Homework 1

CSE 3521 – Summer 2018

Instructor: Andrew R. Plummer

Department of Computer Science and Engineering
The Ohio State University

May 19th, 2018

Task: Work through each set of exercises. If you get stuck on any of the exercises you can ask Yi or myself for help by email or during office hours.

What to submit: Submit your answers for all of the exercises in this document to the appropriate dropbox on the Carmen site. Answers for the concept check and proof sections can be hand-written (e.g., submitted as a scanned image), but please make sure that your writing is readable. Answers to the coding section must be written in python and must be runnable by the grader.

Due date: Submit your answers to the Carmen dropbox by 11:59pm, Jun. 1st.

1 Concept check

1. (1 points) For each of the following search algorithms, state which kind of data structure models their collection of fringe/frontier nodes: breadth-first search, depth-first search, A^*-search.

2. (1 points) Provide an admissible and consistent heuristic for the state space graph shown in Figure 1.

3. (2 points) In general, given a non-zero-sum game setting, can we still make use of alpha-beta pruning? In either case, provide an argument and example illustrating your answer. To get started, try re-expressing the terminal values in the zero-sum game trees we've looked at using vectors where each component represents a value to each respective player.

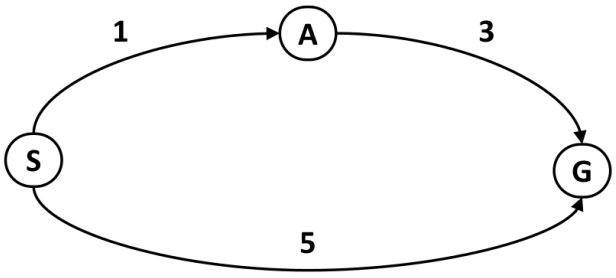

Figure 1: A simple state space graph (from Berkeley AI slides).

4. (2 points) In your own words, explain the relationship between how the expectimax algorithm works and how Q-states work in Markov Decision Processes (MDPs). Try to be as explicit as possible about the parallel between the two concepts.

5. (2 points) In your own words, explain the role of the "Markov assumption" in our formulation of MDPs, i.e., why are we making this assumption about our decision processes?

6. (2 points) In your own words, explain the key differences between value iteration and policy iteration for solving MDPs.

2 Coding

1. (4 points) [From RN, page 116] The "missionaries and cannibals" problem is usually stated as follows. Three missionaries and three cannibals are on one side of a river [let's assume the left side], along with a boat that can hold one or two people [and must hold at least one]. Find a way to get everyone to the [right] side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place. This problem is famous in AI because it was the subject of the first paper that approached problem formulation from an analytical viewpoint (Amarel, 1968).

   Write a python script that implements and solves the problem optimally using an appropriate search algorithm. Make sure your script produces an optimal path from the initial state to the goal state.

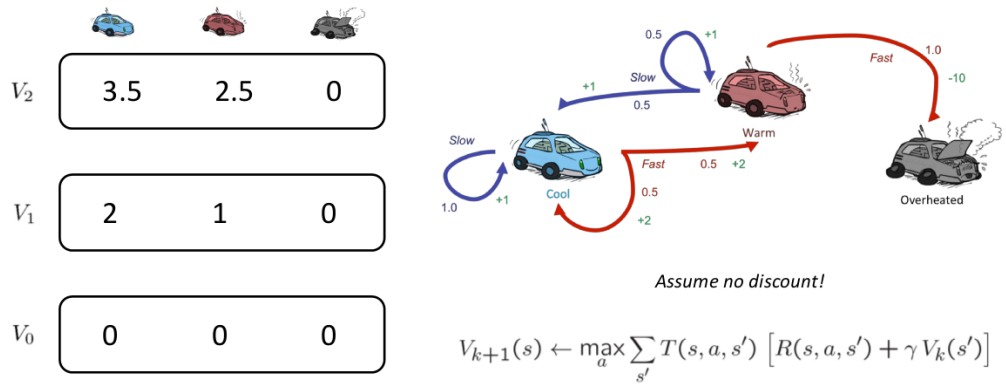

Figure 2: MDP for keeping cars happy (from Berkeley AI slides).

2. (4 points) Implementing A-star search. Using the pacman project resources from the Berkeley AI Materials site (http://ai.berkeley.edu/home.html), do Question 4 from the search project (http://ai.berkeley.edu/search.html#Q4). When you submit your answer, make sure you submit all the scripts needed to run your implementation.

3. (4 points) Implementing value iteration. Implement question 1 from the pacman project on reinforcement learning (http://ai.berkeley.edu/reinforcement.html#Q1). Ignore the autograder instructions since we're not going to use an autograder to grade the submissions. Again, when you submit your answer, make sure you submit all the scripts needed to run your implementation.

4. (4 points) Write a script to compute V_{18} and V_{19} for the cool and warm states in the MDP shown in Figure 2.

3 Fun with proofs

1. (2 points) Prove that every consistent heuristic is admissible. Show by example that there exist admissible heuristics that are not consistent.

2. (2 points) Assuming a discount γ where $0 < \gamma < 1$, prove that the value iteration values V_k converge for all states of an MDP as k increases toward infinity.