ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

Department of Computer Science and Engineering (CSE)

MID SEMESTER EXAMINATION **WINTER SEMESTER, 2020-2021**

DURATION: 1 Hour 30 Minutes **FULL MARKS: 75**

CSE 4711: Artificial Intelligence

Programmable calculators are not allowed.

There are **3 (three)** questions. Answer all **3 (three)** of them.

Figures in the right margin indicate marks of each question.

The square brackets on the start of each question denotes the corresponding CO and PO.

---

1. There are around 4.3×10^{19} possible configurations of a 3×3 Rubik's cube. However, if played optimally, any configuration can be solved in 20 moves or less. Here, one single move consists of a rotation of one of the faces of the cube. There are 27 possible rotations from a single configuration. We can pose the problem of solving Rubik's cube as a search problem. Assume that the closest solution from our start state requires exactly 20 moves.

Figure 1: Rubik's Cube

a) [CO2, PO4] Model the state space graph for this problem. 10

Solution: **State:** Each possible configuration of the cube corresponds to a state. There are 4.3×10^{19} states.
Actions: Rotations. There are 27 actions available from each state.
Successor Function: Go from one state to another state with a single rotation. There will be an edge connected to the state. The cost can be the number of moves.
Start State: Given the initial configuration of the Rubik's cube.
Goal Test: For each face, the colors of the individual cubelets on that face are the same.

Rubric:

- 2 points for each portion

b) [CO2, PO4] Model the search tree from the state space graph. 3

Solution: **Branching Factor:** 27, since we can make 27 moves from each state.
Maximum Depth: 20, since any configuration can be solved in 20 moves or less.
Solution Depth: As mentioned in the question, the depth of the solution is 20.

Rubric:

- 1 point for each portion

c) [CO3, PO2] Considering both the tree search and graph search variants of Breadth-First Search (BFS) and Depth-First Search (DFS), which algorithm will you prefer to find the solution? Justify your choice. 12

Solution: (This is one of the possible answers. Other logical answers are acceptable as well)
Since the solution is at the last level of the search tree, we should opt for DFS over BFS. Because BFS tree search will explore 27^{20} nodes in both the best and the worst case. BFS graph search performs a bit better, expanding 4.3×10^{19} nodes in both the best and the worst case. On the other hand, the DFS tree search can get stuck in a loop since the state space graph can contain cycles. However, the best case solution would require expanding only 20 nodes. The worst-case for DFS graph search occurs if the solution is on the rightmost node of the tree and we go from left to right. In this case, we will expand 4.3×10^{19} nodes. However, the best case can expand only 20 nodes. So the solution would require expanding anything between $[20, 4.3 \times 10^{19}]$ nodes.
Since the state space graph is huge, we can go for a DFS graph search. This will find a solution with possibly less number of nodes expansion than BFS, while also using a relatively small amount of memory.

Rubric:

- Reasoning for the choice: 6 points
- Reasoning for not picking other algorithms: 2×3 points

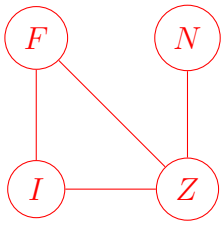
2. Zahid (Z), Ishrak (I), Farabi (F), and Nafisa (N) came to Chini-Come, a restaurant near their university. The restaurant serves Special Rice (S), Biriyani Rice (B), Kashmiri Naan (K), and Paratha (P). You overhear their conversations, and come up with the following preferences:

   i. Zahid does not like Paratha.
   ii. Ishrak and Farabi want to grab a bite of each other's food. So they want to order different dishes.
   iii. Farabi likes Rice items. So he'll either take Special Rice or Biriyani Rice.
   iv. Zahid wants to take a unique dish. However, he loves to copy Ishrak and will order the same dish as Ishrak.
   v. Nafisa will not order Kashmiri Naan as she had them earlier.

   Now you want to figure out who will order what using CSP.

   a) [CO2, PO4] Draw the constraint graph considering each person as a variable. 5

Solution: **Constraint Graph:**



Rubric:

- 0.5 points for each node.

- 0.5 points for each correctly included edge.
- 0.5 points for each correctly omitted edge.

b) [CO1, PO1] Show the remaining values in the domains of each variable after enforcing the unary constraints. 8

Solution:

- $Z \rightarrow S, B, K$
- $I \rightarrow S, B, K, P$
- $F \rightarrow S, B$
- $N \rightarrow S, B, P$

Rubric:

- 0.5 points for each correctly included value
- 0.5 points for each correctly omitted value

c) [CO1, PO1] Based on the result in Question 2(b), determine the cutset that will result in the least number of residual CSPs and provide brief reasoning for your choice. How many residual CSPs will you get? $2 + 1$

Solution: Since F has the least remaining values, removing it will result in 2 residual CSPs. So we'll pick F.

Rubric:

- Correct choice: 1 point
- Explanation: 1 point
- Residual CSP count: 1 point

d) [CO3, PO2] Let's say Ishrak orders Biriyani Rice. Based on the result in Question 2(b), perform Arc Consistency and show the remaining values for the domain of each variable. What would have been the difference if we performed Forward Checking here? $8 + 1$

Solution:

- $Z \rightarrow B$
- $I \rightarrow B$
- $F \rightarrow S$
- $N \rightarrow S, P$

If we performed forward checking, it would not have considered the $N - Z$ edge, and would not remove S from the domain of N.

Rubric:

- 0.5 points for each correctly included value
- 0.5 points for each correctly omitted value
- 1 point for forward checking explanation

3. Consider that you have an encryption software that takes a string, and converts it to an encrypted string. You used this software to encrypt your password and store it in a file, but somehow you forgot the password. You know that the password contains only the letters X, Y, and Z. The only way to recover the password is to provide a guess to the software, which will give you the encrypted string and then compare it with the string stored in the file. You are trying to recover your password using search.

   You formulate your search problem in this manner:
   State: Each possible string consisting of the letters X, Y, Z
   Start State: An empty string
   Successor Function: Appends one letter (X, Y, or Z) to the string
   Path Cost: You suspect that some letters are more likely to occur than others. To encode this in your search, you set $cost(X) = 1, cost(Y) = 2, cost(Z) = 3$.
   Goal Test: Verify a candidate password using the decryption software.

   Your search algorithm will keep generating different passwords and send them to the encryption software for checking until you find the first match. Assume that all ties are broken alphabetically.

   a) [CO3, PO3] Assume that your search algorithm predicts up to 10 character strings. There are 6 correct passwords: $XXXZZZ$, $XYYZZ$, $YXYXY$, $YZXYXZY$, $ZYXZ$, and $ZYXZY$. Matching with any one of them will do. With a brief explanation, select the one that will be returned by: 6×3

      i. Depth-First Search (DFS)

      Solution: Depth-first search will expand states in alphabetical order: X, XX, XXX, $\ldots$, $XXXXXXXXXX$, $XXXXXXXXXY$, $\ldots$. With alphabetical tie-breaking, the depth-first search will return the correct password that sorts first alphabetically: in this case, $XXXZZZ$.
      Rubric:

      - 2 points for correct choice
      - 4 points for correct explanation

      ii. Breadth-First Search (BFS)

      Solution: Breadth-first search will return the shortest correct password, $ZYXZ$. All of the other passwords are longer, so tie-breaking does not affect the answer.
      Rubric:

      - 2 points for correct choice

- 4 points for correct explanation

iii. Uniform Cost Search (UCS)

Solution: $YXYXY$ has the lowest cost (8) among the six goal states listed, so it will be returned by Uniform Cost Search. All of the other correct passwords have higher cost, so tie-breaking does not affect the answer.
Rubric:

- 2 points for correct choice
- 4 points for correct explanation

b) [CO1, PO1] Assume that there is a single correct password of length 10, chosen uniformly at random from the state space. That means you do not know about the likelihood of the letters. The correct password is unknown, but a candidate password can be checked using your decryption software. Justify the statement: Given any heuristic, A* search will, on average, expand the same number of states as Depth-First Search. 7

Solution: Since we don't know the likelihood of the letters, all the letters have cost 1. The correct password is unknown, so the heuristic we define can't be a function of the correct password. Moreover, only full passwords can be checked (i.e. there is no way of measuring partial progress towards the goal.) We have to keep checking candidate passwords until we find the correct one. Since any password could have been chosen, all with equal probability, exploring the state space in any order will (on average) be equally effective.
Rubric:

- 2 points for correct assumption of the cost.
- 2 points for explaining ineffectiveness of the heuristic.
- 3 points for explanation regarding expanding up to the leaf nodes.