Practice Exam C – Solutions

1 Driving

1. (20 points) You are driving an electronic vehicle over a road network that connects up a set of small towns. You know the locations of the towns and the road map, which contains at most one road between each pair of distinct towns. The roads are all two-way.

   - The length of each road segment r is $length(r)$.
   - Your battery can hold m units of charge, maximum.
   - Every mile requires 1 unit of charge.
   - You drive 1 mile per minute.
   - Some towns have chargers that charge at the rate of 10 units per minute, and some have no charger at all.
   - If you run out of charge you can no longer continue.
   - Some roads have cell service and some do not. Your car is so modern it does not have a radio, so if there's no cell service you will be bored. When you are bored, your perceived time goes slower — you will perceive time twice as long as the actual time. (You won't be bored when charging the vechicle, since you can always walk around the town).
   - You would like to find a path (including charging stops) to a goal town that **minimizes your overall perceived time**.

   Formulate this as a min-cost-path problem, where the action space consists of $drive(r)$ where r is a road segment connecting two towns, and $charge(t)$ where $t \in \{10, 20, \ldots, 100\}$ is a duration of charging in minutes.

   (a) (4 points) What is a minimal space of states for this problem?

   Solution: All possible values of (town, battery-charge)

   (b) (4 points) In what states s is the action $drive(r)$ executable?

   Solution: When the town in s is equal to the start town of r and $length(r)$ is less than or equal to the battery charge in s.

   (c) (4 points) What is the cost function $C(s, a, s')$ (in units of minutes)?

   Solution: If $a = drive(r)$ then $length(r)$ if r is not boring, else $2 \cdot length(r)$. Otherwise, for charge action with argument t the cost is t.

   (d) (4 points) Approximately what is the computational complexity of uniform-cost search in this problem, assuming n towns, m battery levels, every town has k roads connecting to it, and the optimal path has d actions?

   Solution: UCS/Dijkstra runs in $O((|V| + |E|) \log |V|)$.
   The states are pairs of town and battery level, so $|V| = O(nm)$.
   From each state (t, b) there are about k drive actions and m charge actions, giving branching factor $b = k + m$. Thus each state has up to $k + m$ outgoing edges, so $|E| = O(nm(k + m))$.

Substituting into $O((|V|+|E|)\log|V|)$ gives $O((nm+nm(k+m))\log(nm))$, without assuming $E \gg V$ or $V \gg E$.
If the optimal solution has depth d, UCS may instead only explore a tree with branching factor $b = k + m$, giving $O(b^d)$.
Therefore the best approximate bound is $O(\min(b^d,\ (|V|+|E|)\log|V|)) = O(\min((k+m)^d,\ (nm+nm(k+10))\log(nm)))$.

For each of the following heuristics, indicate whether it is admissible and whether it has a smaller search space than the original problem.

(e) (2 points) Minimum cost path if the battery level never decreases.
√ **admissible** √ **smaller search space than original** ◯ neither is correct

(f) (2 points) Minimum cost path when the problem is the same as the original, but the penalty for boredom is twice as high.
◯ admissible ◯ smaller search space than original √ **neither is correct**

2 PDDL

2. (12 points) Let's continue the theme of keys and rooms, but this time writing operator descriptions. Use the following predicate definitions, with the obvious interpretations when `k` is a key and `r` is a room.

```
(opens ?k ?r)
(connected ?r1 ?r2)
(unlocked ?r)
(robot-in ?r)
(robot-holding ?k)
(in ?k ?r)
```

(a) (4 points) Here is an operator describing how a robot can move from room to room. In order to move between rooms, the rooms must be connected and the destination unlocked.

```
(:action move
    :parameters (?r1 ?r2)
    :precondition (and (connected ?r1 ?r2) (unlocked ?r2) (robot-in ?r1))
    :effect (and (robot-in ?r2) (not (robot-in ?r1))))
```

Complete the operator description for unlocking a room. In order to unlock a room, the robot must be in a connected room and holding a key to the room it is unlocking.

```
(:action unlock
    :parameters (?r1 ?r2 ?k)
    :precondition


    :effect


)
```

Solution:

```
(:action unlock
    :parameters (?r1 ?r2 ?k)
    :precondition (and (connected ?r1 ?r2) (robot-in ?r1)
                       (opens ?k ?r2) (robot-holding ?k))
    :effect (and (unlocked ?r2)))
```

(b) (8 points) Assume there is one more operator

```
(:action pick-up
    :parameters (?r ?k)
    :precondition (and (robot-in ?r)
                       (in ?k ?r))
    :effect (and (robot-holding ?k) (not (in ?k ?r))))
```

Here is an initial state:

```
(connected R0 R1) (connected R1 R2) (connected R2 R3)
(connected R1 R0) (connected R2 R1) (connected R3 R2)
(unlocked R1) (unlocked R2) (unlocked R3)
(robot-in R1) (opens K0 R0) (in K0 R3)
```

And goal: `(and (robot-in R0) (unlocked R0) (robot-holding K0))`

We want to base a heuristic on a relaxed plan graph for this problem.

i. First, what is the shortest solution for this problem? Please list the actions.

> **Solution:**
> `(move R1 R2) (move R2 R3)`
> `(pick-up R3 K0) (move R3 R2) (move R2 R1)`
> `(unlock R1 R0 K0) (move R1 R0)`

ii. At what level in the RPG is the fluent `(robot-holding K0)`? **3**

iii. At what level in the RPG is the fluent `(unlocked R0)`? **4**

iv. What is the value of H_{add} for this goal? **12**

v. What is the value of $H_{\max}$ for this goal? **5**

3 Searching for Answers

3. Are each of the following claims true or false? Provide a short justification.

(a) (2 points) A* search is equivalent to greedy best-first search if the heuristic is 0 everywhere.

◯ True √ **False**

Solution: In this case, A* will become equivalent to UCS.

(b) (2 points) If a solution to a motion planning problem exists, then RRT will eventually find it.

√ **True** ◯ False

Solution: It is probabilistically complete, and the sampling guarantees that eventually it can find any path.

(c) (2 points) If a solution to a motion planning problem exists, then path optimization will eventually find it.

◯ True √ **False**

Solution: As a gradient-based method, it can get stuck in local optima.

(d) (2 points) On a deterministic min-cost path problem, Monte-Carlo Tree Search (MCTS) has the same worst-case runtime as breadth-first search.

◯ True √ **False**

Solution: The computational complexity of MCTS is hard to analyze, but it will generally do a lot of repeated work, in general.

(e) (2 points) In a large discrete search problem with a high branching factor, MCTS can be a better choice than A* search, even if we use a good heuristic for A* search.

√ **True** ◯ False

Solution: MCTS can take advantage of locality properties in the problem to get additional hints from previous iterations about where to concentrate its search effort.

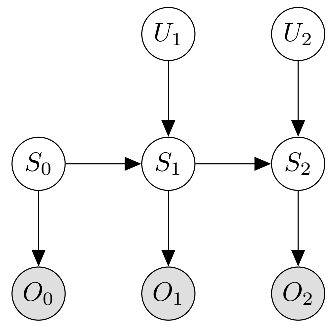
(f) (2 points) Suppose that for any two states in a discrete search problem, there is exactly one path from the first state to the second. This property alone does not tell us anything about which search algorithms might be better or worse to use.

◯ True √ **False**

Solution: In a problem like this, the dynamic programming aspects of search algorithms won't help. So, MCTS or even BFS with no visited list could be good.

4 Injection sampling

4. You are a process engineer, working on a system that is best described with three variables at each time step: U_t, S_t, O_t, except for U_0. Assume all variables are binary. The Bayes net for three steps looks like this:



In several parts of the following questions, we ask for code, but don't worry *at all* about details of syntax, etc. It is fine to write in pseudocode, instead.

(a) (3 points) We'd like to perform *ancestral sampling* to compute a single sample from the joint distribution of an instance of this model with $T = 3$ time steps. Specify a valid ordering of the random variables to sample from.

> **Solution:** There are multiple possible solutions. We just need a topological ordering. One of them is $U_1, U_2, S_0, S_1, S_2, O_0, O_1, O_2$.

(b) (5 points) You're interested in exploring how to make all the observations be 1. Your first step is to try to find the distribution over values of U_1 and U_2 in that case. That is, you want to find $P(U_1, U_2 \mid O_0 = 1, O_1 = 1, O_2 = 1)$. Sketch a program for doing this using rejection sampling with N samples, making use of `ancestral_sample`. Assume `ancestral_sample(T)` returns a length T list of states, a length T list of observations, and a length $T-1$ list of controls. Return a normalized distribution.

```
counts = {(0, 0) : 0, (0, 1) : 0, (1, 0) : 0, (1, 1): 0}
```

Solution:

```
counts = {(0, 0) : 0, (0, 1) : 0, (1, 0) : 0, (1, 1): 0}
for i in range(N):
    states, obs, controls = ancestral_sample(3)
    if obs[0] == obs[1] == obs[2] == 1:
        total += 1
        counts[controls] += 1
return {o : c/total for (o, c) in counts.items()}
```

(c) (3 points) Now we are interested in doing importance sampling. To implement ancestral sampling, we only need to sample from the conditional distribution at each variable node. To implement importance sampling, what additional information about each node would be needed?

Solution: We need the probability values from the conditional probability distributions (they could be unnormalized) to compute importance weights.

(d) (9 points) Willard thinks we should use Gibbs sampling to answer our question $P(U_1, U_2 \mid O_0 = 1, O_1 = 1, O_2 = 1)$. Assume there are no 0's in any of the potentials in our system, so the underlying Markov chain in Gibbs sampling is ergodic.

i. (3 points) The random variables in our problem are $S_0, S_1, S_2, O_0, O_1, O_2, U_1, U_2$. How should we initialize them?

> **Solution:** Assign $O_0 = 1, O_1 = 1, O_2 = 1$. The others can be initialized randomly (or arbitrarily) assuming there are no 0's in the potentials.

ii. (6 points) Assume we have some current assignment to these values, and we select S_1 to re-sample. What distribution should we draw from? Specify your answer symbolically, in terms of the conditional probabilities in our model. Your answer may be unnormalized.

> **Solution:** We draw from the distribution conditioned on S_1's Markov blanket.
>
> $$\begin{aligned} &P(S_1 \mid S_0, S_2, U_1, U_2, O_0, O_1, O_2) \\ &\quad \propto P(S_0, S_1, S_2, U_1, U_2, O_0, O_1, O_2) \\ &\quad = P(S_0)P(U_1)P(U_2)P(S_1 \mid S_0, U_1)P(S_2 \mid S_1, U_2)P(O_0 \mid S_0)P(O_1|S_1)P(O_2 \mid S_2) \\ &\quad \propto P(S_1 \mid S_0, U_1)P(S_2 \mid S_1, U_2)P(O_1|S_1) \end{aligned}$$