HW1: Search

CS6300: Artificial Intelligence, Spring 2018 **Jake Pitkin**
University of Utah

---

1 Graph Search

For the following problems, S will denote Start and G will denote Goal. When choosing an arbitrary order of state expansions, alphabetical ordering will be used. Once visited, a state will not be expanded again.

1. Greedy search

Step	Priority Queue	Expand
1	(S, 0)	S
2	(S-A, 3), (S-B, 2)	B
3	(S-A, 3), (S-B-D, 1), (S-B-C, 2), (S-B-G, 4)	D
4	(S-A, 3), (S-B-C, 2), (S-B-G, 4), (S-B-D-G, 5)	C
5	(S-A, 3), (S-B-G, 4), (S-B-D-G, 5), (S-B-C-G, 1)	G

Table 1: Greedy search

Greedy search final path: $S \to B \to C \to G$

2. Depth first search

   Note: a LIFO queue is used and sibling nodes are enqueued in alphabetical order.

Step	LIFO Queue	Expand
1	S	S
2	B, A	B
3	G, D, C, A	G

Table 2: Depth first search

Depth first search final path: $S \to B \to G$

3. Breadth first search

   Note: a FIFO queue is used and sibling nodes are enqueued in alphabetical order. If a node is already in the queue, it is not re-added.

Step	FIFO Queue	Expand
1	S	S
2	A, B	A
3	B, G	B
4	G, C, D	G

Table 3: Breadth first search

Breadth first search final path: $S \to A \to G$

4. Uniform cost search

Step	Priority Queue	Expand
1	(S, 0)	S
2	(S-A, 3), (S-B, 2)	B
3	(S-A, 3), (S-B-C, 4), (S-B-D, 3), (S-B-G, 6)	A
4	(S-B-C, 4), (S-B-D, 3), (S-B-G, 6), (S-A-G, 6)	D
5	(S-B-C, 4), (S-B-G, 6), (S-A-G, 6), (S-B-D-G, 8)	C
6	(S-B-G, 6), (S-A-G, 6), (S-B-D-G, 8), (S-B-C-G, 5)	G

Table 4: Uniform cost search

Uniform cost search final path: $S \to B \to C \to G$

5. Greedy search with the heuristic values listed at each state

Step	Priority Queue	Expand
1	(S, 0)	S
2	(S-A, 2), (S-B, 3)	A
3	(S-B, 3), (S-A-G, 0)	G

Table 5: Greedy search with heuristic values

Greedy search with heuristic final path: $S \to A \to G$

6. A* search with the heuristic values listed at each state

Step	Priority Queue	Expand
1	(S, 0)	S
2	(S-A, 5), (S-B, 5)	A
3	(S-B, 5), (S-A-G, 6)	B
4	(S-A-G, 6), (S-B-C, 5), (S-B-D, 4), (S-B-G, 6)	D
5	(S-A-G, 6), (S-B-C, 5), (S-B-G, 6), (S-B-D-G, 8)	C
6	(S-A-G, 6), (S-B-G, 6), (S-B-D-G, 8), (S-B-C-G, 5)	G

Table 6: A* search

A* search final path: $S \to B \to C \to G$

2 Downhill Skiing

1. If the mountain is N units tall (eg., it is $N = 6$ units tall in the figure), what is the size of the state space? Justify your answer. (You may ignore "unreachable" states.) What are the start/goal states?

   Let's consider the state space S of the example where $N = 6$. A state $s_i \in S$ is defined by a position and a current velocity as such: $s_i = (position, velocity)$. First we will define the start state s_0 and goal state g.

   $$s_0 = (1, 0),\ g = (6, 0)$$

   For small N, we can simply enumerate all the possible velocities at each position on the hill to determine the reachable states. Her action adjusts her velocity *before* she moves positions, which is how a velocity of 1 at square 1 is possible. Also Alice makes a final action at the goal state.

(1, 0)	(1, 1)	-	-
(2, 0)	(2, 1)	(2, 2)	-
(3, 0)	(3, 1)	(3, 2)	-
(4, 0)	(4, 1)	(4, 2)	(4, 3)
(5, 0)	(5, 1)	(5, 2)	(5, 3)
(6, 0)	(6, 1)	(6, 2)	(6, 3)

Table 7: State space for Alice's ski run.

   To justify this table, we will consider it one column at a time.

   Column 1: Alice can be present in each square with a velocity of 0 by alternating `accelerate` and `decelerate` actions all the way down the hill.

   Column 2: Alice can `accelerate` at square 1 and `coast` the rest of the way down the hill.

   Column 3: From square $n - 1$ with a velocity of 0 before performing an action, `accelerate` twice. The state will be arrived at after the second `accelerate` but before moving.

   Column 4: From square $n - 2$ with a velocity of 1 before performing an action, `accelerate` twice. The state will be arrived at after the second `accelerate` but before moving.

   There is no way to have a velocity greater than 3 when $N = 6$. The best chance at the largest velocity is to `accelerate` for every action (as the higher on the hill you are the more space you have to accelerate before hitting the parking lot). Consider we `accelerate` 3 times. This will take us to space $(4, 3)$ after performing the action but before moving. Once we move, we will be overshooting the goal into the parking lot.

   Thus, $|\mathbf{S}| = \mathbf{20}$. As a note, only 14 of these are reachable *before* making an action. The states $(1, 1), (2, 2), (3, 2), (4, 3), (5, 3)$, and $(6, 3)$ are only reachable after making an action but before moving.

2. Give an example of a state that is not reachable. Suppose that Alice cannot coast (she must either accelerate or decelerate): does this yield *more* unreachable states? If so, give an example of one and justify your answer either way.

   An example of a state that is not reachable is $(1, 2)$ (being at square 1 with a velocity of 2). Her velocity at square 1 is either 0 (she `coasts` as her first move) or 1 (she `accelerates` as her first move).

   When we remove the action of `coast`, the number of unreachable states does **not** change. In question one, I used `coast` to justify being at each square with a velocity of 1. Another way to achieve this without coasting is to alternate between `accelerate` and `decelerate`. This will place Alice in all positions on the hill with a velocity of 1.

3. Is Alice's current elevation (i.e., distance from the chair lift) an admissible heuristic? Why or why not?

   Admissible: A heuristic $h(n)$ is admissible if $h(n) \leq h*(n)$ where $h^*(n)$ is the true cost to the nearest goal.

   Claim: Alice's current elevation is **not** an admissible heuristic for all hills of size N.

   Proof by counter-example: Consider a hill of size $N = 10$ with Alice at the start state. She is 9 positions away from the goal and as such $h(s_0) = 9$.

   Consider the following actions for Alice to get down the hill.

State	Action	New Velocity
(1, 0)	`accelerate`	1
(2, 1)	`accelerate`	2
(4, 2)	`accelerate`	3
(7, 3)	`decelerate`	2
(9, 2)	`decelerate`	1
(10, 1)	`decelerate`	0

Table 8: A possible ski run on a hill of size 10 in 6 actions.

   Proving this heuristic is not admissible as $h(s_0) \leq h^*(s_0)$ or $9 \leq 6$ does not hold.

4. State and justify a non-trivial, admissible heuristic for this problem which is *not* current elevation.

   Consider a *relaxed* version of Alice's ski run. Alice is a pro so regardless of her velocity when she reaches the goal, she can stop and she has no maximum velocity. In the relaxed version, Alice will always be able to get down a non-trivial hill faster than the original problem as she can `accelerate` for every action.

   To define the heuristic $h(s_i)$, we must determine how many times does Alice need to `accelerate` to travel from her current state to the chair lift?

To start, if Alice only accelerates from the top of the hill, her position will change as such:

$$1, 2, 4, 7, 11, 16, 22, ...$$

Using WolframAlpha, we can find a closed form for the number of accelerations required to reach the bottom of the hill given a current elevation:

$$e = \frac{1}{2} * (a^2 - a + 2)$$

Where e is the current elevation and a is the required accelerations. Finally, we must consider Alice's current velocity:

$$e = \frac{1}{2} * ((a + v)^2 - (a + v) + 2)$$

Thus the value of $h(s_i) = \lfloor a \rfloor$, where a is the positive value that satisfies the above equation (it is quadratic so there will be two a positive and negative solution).

This will be an admissible heuristic as it is derived from the relaxed version of the original problem where Alice doesn't have to take into account decelerating as she approaches the bottom of the hill.