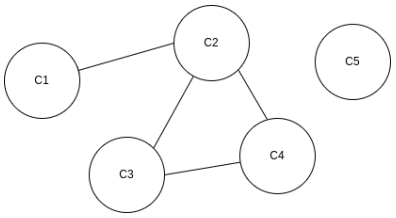
The instructors are:

- Instructor A - Can teach Classes 1, 2, and 5.
- Instructor B - Can teach Classes 3, 4, and 5.
- Instructor C - Can teach Classes 1, 3, and 4.

(1) Formulate this problem as a CSP. Describe the variables, domains and constraints.

Variables	Domains	Constraints
C_1 (Class 1)	A,C	$C_1 \neq C_2$
C_2 (Class 2)	A	$C_2 \neq C_3$
C_3 (Class 3)	B,C	$C_2 \neq C_4$
C_4 (Class 4)	B,C	$C_3 \neq C_4$
C_5 (Class 5)	A,B	

(2) Draw the constraint graph associated with your CSP.



(3) Show the domains of the variables after running arc-consistency on this initial graph (after having already enforced any unary constraints).

Variables	Domains
C_1 (Class 1)	C
C_2 (Class 2)	A
C_3 (Class 3)	B,C
C_4 (Class 4)	B,C
C_5 (Class 5)	A,B

(4) Give one solution to this CSP.

Variables	Value
C_1 (Class 1)	C
C_2 (Class 2)	A
C_3 (Class 3)	B
C_4 (Class 4)	C
C_5 (Class 5)	A

(5) Your CSP should look nearly tree-structured. Briefly explain (one sentence or less) why we might prefer to solve tree-structures CSPs.

- Minimal answer: Can solve them in polynomial time. If a graph is tree structured (i.e. has no loops), then the CSP can be solved in $O(nd^2)$ time as compared to general CSPs, where worst-case time is $O(d^n)$. For tree-structured CSPs you can choose an ordering such that every node's parent precedes it in the ordering. Then you can greedily assign the nodes in order and will find a consistent assignment without backtracking.

Problem 4

(6 points)

Alice, Bob, Chris, and David are ordering food from pizza, quesadillas, ramen, and sushi. They have some strict preferences:

1. Chris will not order sushi.
2. Alice and Bob want to order different food.
3. Bob will only order pizza or ramen.
4. Alice and Chris want to order the same dish as each other but different from the remaining two people.
5. David will not order quesadillas.

a) Draw the constraint graph for this CSP. Edges: A-B, C-A, C-B, C-D
b) Run the basic backtracking search. Use alphabetical order to both select unassigned variables and iterate over values. Write down the food assignment. Alice = pizza, Bob = ramen, Chris = pizza, David = ramen
c) Assume that no variables have been assigned values yet. When running one iteration of forward checking, which value(s) will be removed for each variable if we assign "pizza" to Alice. Write down "None" if no values will be removed.
Values that will be eliminated for Bob: pizza
Values that will be eliminated for Chris: quesadillas, sushi, ramen (or just quesadillas, ramen)
Values that will be eliminated for David: none