Assignment #2

E0503474 / Yu-Ting TSENG (with None Collaborator)

Oct 8, 2020

Problem 1: Markov Decision Process

(a) Assume that you are given a directed graph with n vertices and positive weights and would like to compute the shortest path from every vertex to a goal vertex g. Describe how to model the problem as a MDP so that it can be solved with an appropriately initialized value iteration algorithm, i.e. describe the state space, action space, transition function, and reward function.

State Space: n binary state variables (on the vertex or not) with size $O(2^n)$;
Action Space: arrows from a vertex to another with size $O(n^2)$;
Transition Function: $P(i, j) = \frac{w_{ij}}{\sum_{k \in out-degree} w_{ik}}$, if arrow from i to j exists, else 0;
Reward Function: the positive weights of edge.

(b) Assume that we now have M agents on the same graph and the reward is the sum of the rewards of the agents. Can value iterations be used to solve the problem efficiently (as a function of M) in the following two cases? Why?

i. There is no restriction on the number of agents that can be at each vertex.

In this case, the answer would be "yes". The size of state space for single agent is $O(2^n)$. Every agent takes action simultaneously and do not need to consider other, therefore we can do the value iteration for M times. In other words, the problem could be solved efficiently with value iteration.

ii. Only one agent can be at each vertex at any time, i.e. having more than one agent at a vertex incurs a very large penalty. Note that every agent must move to another vertex at every time instance.

In this case, the answer would be "no". We only consider the case that $m <= n$; the size of state space would be $O(n^M)$. The number of possible states grows exponentially as M grows. On the other hand, value iteration considered all possible states and therefore can't be solved efficiently.

(c) For the problem with M agents, the number of actions grows exponentially with M. Suggest one way to allow each trial of Monte Carlo Tree Search with UCT to run efficiently.

Start from initial state, we repeatedly give trials on each state, and decide the best action to take for each state sequentially. In this case, we can eventually get a sequence of actions we need by greedy algorithm.

Problem 2: Search Tree

Consider a MDP where the state is described using M variables where each variable can take n values. The MDP has 2 actions and at each state each action can only lead to 2 possible next states.

(a) What is the size of the state space of this MDP? Can this MDP be efficiently solvable with value iteration as M grows?

The size of state space is n^M. When M grows, the size would grow exponentially and lead to the difficulties for value iteration.

(b) A search tree of depth D (number of actions from the root to any leaf is D) is constructed from an initial state s. What is the size of the search tree (the number of nodes and edges) as a function of M and D, in O-notation? Can online search be done efficiently as M grows if D is a fixed small constant?

We first consider the situation generally (no constraints). The size of search tree is $O(|A|^D|S|^D)$. Although there is no more curse of dimensionality if the M grows with D fixed, the complexity still grows exponentially. As a result, online search can't be done efficiently.
However, in our task, the MDP has 2 actions and each action can only lead to 2 possible states, the size of the search tree is $O(2^D 2^D)$; the complexity is nothing matter to M, and thus online search can be done efficiently.

(c) MCTS is used for solving this MDP. What is the size of the search tree if T trials of MTCS is performed up to a search depth of D, as a function of M, D and T in O-notation?

We replace $|S|$ of the O-notation in the previous problem with T. We don't actually observe whole possible states, instead we do some trials and sampling. The size of the MCTS is actually $O(|A|^D T^D)$ in general case and $O(2^D T^D)$ in our task.

(d) Consider a search tree where the reward is zero everywhere except possibly at some leaves. When a MCTS trial goes through a node, we say that an action at the node wins if the trial ends in a leaf with reward 1. Consider an MCTS simulation where a node has been visited 16 times and has two actions, A and B. Action A has a won 2 out 4 times whereas action B has won 8 out of 12 times. Which action will the MCTS algorithm chose given the exploration parameter c is set to 1? Give the values of π_{UCT} for the node (consider log base 2 in UCT bound).

$\pi_{UCT} = \arg\max_a(\hat{Q}(n,a) + 1 * \sqrt{\frac{\log_2 16}{N(n,a)}}) = \arg\max_a(\hat{Q}(n,a) + \frac{2}{\sqrt{N(n,a)}});$
For action A, $\frac{1*2}{4} + \frac{2}{\sqrt{4}} = 1.5;$
For action B, $\frac{1*8}{12} + \frac{2}{\sqrt{12}} \approx 1.244;$
As a result, we might choose action A.