NATIONAL UNIVERSITY OF SINGAPORE

CS3243 – Introduction to Artificial Intelligence

(Semester 1: AY2019/20)

Mid-term Assessment

Time Allowed: 1 Hour

INSTRUCTIONS TO STUDENTS

1. Please write your Student Number only. Do not write your name.
2. This assessment paper contains **XXXX** questions and comprises **XXXX** printed pages.
3. You are required to answer **ALL** questions.
4. Write your answers in the space provided.
5. This assessment is **CLOSE-BOOK**; you are only allowed to carry one A4 size "cheat-sheet."
6. You may **use pencils** to write your answers but write legibly.

STUDENT NO:	**A**								

This portion is for examiner's use only

Question	**Marks**	**Remarks**
Q1	**/6**	
Q2	**/4**	
Q3	**/4**	
Q4	**/6**	
Total	**/20**	

Q1. [1 * 6 = 6 marks] **Circle** the correct answer in each case. Explain your choice briefly

0.5 points: Getting T/F correct
0.5 points for getting explanation right.

1. A^* graph search with consistent heuristic is guaranteed to expand no more nodes than the uniform cost search.	True False
Reason: The heuristic could help to guide the search and reduce the number of nodes expanded. In the extreme case where the heuristic function returns zero for every state, A* and UCS will expand the same number of nodes. In any case, A* with a consistent heuristic will never expand more nodes than UCS. Anything resembling in the worst case h=0 => A* is UCS is given 0.5. All else 0	
2. The DBS digibot interacting with a customer fits into both "acting humanly" and "acting rationally" views of AI.	True False
Reason: The bot tries to be as human as possible in its responses. Its responses (decisions to respond in a particular manner) are based on the information the user provides and the data the bot has in hand Anything resembling best response to keep human happy + respond like human has also been awarded 0.5	
3. One can easily specify the performance measure for an agent, based on what (s)he wants the agent to do in the task.	True False
Reason: In general, defining a performance measure is hard. If we design the performance measure based on what the agent must do, we may end up with unexpected behavior from the agent. E.g., the sequence: dump garbage and clean, can be executed repeatedly to get rewarded for picking up garbage. Any other reason to say defining performance measure is hard (e.g., cannot consider all possible outcomes etc.,) have also been given 0.5	
4. We can always extend the solution to a search problem using state abstraction to the real world.	True False
Reason: You may have a lossy abstraction that doesn't expand to the real world state/action. Hence, not all solutions using state abstraction can be extended to the real world.	

Anything that indicates real world is complex have been awarded 0.5	
5. Assuming left-to-right ordering for alpha-beta pruning, always the leftmost branch is evaluated	True False
Reason: There is no other alternative to prune before we evaluate the leftmost branch. Hence, we cannot prune the leftmost branch. If you gave specific contrieved examples of right-most tree and its left subtree and said it will be pruned, it is incorrect. Hence haven't been given points. In all other cases, you should have got 0.5	
6. If a CSP is arc consistent, it can be solved without backtracking.	True False
Reason: Arc consistency may not determine the entire solution. With AC domains are fixed, you may still need to backtrack to identify the exact solution. If this doesn't come out in your answer, you haven't been given marks.	

Q2. [4 marks]

1. You are tasked to benchmark the running time and memory required to run some search algorithms. In your implementation, a node requires 1024 bytes. The computer you have can process 1024000 nodes per second. The agent can take ten actions in each state. Now calculate:

   For (a) and (b), given that the question missed the important detail on whether graph search or tree search is used has not been given, either way of treating the problem has been given full marks.

   If you have missed the units, 0.5 points are cut.

   a. Memory required for BFS, search depth of 8.
      Number of nodes: $b + b^2 + \cdots + b^8 = 10 + 10^2 + \cdots + 10^8 = 111111110$
      Space required for each node: 1024 bytes;
      Total space required: $111111110 \times 1024 =$ 113,777,776,640 bytes ≈ 108506.94 MB ≈ 105.96 GB.

   b. Memory required for DFS, search depth of 8.
      Number of nodes: $bm = 10 \times 8 = 80$
      Space required for each node: 1024 bytes;
      Total space required = $80 \times 1024 = 80$ KB.

   c. Comment on the time overhead/advantage of IDS over BFS, search depth of 6.
      Running time for BFS, search depth of 6.
      Number of nodes: $b + b^2 + \cdots + b^6 = 10 + 10^2 + \cdots + 10^6 = 1111110$.
      The computer can process: 1024000 nodes per sec;
      Time required: $\frac{1111110}{1024000} \approx 1.085\ seconds$
      BFS search depth 6 needs $1.085\ seconds$.
      Running time for IDS, search depth $1 \cdots 6$.
      Number of nodes: $(d+1)b^0 + db^1 + (d-1)b^2 + (d-2)b^3 + (d-3)b^4 + (d-4)b^5 + (d-5)b^6$. Now substitute $d = 6$
      $$\Rightarrow 7 + 6 \times 10 + 5 \times 100 + 4 \times 1000 + 3 \times 10000 + 2 \times 100000 + 1000000$$
      $$= 1{,}234{,}567$$
      Time required: $\frac{1234567}{1024000} \approx 1.205\ seconds$
      Hence, there is an overhead of using IDS. Overhead: $\frac{1.205 - 1.085}{1.085} = 0.11$

      For (c) just giving calculations did not get full marks. You need to say which one has a overhead.

Q3. [2 + 2 = 4 marks]

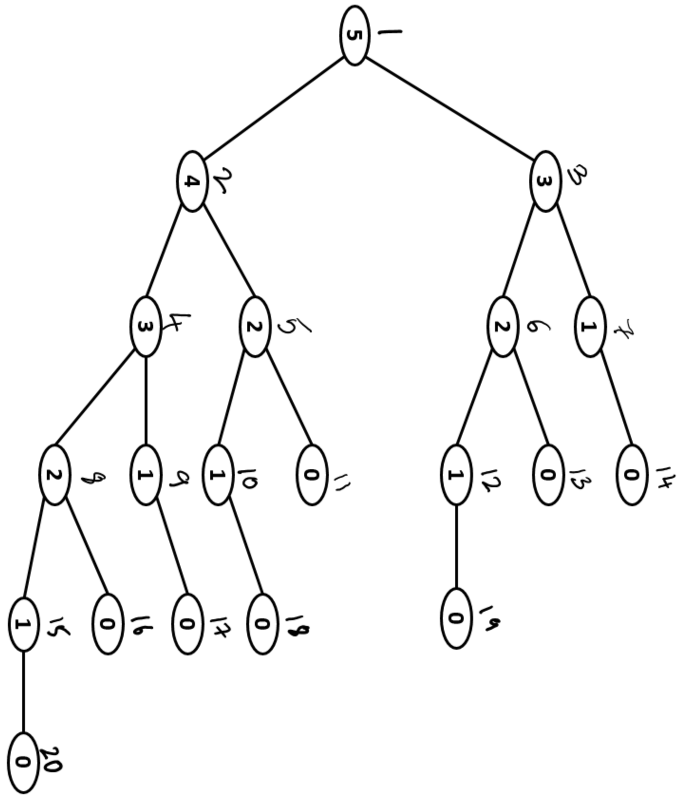
1. Let $h_1(s)$ be an admissible heuristic for the A^*. Let $h_2(s) = 2\,h_1(s)$. Prove that the solution found by A^* tree search with h_2 is guaranteed to have a cost at most twice as much as the optimal path.

The cost $\bar{g}$ of a non-optimal solution when popped is the f-cost. The prefix of the optimal path to the goal has a f-cost of $g + h_2 = g + 2h_1 \leq 2(g + h_1) \leq 2C^*$, with C^* being the optimal cost to the goal. Hence we have that $\bar{g} \leq 2C^*$and the path is at most twice as long as the optimal path.

If you have argued using $f = g + h$ and the solution has a sense of what is stated above, you have been given full points. If you only managed to say $h_2 = 2h_1$, no points (that is given).

If you mentioned about h^* and argued that $h_1 < h^*$ so $2h_1 < 2h^*$, it is a weaker argument and you have been given 1 point unless you captured the essence of above.

2. Consider a variant of NIM game, where the players are allowed to draw either one or two sticks from a single pile. The extensive form of the game is as shown in the figure below.



A zero (0) in the node indicates the player doesn't have anything to draw from the pile and hence loses the game. The actions are $draw1$ or $draw2$. As always, the max player starts the game.
Write the optimal strategy from the max player's perspective. Use the numbers beside the node to refer to the node.

Ans:$\langle 1, draw2\rangle, \langle 4, draw1/draw2\rangle, \langle 5, draw2\rangle, \langle 15, draw1\rangle, \langle 6, draw2\rangle, \langle 7, draw1\rangle$

Remember from the lecture that optimal stratey need to include all the possible states (nodes) and corresponding actions, irrespective of the agent reaching the state or not. In light of that if you only wrote nodes corresponding to the “winning” moves you have been given part points. Else you get full.

Q4. [2 + 4 = 6 marks]

You have developed a search-based game AI engine. Search for game solutions and execution happen via a pipeline as follows.

Step 1: Data processing unit parses the previous game log to abstract information for the search agent.

Step2: Develop a search tree (let us call it a model) based on the data from Step 1.

Step 3: The solution is executed on a new instance of the game once the full search tree is computed by the agent.

There are resource restrictions, and you cannot process all of the data at once.
You are working with a dataset of size 50 GB, the maximum size of the system memory is 24 GB. However, the system needs 9 GB for its normal operations. You need to adopt a batch processing approach where you process one chunk of data and generate a part of the search tree at a time. The search tree can be glued together in the end before the execution starts.
Parsing requires 1.5 times the memory of the actual data. Once parsing finishes, you can unload the actual data from memory. Data parsing takes 6 minutes for every GB of data; search needs 2/3 the time taken for preprocessing.
Model this as a constraint satisfaction problem: *You need to develop a precedence order for executing the tasks such that each batch is processed completely before the next batch starts.*

a) Identify the set of variables in this problem.
b) Specify the constraints in an appropriate mathematical form.

Hint: you need to first decide on the batch size.

Available memory for the task: 15 GB. Data processing needs 1.5x the data; hence, 1.5x = 15, batch size = 10 GB. Number of batches = 5.
Data processing time: 10 gb * 6 min per GB = 60 min; Search 60*2/3 = 40 min
If you have treated the problem such that 2.5x = 15 also you have been given full marks where appropriate.

Variables: Once you have the batch size, you can capture the variables in a few different formats. Some variations are:

- Batches $b_1 \ldots b_5$; Execution E; Within each batch $b_{i \in \{1,..,5\}}$ preprocessing and search tree generation can be different variables.
- Preprocessing $P_1 \ldots P_5$; Search $S_1 \ldots S_5$; Execution E (this format doesn't capture batches explicitly, but the essence is there)

If you have captured the essence that data has to be processed in batches and that there are 3 steps in the pipeline, you have been given full marks.
If your variables only talk about batch size, system memory etc., you get part points based on how relevant is the information to obtain a schedule (a precedence order of executing the tasks).

Constraints:
Again there are quite a few ways to capture the constraints.
If your solution doesn't reflect a schedule, you get minimal points (<= 2 points) based on how much information relating to the schedule is captured.
If you missed capturing step 3 of the pipeline, you lose 1 point.

If you modeled the solution such that all preprocessing happens before the search tree generation, you lose 1 point

If you have only mentioned batches but not indicated that within each batch preprocessing preceeds search, you lose 1 point.

Based on the variables above, you can have the following solutions

- $b_1 = P_1 + 60 + S_1 + 40 ; \cdots ; b_5 = P_5 + 60 + S_5 + 40$ [Each batch comprises processing (60 min) and search tree generation (40 min)].
- $b_1 < b_2 < b_3 < b_4 < b_5 < E$ [Capturing the precedence order of execution]

- $P_1 + 60 \leq S_1; \cdots ; P_5 + 60 \leq S_5$ [Preprocessing takes 60 min and search tree generation happens after that]
- $S_1 + 40 \leq P_2; \cdots ; S_4 + 40 \leq P_5$ [Each search tree generation needs 40 min, next preprocessing starts after the search tree generation of the previous batch]
- $S_5 + 40 \leq E$ [Execution can start only after the last search tree is generated, which takes 40 min]