CPSC 322 **Artificial Intelligence I** **Fall 2020**

Sample Midterm Exam #1

Solution

1 Agents

(a) (i) [3 marks] As an input to the agent, what is "abilities"?
(ii) [5 marks] Suppose Pat claimed that the abilities should not be one of the inputs to the agent, as it does not affect the actions. Give an example showing why Pat is wrong.

Solution (i) The abilities is the set of actions available to the agent.
(ii) Think of an example where the best actions become available or not available. E.g., If the robot had the ability to move through walls it would act differently than if it had to go around. E.g. If an agent has the ability to solve the goal in one step, it would do that. (One example that is clear is better than multiple examples).

(b) [5 marks] Describe the difference between a fully-observable deterministic system and a partially-observable stochastic system.

Solution In a fully-observable deterministic system the agent knows the state of the world and can predict the effects of its action. In a partially-observable stochastic system, the agent doesn't know what state it is in, and even if it did, it would not know the effect of its action.

(c) [5 marks] What is the "planning horizon"? Describe the difference between a finite stage and indefinite stage planning horizon.

Solution The planning horizon is how far into the future the agent considers when planning. In a finite stage planning horizon the agent considers a fixed finite number of time steps ahead. In a indefinite stage planning problem the agent considers a finite number of steps ahead, but the number of steps depends on the problem (e.g., in a search problem, we don't know how many arcs we will need to consider before starting).

(d) [5 marks] Give a specific examples of an ordinal preference. How is an ordinal preference different from a goal?

Solution Example of ordinal preference: I prefer a challenging exam in which I learn something than a simple exam that is boring. I would prefer to get an A than a C.
A goal is a condition that an agent acts to achieve. Whereas a ordinal preference specifies a preference. The agent would try to get what is more preferred out of those that are possible, even if it may not be able to achieve its most preferred, whereas a goal is all-or-nothing.

2 Search

(a) [5 marks] Explain what the frontier is in a graph-search problem, and what it is used for.

Solution The frontier is a set of paths from the start node to intermediate nodes which are stored by the search algorithm. The frontier is used for determining the next node that will be explored in the search.

(b) [3 marks] Is the worst-case time complexity different for depth-first search and breadth-first search? (You must be specific about what the complexity is with respect to.) Why or why not?

Solution Yes. On infinite graphs or graphs with cycles, DFS may not terminate, and so has infinite time complexity, but breadth-first search is exponential in the path length. Note: you must say that it is exponential in path length, not just exponential.

(c) [5 marks] Consider the following generic search algorithm:

```
1   **Input:** a graph,
2          a set of start nodes,
3          Boolean procedure $goal(n)$ that tests if $n$ is a goal node.
4   $frontier := \{\langle s \rangle : s \text{ is a start node}\}$;
5   **while** *frontier* is not empty:
6          **select** and **remove** path $\langle n_0, \ldots, n_k \rangle$ from *frontier*;
7          **if** $goal(n_k)$
8             **return** $\langle n_0, \ldots, n_k \rangle$;
9          **for every** neighbor $n$ of $n_k$
10            **add** $\langle n_0, \ldots, n_k, n \rangle$ to *frontier*;
11  **end while**
```

Imagine that this generic algorithm was used as the basis for an implementation of depth-first search and of breadth-first search. Which line or lines of the pseudocode above *must* have different implementations? Briefly, how would those implementations differ?

Solution Only line 6 must be different. DFS must select one of the most recently added paths; BFS must select one of the least recently added paths.

(d) [4 marks] In what sense is branch and bound better than A*? In what sense is A* better than branch and bound?

Solution A* is better in the sense that it expands fewer nodes (up to tie breaking); to be as good branch-and-bound needs a perfect initial guess of the bound. Branch and bound is better in the sense that it uses less memory.

(e) [3 marks] Is the maximum of two admissible heuristics also admissible? Why or why not?

Solution Yes, the max of two admissible heuristics is itself admissible, because each of the two heuristics is guaranteed to underestimate the distance from the given node to the goal, and so therefore must their max.

Is the sum of two admissible heuristics also admissible? Why or why not?

Solution No. Suppose h were admissible, then $h + h$ would also me admissible, which means that any constant times h would also be admissible, but if it is not zero, then for some constant, the product will be greater than the actual cost.

(f) [10 marks] Consider the cyclic delivery graph of Figure 3.7 of the textbook, with the heuristic function of Example 3.13. [Of course, in the exam, we will give you the graph and the heuristic costs.] Suppose the start node is $b1$ and the goal is $r123$.

For each iteration of A^*, a trace would show the path removed from the frontier and the path(s) added to the frontier to get from $b1$ to $r123$. We have started the trace, please continue it. You only need to show **three iterations**; that is, for the next three values selected from the frontier show which paths are added to the frontier (together with their *cost*-value, h-value and f-value). [The path number has no meaning; it is just a number so you can refer to it.]

Path Number	Path	*cost*	h	f
1.	$b1$	0	18	18
Remove path number 1.				
2.	$b1 \rightarrow b2$	6	15	21
3.	$b1 \rightarrow b3$	4	17	21
4.	$b1 \rightarrow c2$	3	10	13

Solution

Path Number	Path	*cost*	h	f
Remove path number 4.				
5.	$b1 \rightarrow c2 \rightarrow c1$	7	6	13
6.	$b1 \rightarrow c2 \rightarrow c3$	9	12	21
Remove path number 5.				
7.	$b1 \rightarrow c2 \rightarrow c1 \rightarrow c2$	11	10	21
8.	$b1 \rightarrow c2 \rightarrow c1 \rightarrow c3$	15	12	27
Remove path number 6.				
9.	$b1 \rightarrow c2 \rightarrow c3 \rightarrow c2$	15	10	25
10.	$b1 \rightarrow c2 \rightarrow c3 \rightarrow c1$	17	6	23

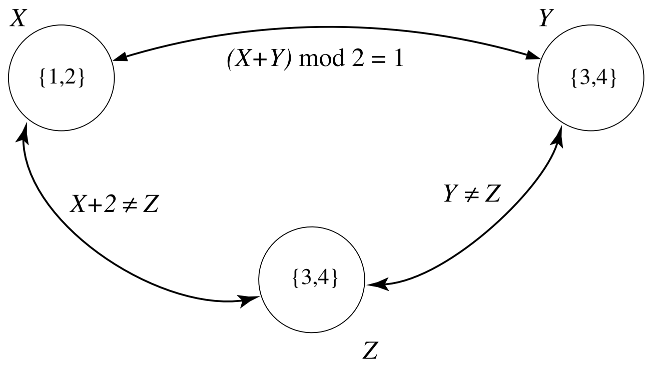
It is also possible to remove path 7 at the last step. With multiple-path pruning or cycle checking, it would be removed. Without multiple-path pruning, it would be expanded with a cycle.

(g) [3 marks] For this particular example, when is the first time that multiple-path pruning will prune a path? Be specific about which path is pruned and why.

Solution When removing path 7, it can be pruned, as there is already a path to c2 that has been expanded (path 4).

3 Constraints

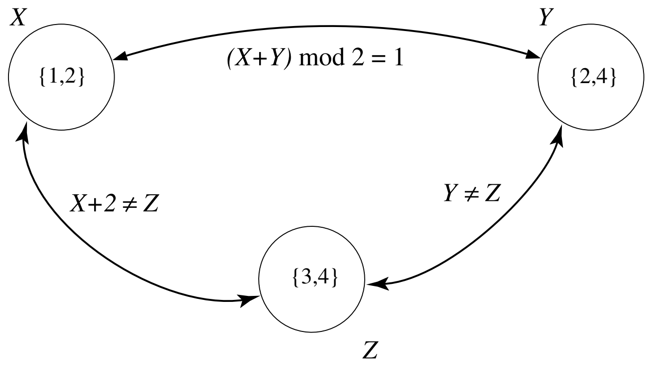
(a) [5 marks] Consider the following constraint graph. Note that $(X + Y) \bmod 2 = 1$ means that $X + Y$ is odd.



Is it arc consistent? If it is, explain why. If it isn't, which domain element(s) that can be pruned? For each domain element(s) that can be pruned, give the arc (or arcs) that can be used to prune it.

Solution Yes it is arc consistent. For every value of X, there is a value for Y such that $X + Y$ is odd (for $X = 1$ there is $Y = 4$, and for $X = 2$ there is $Y = 3$). Similarly for every Y there is an X. For each X there is a Z such that $X + 2 \neq Z$ (for $X = 1$ there is $Z = 4$ and for $X = 2$ there is $Z = 3$), and for each value of the Z there is a value for X. For each value of Y and Z there is a value of the other that is different (for 3 there is 4, and for 4 there is 3).

(b) [5 marks] Consider the following constraint graph:



Is it arc consistent? If it is, explain why. If it isn't, which domain element(s) that can be pruned? For each domain element(s) that can be pruned, give the arc (or arcs) that can be used to prune it.

Solution No it is not arc consistent. For $X = 2$ there is no value for Y such that $X + Y$ is odd. Thus 2 can be pruned from the domain of X.

(c) [3 marks] Why does randomization help in local search?

Solution Randomization can help escape from local minima, whereas if there was no randomization the search may cycle.

(d) [3 marks] Explain how the next assignment is selected in simulated annealing.

Solution The next assignment is obtained by selecting a variable at random and selecting a value for that variable at random. If it is not worse, that assignment is accepted. If it is worse, whether it is accepted probabilistically depends on how much worse it is, and a temperature parameter.

(e) [5 marks] What is a runtime distribution? What is on the x-axis? What is on the y-axis? How is a runtime distribution constructed?

Solution A runtime distribution plots the distribution of how long various runs of the same parameter setting for the same problem take. One the x-axis is the number of steps or the run time. On the y-axis is the number (or proportion) of run that were solved in that runtime. It is constructed by running the algorithm for a number of trials, sorting the runtimes, and then plotting the runtime with the index of the trial in the sorted list.

(f) [3 marks] How can a runtime distribution be used to show that one local search algorithm dominates another? If one algorithm dominates another is it always better?

Solution One algorithm dominates another if it is above and to the left of the other algorithm. It is not always better unless every point of one is to the left of every point of the other; otherwise some of the runs of the dominated one may be faster than the other.

(g) [2 marks] Given a runtime distribution, how can the median runtime be determined?

Solution The median runtime can be determined by finding the value on the x-axis that corresponds to the halfway down the y-axis. (So that half of the runtimes are above it and half are below it.) [It is probably easier to draw – and explain – a diagram.]

(h) [3 marks] Why is the mean runtime not a good measure for the performance of a local search algorithm?

Solution All of the setting where there is a chance they will get stuck in a loop (and so never halt) will have infinite mean, and so cannot be compared among each other and will be worse than any algorithm that eventually finds a solution, no matter how slow it is.