

Feedback — Foundation Level - Mid Course Quiz

You submitted this quiz on **Fri 8 Mar 2013 12:30 PM UTC**. You got a score of **37.00** out of **37.00**. However, you will not get credit for it, since it was submitted past the deadline.

Note that this mid course assessment lets you monitor your progress, but the final week 5 assessment will be the one that counts.

Question 1

Tick those statements that you believe to be true:

Marks available: 4

Your Answer	Score	Explanation
☒ All types of best-first search algorithms use an evaluation function to determine the order in which nodes in the search space are explored.	✓ 1.00	That is what characterizes best-first search.
☒ A* always finds a minimal-cost solution if the given heuristic is admissible.	✓ 1.00	Yes.
☐ A* graph search is always more efficient than A* tree search.	✓ 1.00	Ideally, we would define the search problem in such a way that it spans a search space that is a tree. See the 8-queens problem as an example. In this case there is no need to check for repeated states and tree search is more efficient.
☐ In A* search, the value of the evaluation function is the estimated distance to the goal.	✓ 1.00	In A*, the evaluation function for a node n is the estimated cost of the cheapest path from the initial state through n to a goal state.
Total	4.00 / 4.00	

Question 2

A* Tree Search

Apply the A* search algorithm to the touring Romania problem, this time starting from Timisoara. To do this, you will need the [map](#) and the [heuristic function](#) defined for this problem. Apply the tree-search version of the algorithm, that is, treat repeated occurrences of the same state as new search nodes.

To generate the answer, make a string out of the first letter of each city name in the order in which the nodes are expanded.

The string will start with T for Timisoara. The next letter will be A or L, depending on which node the algorithm will expand next. The last letter in the string should be B for Bucharest. Note, letters should be added for nodes that are expanded, not for each successor that is generated. Don't leave any spaces between the letters.

Marks available: 6

You entered:

TLMALDSRFPB

Your Answer	Score	Explanation
TLMALDSRFPB	✓ 6.00	
Total	6.00 / 6.00	

Question Explanation

A* expands nodes in the following order with the given, increasing f -values:

1. Timisoara (329) - the initial state
2. Lugoj (355) - the southern route looks better
3. Mehadia (422)
4. Arad (484) - now the northern route looks better

5. Lugoj (495) - A* expands a cyclic path!
6. Debrete (498) - back on the southern route
7. Sibiu (511) - back north
8. Rimnicu Vilea (531)
9. Fagaras (533)
10. Pitesti (535)

The queue now contains two nodes at the lowest f -value of 536, namely, Craiova and Bucharest. If Craiova is selected first, it will be expanded next, followed by Bucharest. Otherwise Bucharest is a solution node to be expanded and A* terminates.

Question 3

Tick those statements that you believe to be true:

Marks available: 5

Your Answer	Score	Explanation
☐ States in PDDL are black boxes, where we cannot see the internal structure.	✓ 1.00	No, states are sets of first-order atoms representing relations between objects.
☒ The set of all possible states in the STRIPS representation is finite.	✓ 1.00	Yes. A finite number of objects related by a finite set of relations is finite, and so is its power set.
☒ (holding crane1 cont1) and (empty crane1) can hold in the same state in PDDL.	✓ 1.00	Since PDDL is not aware of the semantics of the two relations, both can hold.
☒ In the STRIPS representation, the closed world assumption is adopted.	✓ 1.00	It is. What is not mentioned in a state is considered false.
☒ Forward state-space search is always ground, i.e. there is no "lifted" forward search.	✓ 1.00	Yes. All parameters of an operator are expected to be in the preconditions, which means they are ground when we show applicability of an action.
Total	5.00 / 5.00	

Question 4

For this question, you need look at a random planning **domain** and **problem**. Note that these are different from the random domain you may have looked at previously. Which of the following actions are applicable in the initial state?

Marks available: 5

Your Answer	Score	Explanation
☒ (op1 C B B)	✓ 1.00	All preconditions are satisfied.
☐ (op2 B B C)	✓ 1.00	Neither precondition is not satisfied.
☐ (op1 B B C)	✓ 1.00	The precondition (R ?x3 ?x3) is not satisfied.
☒ (op2 C B B)	✓ 1.00	All preconditions are satisfied.
☒ (op2 A A C)	✓ 1.00	All preconditions are satisfied.
Total	5.00 / 5.00	

Question 5

Tick those statements that you believe to be true:

Marks available: 4

Your Answer	Score	Explanation
☒ Forward state-space planning and plan-space planning solve the same problem.	✓ 1.00	Yes.

☒ Partial plans must not have inconsistent ordering and variable binding constraints during the search.	✓ 1.00	A plan with inconsistent constraints can never be refined into a solution plan, so there is no need to consider it in the search.
☐ Partial plans must not have flaws during the search.	✓ 1.00	Partial plans have flaws that must be resolved by the planning algorithm. Plans without flaws are solution plans.
☐ Forward state-space planning and plan-space planning search the same space.	✓ 1.00	No, in one case the search states are world states, in the other they are partial plans.
Total	4.00 / 4.00	

Question 6

How many threats are there in this [partial plan](#)? - To answer this question, you may need to look at the operator definitions in the [DWR domain](#).

Marks available: 3

You entered:

2

Your Answer	Score	Explanation
2	✓ 3.00	
Total	3.00 / 3.00	

Question Explanation

Correct answer is 2.

Threats are on causal links, so we can look at them in turn:

- `in(...)`: This relation can only be threatened by a take action (the only operator that has `not in(...)` as an effect), but there is no such action in the partial plan.
- `holding(...)`: This relation can be threatened by a load action, of which there are none in the plan, or a put action, of which we have two. However, the put action cannot unify with the causal link from the other branch as the two containers are given and different.
- `at(...)`: This relation can only be threatened by a move action, of which we have two. As we do not know the exact robot or location the unification is possible and both causal links are threatened by the move action in the other branch.

Marks available: 3

Question 7

How many flaws are there in the same [partial plan](#)? - To answer this question, once again you may need to look at the operator definitions in the [DWR domain](#).

Marks available: 3

You entered:

20

Your Answer	Score	Explanation
20	✓ 3.00	
Total	3.00 / 3.00	

Question Explanation

Correct answer is 20.

There are two threats. In addition, there are many unsatisfied sub-goals, namely all preconditions for all actions minus one for four of them. So, there 2 x 3 preconditions for the two move actions, 2 x 3 preconditions for the two unload actions, and 2 x 3 preconditions for the two put actions. That is, 2 threats plus 18 open sub-goals.

Question 8

Tick those statements that you believe to be true:

Marks available: 5

Your Answer	Score	Explanation
☒ An action accomplishes a task in a given state if and only if the action name matches the task name and the action is applicable in the given state.	✓ 1.00	That's the definition.
☐ A method is primitive if it refines a task into at most one sub-task.	✓ 1.00	Tasks and task networks can be primitive, but we never talked about primitive methods.
☒ Applying the decomposition function corresponds to a plan refinement operation in the STN search space.	✓ 1.00	Yes
☐ The solution to an STN planning problem is a decomposition tree.	✓ 1.00	No, its a plan, that is, a totally or partially ordered set of actions.
☐ The Ground-TFD and the Ground-PFD algorithms explore the same search space.	✓ 1.00	No, TFD searches for a totally ordered plan, whereas PFD searches for a partially ordered plan. Hence the difference in the first letter.
Total	5.00 / 5.00	

Question 9

Suppose your current computer can solve planning problems that require up to 50 steps in the solution. Your planning algorithm takes roughly 2^l time to solve problems, where l is the length of the plan. Now, suppose you are given a computer that is 1000 times faster than your current one. How long will the plans be that you can find with this new computer, assuming the algorithm is the same?

Round your answer to the nearest integer.

Marks available: 2

You entered:

60

Your Answer	Score	Explanation
60	✓ 2.00	
Total	2.00 / 2.00	

Question Explanation

Correct answer is 60. Your new computer is about 2^{10} times faster, meaning the solvable problem size increases by ten.