Summary of Marking Guidelines:

- If you want you can use this as a convenient place to keep your overall instructions to your markers. This way you can find these instructions years later when you are looking back at the exam.

  If you are seeing this and not a cover page, you have solutions set to true in the exam.tex file.

Question 1. [10 MARKS]

This question deals with search problem, uninformed search techniques, and informed search techniques.

Part (a) [3 MARKS]

Consider the following variation of the jug puzzle problem: You have three jugs measuring 12 gallons, 8 gallons, and 3 gallons, and a water faucet. Initially no jug conatins any water. You can fill the jugs up or empty them out from one to another or onto the ground. You need to measure out exactly one gallon.

Cast this problem as a search problem. Write down a precise description of the state space, actions, intial state, and goal state. You are NOT required to solve the problem.

[0.5 mark] State space: all triples $(x,y,z) \in \{0,\dots,12\} \times \{0,\dots,8\} \times \{0,\dots,3\}$.

[1.5 marks] Actions:

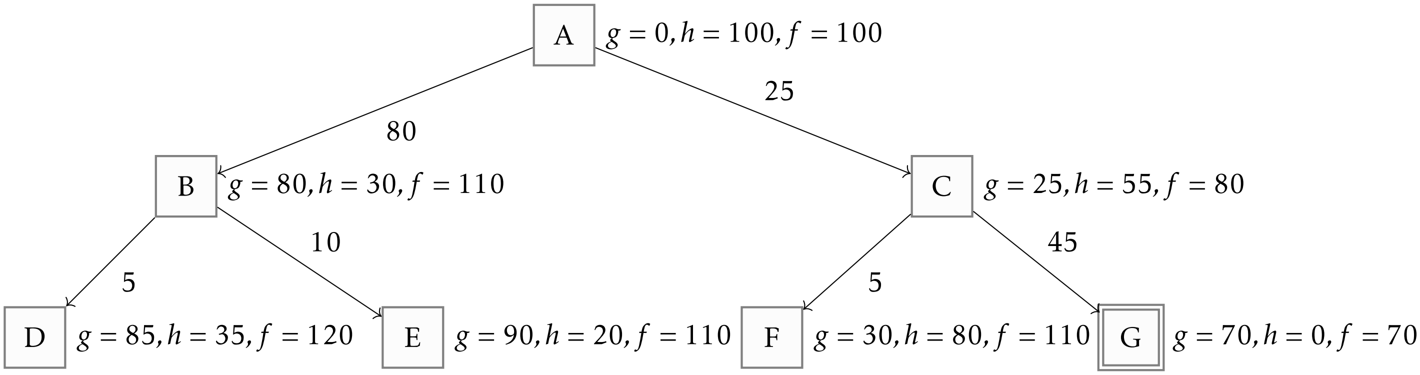
- Fill up from the tap: $(x,y,z) \to (12,y,z), (x,y,z) \to (x,8,z), (x,y,z) \to (x,y,3)$
- Dump to the ground: $(x,y,z) \to (0,y,z), (x,y,z) \to (x,0,z), (x,y,z) \to (x,y,0)$
- Dump first jar to the second: let $a = min(x, 8-y)$, then $(x,y,z) \to (x-a, y+a, z)$.
- Repeat for x,z and y,z.

[0.5 mark] Initial state: $(0,0,0)$

[0.5 mark] Goal state: $(1,*,*), (*,1,*), (*,*,1)$.

Parts b to g setup. Use the following tree to indicate the order that nodes are expanded, for different types of search. Here, path costs are shown to the right of each path.

- A is the start node.
- G (double box) is the only goal node.
- g = cost of path so far (g-cost).
- h = estimate of remaining cost to goal (heuristic or h-cost).
- f = estimate of total path cost (f-cost).



Part (b) [1 MARK]

Uniform cost-search. A,C,F,G

Part (c) [1 MARK]

Iterative deepening depth-first search. A; A, B, C; A, B, D, E, C, F, G.

Part (d) [1 mark]

Iterative deepening depth-first search. A; A, B, C; A, B, D, E, C, F, G.

Part (e) [1 mark]

Iterative deepening depth-first search. A; A, B, C; A, B, D, E, C, F, G.

Part (f) [1 mark]

A*-search. A, C, G.

Part (g) [2 marks]

Is the heuristic h admissible?. Circle one of the following:
Yes
No
In either case, write a detailed explanation why.

Answer: NO. For example, h(A)=100, but the optimal cost to the goal node G is only 70. Thus, h(A) sometimes OVER-ESTIMATES the remaining optimal distance to G, and so is not admissible.

Question 2. [5 MARKS]

Recall that

- True path cost so far = $g(n)$.
- Estimated cost to goal = $h(n)$.
- Estimated total cost = $f(n) = g(n) + h(n)$.

Ilir wrote a proof that A* tree search (queue sorted by f) is optimal if the heuristic h is admissible. He wrote each line in a piece of paper and labelled them from A to G, not necessarily in the correct alphabetic order. Unfortunately, the proof scrambler showed up and scrambled most of the papers.

Let n_g be the first goal node popped off the queue. Let n_0 be any other node on the queue. We wish to prove that n_0 can never be extended to a path to any goal node that costs less than the path to n_g that we just found.

(A) true total cost of n_g

(B) $\leqslant f(n_0)$ // because queue is sorted by f

(C) $\leqslant g(n_0)+$ true cost to goal from n_0 // because h is admissible

(D) $= f(n_g)$ // by definition of f with $h(n_g) = 0$ since n_g is a goal

(E) $= g(n_0) + h(n_0)$ // by definition of f

(F) $= g(n_g)$ // because n_g represents a complete path

(G) = true total cost of no extended to a path to any goal node

Fill in the blanks with the letters B, C, D, E, F in the correct order to prove that the true total cost of $n_g \leq$ true total cost of n_0. The first and last letters, A and G, have been done for you as an example. (MARKING: You will earn one mark for each letter (from B to F) placed in the correct position)

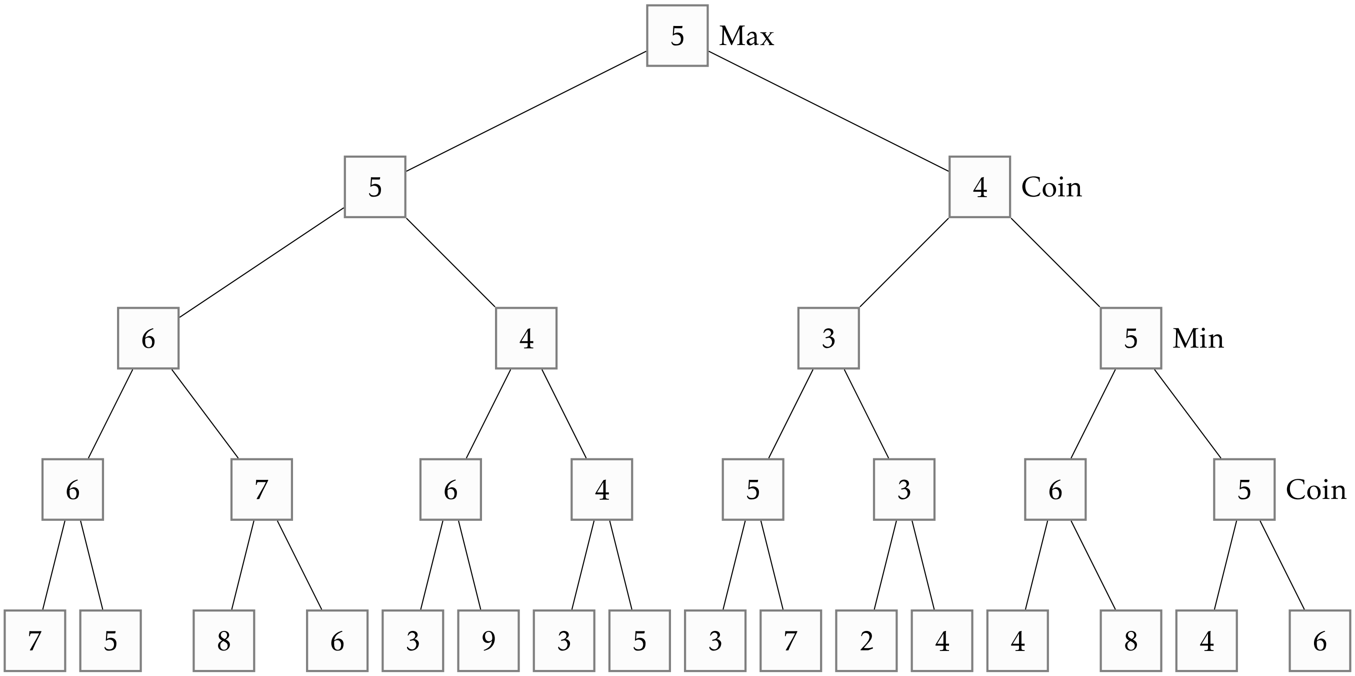
Answer:

A F D B E C G

Question 3. [5 MARKS]

Part (a) [3 MARKS]

The following problem asks about MINIMAX search in game trees with chance (also called EXPECTI-MAX search). This is similar to MINI-MAX, but there is a coin flip ("COIN") before every other player's move, as shown in the image. That is, the sequence of moves is: COIN, MIN, COIN, MAX (moving from bottom up in the tree). If the coin turns up heads, the path labeled "H" is taken; if it turns up tails, the path labeled "T" is taken. The value passed upward from the COIN level is the probabilistic weighted average (the expected value) of the two paths "H" and "T". Assume a fair coin, i.e., $P(H) = P(T) = 0.5$. Please think carefully about how to pass values upwards from the COIN level before working this problem. The game tree below illustrates one position reached in the game. It is MAX's turn to move. Below the leaf nodes are the estimated score of each resulting position returned by some heuristic static evaluator. FILL IN ALL BLANK SQUARES WITH THE VALUES PASSED UPWARDS STARTING FROM THE LEAF NODES.



Part (b) [2 marks]

Most game-playing programs do not save search results from one move to the next. Instead, they usually start completely over whenever it is the machine's turn to move. Why?

Solution: Extending by one ply the search tree produced in selecting a previous move requires the generation of $b \times b$ times as many nodes as are in the saved search tree, where b is the average branching factor. For $b = 10$, say, only 1 percent of the nodes needed to be generated to select a next move are already in the saved tree. Thus, saving the tree would not contribute meaningfully to efficency.

Use the space on this "blank" page for scratch work, or for any solution that did not fit elsewhere.
Clearly label each such solution with the appropriate question and part number.

Total Marks = 20