## Question 1. True/False [6 MARKS]

Circle either True or False to indicate the truth of each of the following statements. 1 mark each. No marks will be deducted for incorrect answers.

**(a)** [1 MARK] A* search will always expand fewer search nodes than uniform cost search.

TRUE FALSE

**(b)** [1 MARK] Assume Iterative Deepening DFS increased the depth limit by 2 at each iteration rather than by 1. Assuming the cost to get from each state to its successors is 1, this version of ID-DFS will be optimal.

TRUE FALSE

**(c)** [1 MARK] Any Bayes Net defined over 5 variables $X1, X2, ...X5$ tells us how to factor the joint probability distribution $P(X1, X2, ...X5)$ into a product containing exactly 5 terms.

TRUE FALSE

**(d)** [1 MARK] Alpha-beta pruning that uses a heuristic evaluation function in some areas of the game tree will yield an optimal playing strategy.

TRUE FALSE

**(e)** [1 MARK] The time and space complexity of uniform cost search is $O(b^{C*/\epsilon})$ where $\epsilon$ is the cost of the most expensive transition between explored states.

TRUE FALSE

**(f)** [1 MARK] The initial state for the min-conflicts algorithm is a complete but inconsistent assignment of values to variables.

TRUE FALSE

## Question 2. Short Answer [6 MARKS]

**(a)** [2 MARKS] Why do most search algorithms we have covered stop only when a goal node is expanded or removed from the frontier, instead of stopping when the goal node is first discovered?

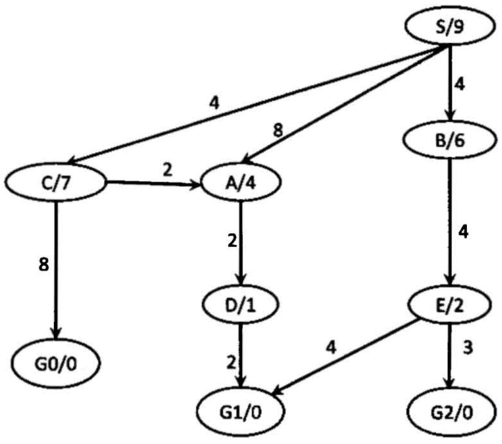
**(b)** [2 MARKS] Explain in English what the following statement means:

If $KB \models f$ then $KB \vdash f$

**(c)** [2 MARKS] Why is the Upper Confidence Bound useful in making selection decisions in a Monte Carlo Tree Search? (Recall: the UCB1 formula selects some node $i$ such that $v_i + C\sqrt{\frac{ln(N)}{n_i}}$ is maximized, where $v_i$ is the value estimate for node $i$, $C$ is a parameter that can be tunied, $N$ is total number of trials, and $n_i$ the number of trials through node $i$.)

1. **Search—15/140 points (10.7%)**

   (a) Consider the search graph below, where **A** is the start node and **G0**, **G1**, and **G2** are goal states. Arcs are labeled with the cost of traversing them and the heuristic estimate of the cost to a goal is shown inside the nodes. For each of the three search strategies below, indicate which of the three goal states is reached.



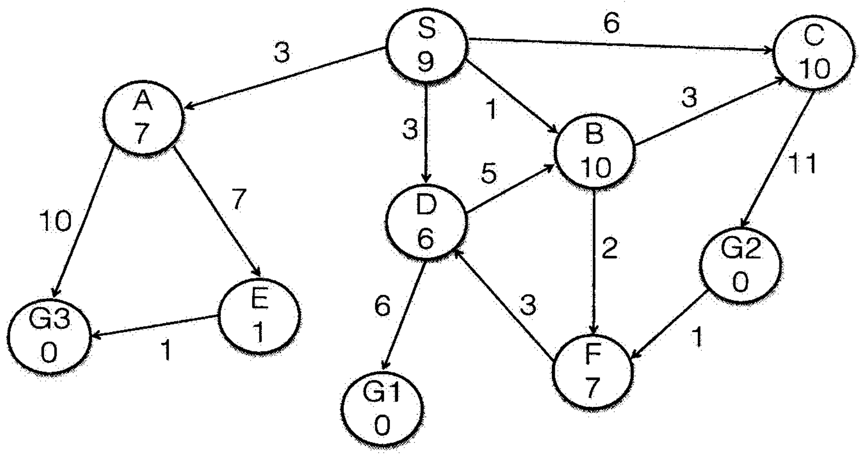
i. [2] Breadth-first search:

ii. [2] Uniform Cost Search:

iii. [2] A* Search:

## Question 1. Search [6 MARKS]

Consider the search graph below, where **S** is the start node and **G1**, **G2**, and **G3** are goal states. Arcs are labeled with the cost of traversing them and the heuristic cost to a goal is shown inside the nodes. For each of the three search strategies below, indicate which of the goal states is reached. If tie-breaking is required, assume it is left-to-right. E.g., if two nodes are tied on the OPEN list, first expand the one that appears to the left of the other in the figure.



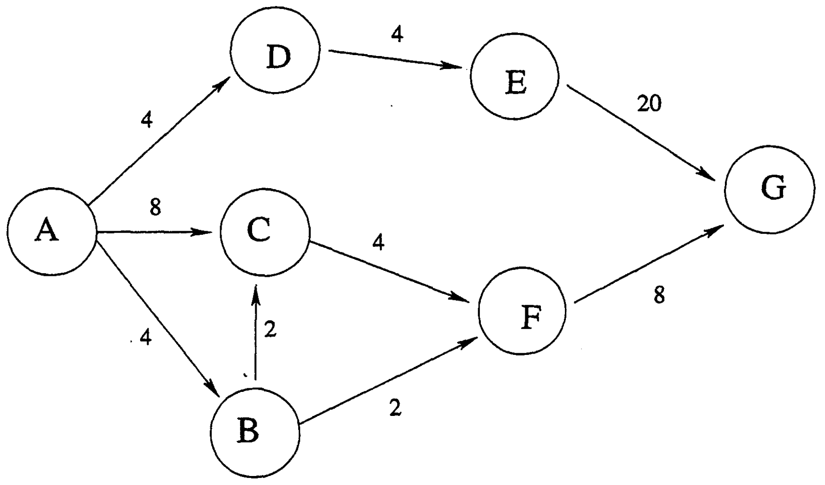
**(a)** [2 MARKS] Breadth-first search:

**(b)** [2 MARKS] Uniform cost search:

**(c)** [2 MARKS] A* search:

## Question 1. Search [12 MARKS]

Consider the problem of searching the following graph with $A$ being the start state and $G$ being the goal state. Each edge in the graph is labeled by its cost, and the heuristic value of each node $n$ in the graph, $h(n)$, is also indicated. For example, $h(A) = 18$ and $c(A \to D) = 4$.

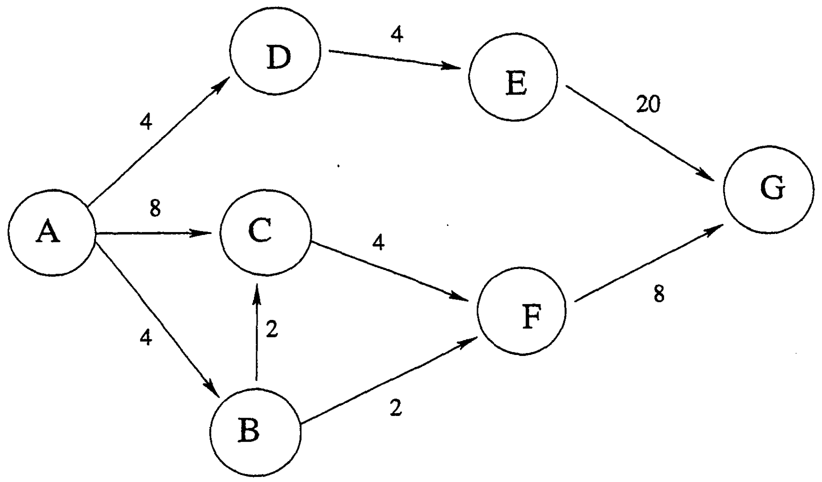


h(A) = 18
h(B) = 10
h(C) = 4
h(D) = 4
h(E) = 4
h(F) = 6
h(G) = 0

**(a)** [6 MARKS] Trace the execution of A* on the graph. Assume that **no cycle checking** is performed. Show the successive configurations of the frontier where the elements on the frontier are paths. That is, the path $n_1 \to n_2 \to n_3$ would be written as $[n_1, n_2, n_3]$. Next to each path of the frontier, indicate the $f$ value of the final node in the path (e.g., if $f(n_3) = 10$ write "10" next to the path and enclosed in parens with the path). Indicate the path that is expanded at each stage. Break **ties** (if there are any) by expanding the path most recently added to the frontier first.

| Iteration | Node Expanded | Frontier |
|---|---|---|
| *0* | | ([A],18) |
| *1* | | |
| *2* | | |
| *3* | | |
| *4* | | |
| *5* | | |
| *6* | | |
| *7* | | |
| *8* | | |
| *9* | | |
| *10* | | |

**(b)** [2 MARKS] The path to the goal found by A*is:



h(A) = 18
h(B) = 10
h(C) = 4
h(D) = 4
h(E) = 4
h(F) = 6
h(G) = 0

**(c)** [4 MARKS] Trace the execution of uniform cost search on the graph. Assume that **no cycle checking** is being performed. Show the successive configurations of the frontier (as before the elements of the frontier are paths written in reverse order). Next to each path of the frontier, indicate the cost of the path. Indicate the path that is expanded at each stage. Expand nodes bottom to top. Break **ties** by expanding the path that was more recently added to the frontier first.

| Iteration | Node Expanded | Frontier |
|---|---|---|
| *0* | | ([A],0) |
| *1* | | |
| *2* | | |
| *3* | | |
| *4* | | |
| *5* | | |
| *6* | | |
| *7* | | |
| *8* | | |
| *9* | | |
| *10* | | |

**(d)** [2 MARKS] The path to the goal found by uniform cost is:

## Question 2. Search – Short Answer [12 MARKS]

**(a)** [3 MARKS] Is A*'s search behaviour necessarily exponential? That is, does the search time always grow at least exponentially with the length of the optimal solution? Explain your answer.

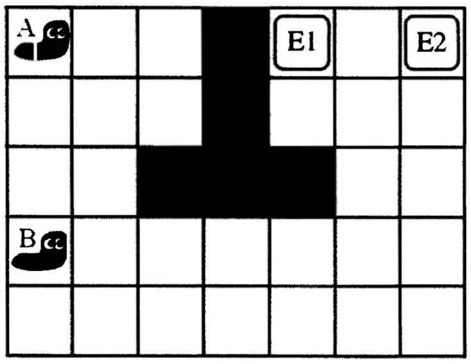
**(b)** [4 MARKS] It can be shown that when $h(n) = h^*(n)$ (i.e., when h(n) is a "perfect" heuristic) A* only expands nodes that lie on an optimal path to a goal. When this condition on $h(n)$ holds does it imply that A* will always take time linear in the solution length to find an optimal solution? Explain your answer.

**(c)** [5 MARKS] Suppose we are solving a search problem using A* search where actions have non-zero positive costs and the heuristic function is a so-called $\epsilon$-admissible heuristic function, meaning that there is some $\epsilon \geq 0$ and $h(n) \leq h^*(n) + \epsilon$ for all nodes $n$. Here, $h^*(n)$ is the optimal path cost and the lowest cost solution is of cost $C^*$.

(i) [2 marks] Will this version of A* be complete, i.e., will it be guaranteed to find a solution if there is one? Why?

(ii) [3 marks] When using an $\epsilon$-admissible heuristic, how far from the optimal cost ($C^*$) might our final solution be? Why?

(b) As shown in the diagram on the left, two slugs $A$ and $B$ want to exit a maze via exits $E_1$ or $E_2$. At each time step, each slug can either stay in place or move to an adjacent free square. A slug cannot move into a square that the other slug is moving into. Either slug may use either exit, but they cannot both use the same exit.



You wish to pose a search problem that will allow both slugs to get to exit in as few time steps as possible. Let $d(c_1, c_2)$ denote **Manhattan distance** between locations $c_1$ and $c_2$ (all locations consists of a pair of numbers $(x, y)$). Consider three different search heuristics, defined in terms of the location of slug $A$ and slug $B$ and exit locations $E_1$ and $E_2$. Remember that either slug can use either exit, but they must use different exits.

| *Definition* | *Explanation* |
|---|---|
| $h_1 = \max_{s \in \{A,B\}} \min(d(s, E_1), d(s, E_2))$ | Return the maximum, over the two slugs, of the distance from the slug to its closest exit. |
| $h_2 = \max(d(A, E_1), d(B, E_2))$ | Assign slug A to exit $E_1$ and B to $E_2$; then return the maximum distance from either slug to its assigned exit. |
| $h_3 = \min_{(e,e') \in \{(E_1,E_2),(E_2,E_1)\}} \max(d(A, e), d(B, e'))$ | Return the max distance from a slug to its assigned exit, under the assignment of slugs to distinct exits which minimizes this quantity. |

i. [4] Which of these three heuristics are admissible? Mark **all** admissible heuristics.

☐ $h_1$ ☐ $h_2$ ☐ $h_3$

ii. [5] For two heuristics $h_1$ and $h_2$ we say that $h_1$ **dominates** $h_2$ if $h_1(s) \geq h_2(s)$ for all states. (Note, the heuristics need not be admissible). For each pair of heuristics, mark the box that correctly describes their dominance relationship.

| | | | |
|---|---|---|---|
| ☐ $h_1$ dominates $h_2$ | ☐ $h_2$ dominates $h_1$ | ☐ $h_1 = h_2$ | ☐ none |
| ☐ $h_1$ dominates $h_3$ | ☐ $h_3$ dominates $h_1$ | ☐ $h_1 = h_3$ | ☐ none |
| ☐ $h_2$ dominates $h_3$ | ☐ $h_3$ dominates $h_2$ | ☐ $h_2 = h_3$ | ☐ none |

## Question 1. [14 MARKS]

**Search Algorithms.**

**Parts a and b setup**. Consider a state space where the start state is the number 1 and each state $k$ has two successors: numbers $2k$ and $2k + 1$.

### Part (a) [1 MARK]

Draw the portion of the state space for states 1 to 15.

### Part (b) [3 MARKS]

Suppose the goal state is 11. List the order in which nodes will be visited for breadth-first search, depth-limited search with limit 3, and iterative deepening search.

**Which of the following are true and which are false?** Explain your answers. Unless stated otherwise, assume a finite branching factor, step costs $\geq \epsilon > 0$, and at least one goal at a finite depth. You may be in either a tree or a graph.

## Part (c) [2 MARKS]

Breadth-first search is a special case of uniform-cost search.

## Part (d) [2 MARKS]

Depth-first search always expands at least as many nodes as A* search with an admissible heuristic.

## Part (e) [2 MARKS]

$h(n) = 0$ is an admissible heuristic for the 8-puzzle.

**Part (f)** [2 MARKS]

Breadth-first search is complete whenever the branching factor is finite, even if zero step costs are allowed.

**Part (g)** [2 MARKS]

Assume that a rook can move on a chessboard any number of squares in a straight line, vertically or horizontally, but cannot jump over other pieces; then Manhattan distance is an admissible heuristic for the problem of moving the rook from square A to square B in the smallest number of moves.

## Question 1. [21 MARKS]

This question deals with (uninformed and informed) search problems.

### Part (a) [1 MARK]

In state space search, what does it mean for an algorithm to be *complete*?

### Part (b) [1 MARK]

In state space search, what does it mean for an algorithm to be *optimal*?

### Part (c) [3 MARKS]

The *heuristic path algorithm* (Pohl, 1977) is a best-first search algorithm with the evaluation function

$$f(n) = (2 - w)g(n) + wh(n).$$

Assuming $h(n)$ is admissible, what is the name of the algorithm for the following settings of $w$?

(i) $w = 0$ ? ______

(ii) $w = 1$ ? ______

(iii) $w = 2$ ? ______

**Part (d)** [4 MARKS]

Consider a finite tree of depth d and branching factor $b$. (A tree consisting of only a root node has depth zero; a tree consisting of a root node and its $b$ successors has depth 1; etc.) Suppose the shallowest goal node is at depth $g \leq d$.

Write down a closed form expression that yields minimum and maximum number of nodes that might be generated by a depth-first iterative-deepening search? (Assume that you start with an initial depth limit of 1 and increment the depth limit by 1 each time no goal is found within the current limit.)

(Marking: 2 marks for the minimum nodes expression, and 2 marks for the maximum nodes expression)

**Part (e).** [3 MARKS]

In the "Four-Queens Puzzle" we try to place four queens on a $4 \times 4$ chess board so none can capture any other. (That is, only one queen can be on any row, column, dor diagonal of the board). Suppose we try to solve this puzzle using the following problem space:

- The *Start* node is labelled by an empty $4 \times 4$ board
- The *Successor* function creates new $4 \times 4$ boards containing one additional legal placement of a queen anywhere in the board
- The *Goal* node is labelled by a board containing precisely four legally positioned queens.

Invent an *admisible* heuristic function for this problem based on the number of the queen placements remaining to achieve the goal, different from the trivial heuristic $h(n) = 0$. (Note that all goal nodes are precisely four steps from the start node!). Prove that your heuristic is actually admissible.

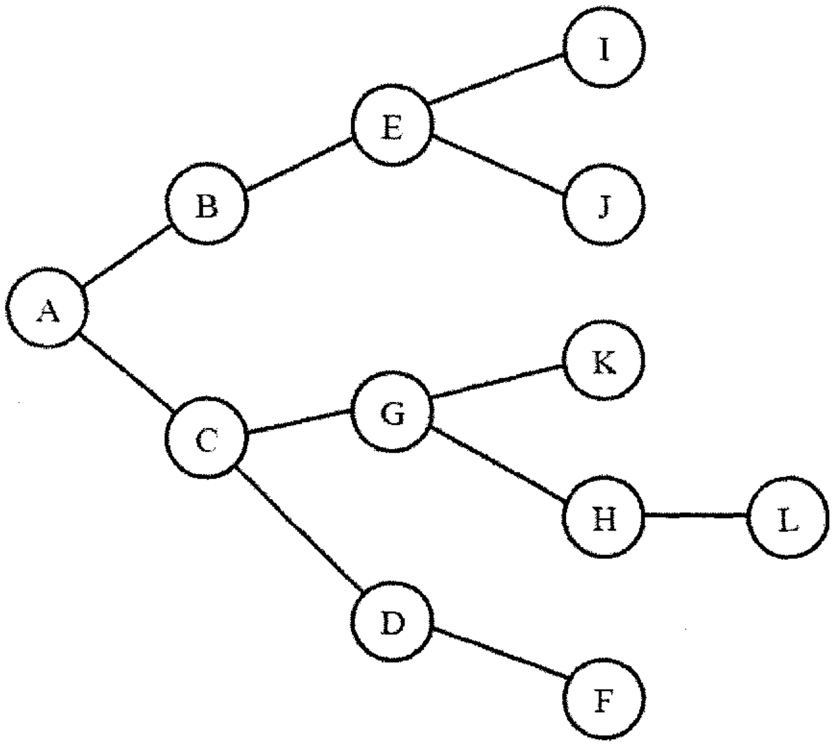
(Marking: 1 mark for a correct heuristic, 2 mark for the proof your heuristic is admissible).

**Part (f)** [3 MARKS]

Algorithm A* does not terminate until a goal node is selected for expansion. However a path to a goal node might be reached long before that node is selected for expansion. Explain why A* does not terminate as soon as a goal node has been insterted into OPEN? For full marks, provide an explicit example.

**Part (g)** [6 MARKS]

Consider the following search tree:



The goal state is K, and the heurstic function is given by the following table:

| | | | |
|---|---|---|---|
| A: | 9 | G: | 3 |
| B: | 4 | H: | 5 |
| C: | 5 | I: | 8 |
| D: | 7 | J: | 2 |
| E: | 3 | K: | 0 |
| F: | 10 | L: | 7 |

A-Given these values, draw a diagram in the next page that illustrates the search tree of explored states, given an A* search (the path cost is uniform, set to 3 units). Indicate the calculated cost at each node in the tree.

B-Is the given heuristic admissible? Answer "Yes" or "No".