

1. (Short Answers; 8 points)

- (a) [2 points] What is an *admissible* heuristic? A function that takes a node as an argument and returns a weakly positive number that is guaranteed to be smaller than the cost of the cheapest path from its argument to a goal node. I.e.,

$$\forall n_0, n_1, \dots, n_k \text{ where } \text{goal}(n_k) = 1 : \text{cost}(\langle n_0, \dots, n_k \rangle) \geq h(n_0).$$

- (b) [2 points] Is depth first search *complete*? Why or why not?

It is NOT COMPLETE. For a graph with unbounded path length, depth-first may continue searching the wrong path forever. (Appealing to the presence of cycles is also acceptable for full marks, even though that is less general)

- (c) [2 points] What does it mean for one heuristic h_1 to *dominate* another heuristic h_2 ?

h_1 dominates h_2 if $h_1(s) \geq h_2(s)$ for **all** states s , and $h_1(t) > h_2(t)$ for **at least one** state t .

- (d) [2 points] Which of depth-first search, least-cost-first search, and A* search has the best worst-case *space complexity*? Why?

Depth-first search has the best space complexity of $O(mb)$; the other two have worst-case space complexity of $O(b^m)$.

2. (Uninformed Search; 6 points) Consider the search graph below. There are 6 nodes: A, B, C, D, E, F . The nodes C, E, F are *goal nodes*. The edge costs are:

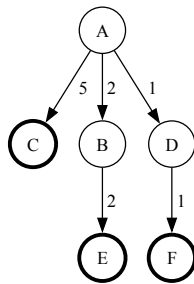
$$\text{cost}(A, B) = 2$$

$$\text{cost}(A, C) = 5$$

$$\text{cost}(A, D) = 1$$

$$\text{cost}(B, E) = 2$$

$$\text{cost}(D, F) = 1$$

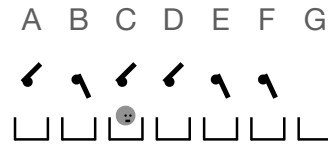


- (a) [2 points] Suppose that *depth-first search* has removed the following paths from the frontier: $\langle A \rangle$, $\langle A, B \rangle$. What path will depth-first search remove next? $\langle A, B, E \rangle$ will be the next path removed.

- (b) [2 points] What solution will *breadth-first search* find? BFS will find the $\langle A, C \rangle$ solution, because that is the solution with the smallest number of arcs (even though it is also the costliest solution).

- (c) [2 points] What solution will *least-cost-first search* find? LCFS will find the $\langle A, D, F \rangle$ solution.

3. (Heuristic Search; 22 points)



SwitchBot is an agent that lives in the 1-dimensional world pictured above. There are 7 positions in a row, labelled from position A on the left through G on the right. There is a switch associated with every position *except* G; initially, three of the switches (A,C,D) are set to “on” (pointing up), and the other three (B,E,F) are set to “off” (pointing down). SwitchBot starts in position C, as pictured.

In positions B,C,D,E, and F, SwitchBot can perform one of three actions: it can go **left**, which moves it to the position to its left (e.g., from C to B); it can go **right**, which moves it to the position to the right (e.g., from E to F); or it can **FLIP**, which will toggle the switch at the current position *and then* move SwitchBot to the position on its right. In position A, only the **right** and **FLIP** actions are available; in position G, only the **left** action is available.

SwitchBot’s goal is to get all of the switches into the “on” position using the fewest number of actions.

- (a) [3 points] How would you represent the *states* for this search problem?
($x, p_A, \dots, p_F$), where $x \in \{A, \dots, G\}$ is the position of bot and each p_j is a boolean indicating whether the switch at position j is on.
This is just an example of course; any state representation that can capture every valid state will do.
- (b) [2 points] How many states does this search problem have? $7 \times 2^6 = 448$. *It’s fine if they don’t evaluate the expression to get 448. It’s also fine if they give a different number so long as that number matches the representation they defined and the representation they defined is sufficiently expressive. (E.g., if someone got a larger number of states because they had a binary number per location for “robot here” and another binary number per switch for “switch on”, then that is fine).*
- (c) [2 points] What are the *successor* states for the starting state?

$$\begin{aligned} \text{succ}(C, 1, 0, 1, 1, 0, 0) &= \{(B, 1, 0, 1, 1, 0, 0) && \text{(after left)} \\ & & (D, 1, 0, 1, 1, 0, 0) && \text{(after right)} \\ & & (D, 1, 0, 0, 1, 0, 0)\} && \text{(after FLIP)} \end{aligned}$$

Half marks if they get everything right under the mistaken assumption that FLIP doesn’t move the agent. Full marks for listing the states without a leading “*succ*(...) =”; can also get full marks for specifying a function to calculate it instead of listing the states explicitly.

- (d) [2 points] Give an expression for the *goal function* for this search problem.

$$\text{goal}(x, p_A, \dots, p_F) = \begin{cases} 1 & \text{if } p_A = p_B = \dots = p_F = 1, \\ 0 & \text{otherwise.} \end{cases}$$

- (e) [4 points] Consider the following heuristic function:

$$h(s) = \sum_{y \in \{A,B,C,D,E,F\}} \mathbb{I}[\text{switch at position } y \text{ is off in state } s],$$

where the “indicator function” $\mathbb{I}[\cdot]$ returns 1 when its argument is true, and 0 otherwise. That is, $h(s)$ returns the number of switches at the “off” position in state s .

Is this heuristic function h an *admissible heuristic* for this problem? Why or why not?

Yes, this is an admissible heuristic. Any state with k switches set to off will need to execute at least k FLIP actions to reach a goal state, so $h(n) = k$ is a lower bound on the number of actions.

- (f) [3 points] Either construct an admissible heuristic function h' for this problem that *dominates* h , or else explain why this is impossible. **It is possible. Here is an example.** Consider the state $s^* = (G, 1, 1, 1, 1, 0)$. $h(s^*) = 1$, but it will in fact require 2 moves (left and then FLIP) to get to the solution. So let

$$h'(s) = \begin{cases} 2 & \text{if } s = s^*, \\ h(s) & \text{otherwise.} \end{cases}$$

Clearly h' dominates h , since $h'(s) \geq h(s)$ for all s , and $h'(s^*) > h(s^*)$.

- (g) [6 points] Is h a *consistent* heuristic? Why or why not? **Yes, this heuristic is consistent.** A consistent heuristic satisfies $h(n) \leq \text{cost}(n, n') + h(n')$ for all n, n' . There are three cases:

- i. After every left or right action, the number of switches in the off position is unchanged, so $h(n) = h(n')$, and thus

$$h(n) = h(n') \leq 1 + h(n') = \text{cost}(n, n') + h(n').$$

- ii. Same argument for a FLIP action when the bot is located at an on switch

- iii. When the bot is located at an off switch and performs a FLIP action, the number of off switches decreases by exactly 1, so

$$h(n) = 1 + h(n') = \text{cost}(n, n') + h(n'),$$

which satisfies $h(n) \leq \text{cost}(n, n') + h(n')$.

Only a single point for a correct guess with no supporting argument. Full marks for a correct argument; two points for correctly stating the definition.