Practice Midterm Examination

# CSCI 561 Fall 2016 : Artificial Intelligence

Student ID: | | | | | | | | | | |

Last Name: _____

First Name: _____

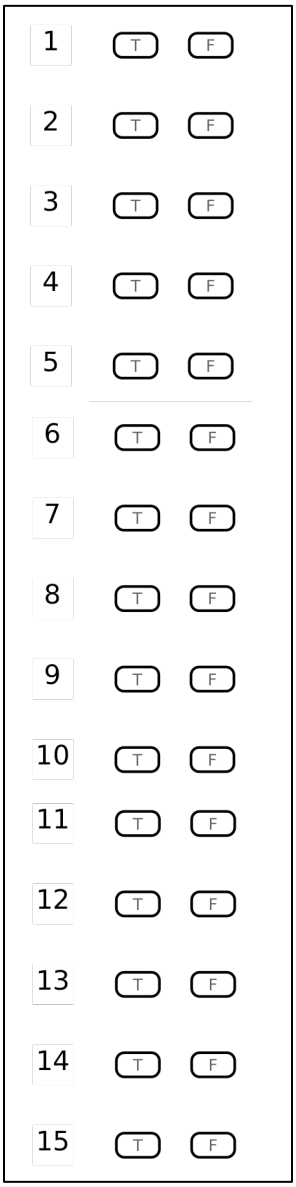
USC email: | | | | | | | | @usc.edu

## Instructions:

1. Date: **9/30/2016 3-5pm at THH**
2. Maximum credits/points for this midterm: 100 points.
3. Credits/points for each question is indicated in the brackets [ ] before the question.
4. **No books or calculators** (or any other material) are allowed.
5. **Write down name, student ID and USC email address.**
6. **Your exam will be scanned and uploaded online.**
7. **Write within the boxes provided for your answers.**
8. **Do NOT write on the 2D barcode.**
9. **The back of the pages will not be graded. You may use it for scratch paper.**
10. No questions during the exam. **If something is unclear to you, write that in your exam.**
11. **Be brief: a few words are often enough if they are precise and use the correct vocabulary studied in class.**
12. **Adhere to the Academic Integrity code.**

# 1. General AI knowledge

For each of the statements below, fill in the bubble **T** if the statement is always and unconditionally true, or fill in the bubble **F** if it is always false, sometimes false, or just does not make sense:




1. A reflex agent that senses only partial information about the state cannot be rational.
2. There exists a deterministic task environment in which any agent is rational.
3. No agent is rational in an unobservable environment.

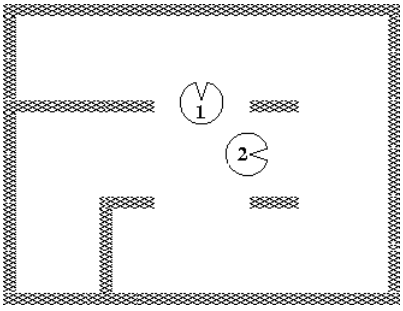
4. An agent function that specifies the agent print true when the percept is a Turing machine program that halts, and false otherwise, cannot be implemented.
5. Local beam search is an entirely deterministic algorithm.

6. Alpha-beta pruning may not find the optimal solution, because it prunes some branches.
7. Local beam search with one initial state and no limit on the number of states retained is Depth First Search.
8. Simulated annealing with $T = \infty$ at all times is a random-walk search

9. Mutation and crossover are methods to handle local extrema in Genetic Algorithms.
10. Even if we use a beam width of 1 and the same tie-breaking procedure, beam search and hill climbing are not complete.
11. If we have an admissible heuristic function, then when a node is expanded and placed in CLOSED, we have always reached it via the shortest possible path.
12. If greedy search finds some goal node in finite time, then depth-first search will also find a goal node in finite time.
13. If depth-first search finds some goal node in finite time, then beam search will also find a goal node in finite time.
14. The primary purpose of the CLOSED list in search is to produce the path from the goal node found back to the initial state.
15. Genetic algorithm with population size $N = 1$ is the same as local beam search with $k=1$.

# 2. Search Algorithms Concepts

Pacman (1) and Ms. Pacman (2) are lost (each at random initial state) in an NxN maze and would like to meet; they don't care where. In each time step, both simultaneously move in one of the following directions: {NORTH, SOUTH, EAST, WEST, STOP}. They do not alternate turns. You must devise a plan that positions them together, somewhere, in as few time steps as possible. Passing each other does not count as meeting; they must occupy the same square at the same time.




(a) [4pts] Formally state this problem as a **single-agent** state-space search problem.

States:

Maximum size of state space:

Maximum branching factor:

Goal test:

(b) [2pts] Give a non-trivial admissible heuristic for this problem.

(c) [2 pts] Circle all of the following graph search methods which are guaranteed to output optimal solutions to this problem:

(i) DFS

(ii) BFS

(ii) UCS

(iv) A* (with a consistent and admissible heuristic)

(v) A* (with heuristic that returns zero for each state)

(vi) Greedy search (with a consistent and admissible heuristic)

(d) [2 pts] If h1 and h2 are admissible, which of the following are also guaranteed to be admissible? Circle all that apply:

(i) $h1 + h2$

(ii) $h1 * h2$

(iii) $\max(h1, h2)$

(iv) $\min(h1, h2)$

(v) $(\alpha)h1 + (1-\alpha)h2$, for $\alpha \in [0,1]$

# 3. Comparing Search Strategies

Consider the search space on the following page, where S is the start node and G1 and G2 satisfy the goal test. Arcs are labeled with the cost of traversing them and the estimated cost to a goal is reported inside nodes.

For each of the following search strategies, indicate which goal state is reached (if any) and list, in order, all the states of the nodes **popped off** of the OPEN queue. When all else is equal, nodes should be removed from OPEN in alphabetical order.

You should not expand nodes with states that have already been visited. Note how the arcs in the figure are oriented, which means that you can only go from one state to another if the arrow points from the first to the second. For example, you can go from S to C (i.e., C is a successor of S) but not from C to S (i.e., S is not a successor of C).

(a) [6 pts] Greedy search

States popped off OPEN: | Goal state reached:

(b) [8 pts] Uniform Cost search

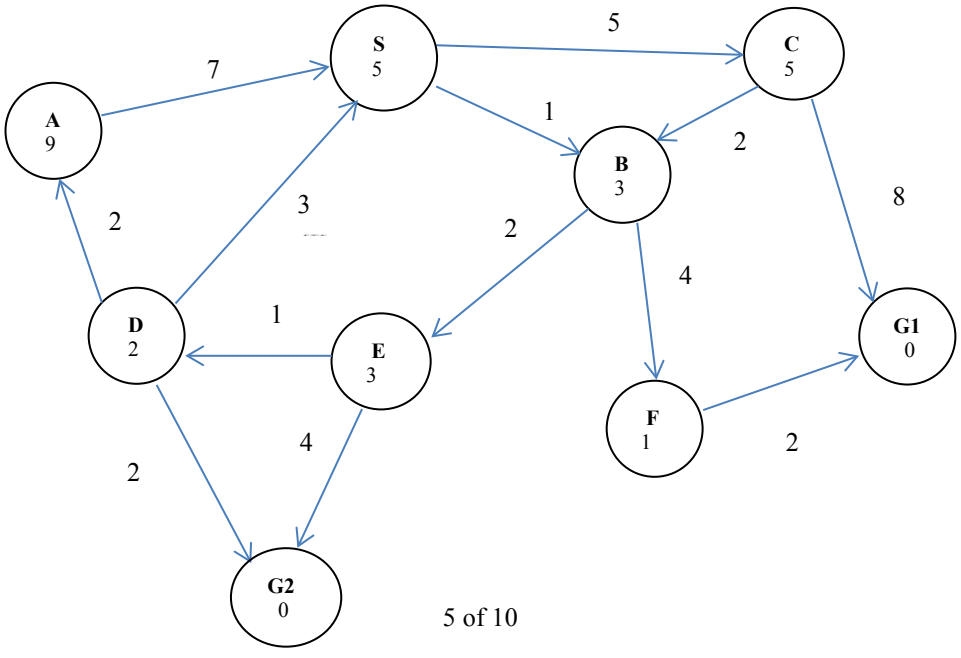
States popped off OPEN: | Goal state reached:

(c) [8 pts] Iterative Deepening Search

States popped off OPEN: | Goal state reached:

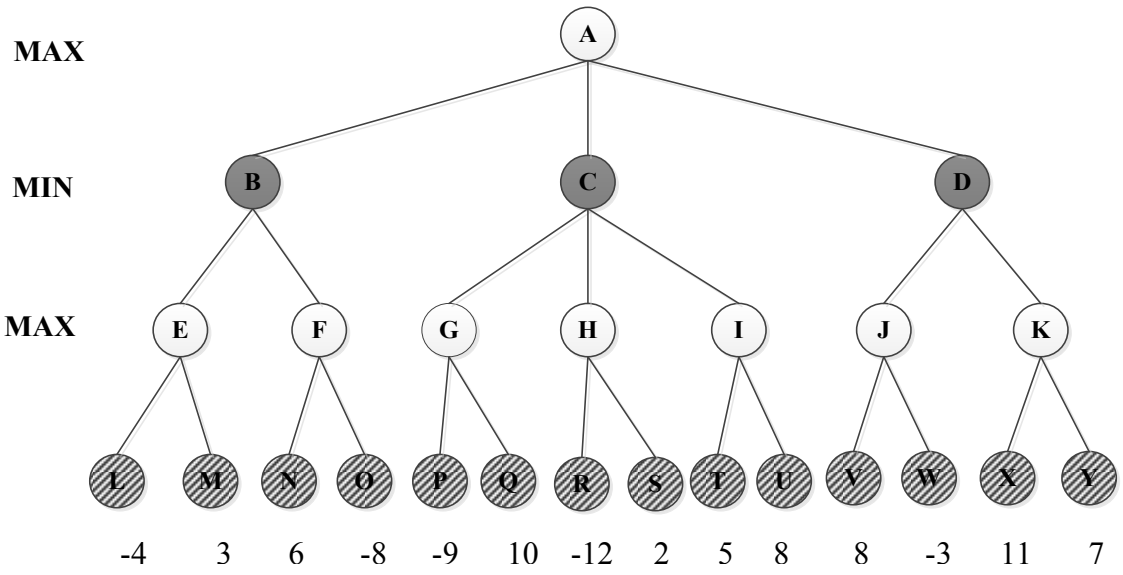
(d) [8 pts] A* Search

States popped off OPEN: | Goal state reached:



# 4. Game Playing

Consider the following game tree in which the evaluation function values are shown below each leaf node. Assume that the root node corresponds to the maximizing player. Assume that the search always visits children left-to-right.




(a) [4 pts] Compute the backed-up values computed by the minimax algorithm. Show your answer by writing values next to the appropriate nodes in the above tree.

(b) [6 pts] Which nodes will not be examined by the alpha-beta pruning algorithm? List the letter of the prune nodes.

(c) [5 pts] Can the nodes B, C, D be expanded in a different order to improve the number of nodes pruned? If yes, list the new order of nodes B, C, D and the letter of the prune nodes, or if no, explain why not.

# 5. Constraint Satisfaction

You are an IT manager in charge of customer support. You have **five important customers** that need maintenance once each week during specific time slot.

Your job is to assign your three engineers to these companies on one day for regular maintenance. The engineers have to stay there until the task is done. The engineers have different skills, so not all of them can maintain all companies. Also, each engineer can only serve one company in the assigned period.

The schedules of the customers are:
- Company 1: Webflix: 8:00-9:00am
- Company 2: Anazon: 8:30-9:30am
- Company 3: Pied Piper: 9:00-10:00am
- Company 4: Hooli: 9:00-10:00am
- Company 5: Gulu: 9:30-10:30am

The profiles of your engineers are:
- Albacore can maintain Pied Piper and Hooli.
- Bosam can maintain all companies, except Webflix.
- Coleslaw can maintain all companies.

a) [3%] Using Company as variable, formulate this problem as a CSP problem with variables, domains, and constraints. Constraints should be specified formally and precisely, but may be implicit rather than explicit.

b) [3%] Draw the constraint graph associated with your CSP.

c) [10%] Show the domains of the variables after running arc-consistency on this initial graph (after having already enforced any unary constraints).

d) [4%] Give one solution to this CSP

# 6. Local Search

Sudoku problem, requires that the same single integer may not appear twice in the same row, column or in any of the nine 3×3 subregions of the 9x9 playing board. To simplify our problem, we do not consider the subregions, which means you can have the same integers in any of the nine 3×3 subregions. Each state is represented by a 9*9 2D-array of numbers. The only action we can do is to swap two numbers in the same column.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 5 | 3 | 4 | 6 | 7 | 8 | 9 | 1 | 2 |
| 6 | 7 | 2 | 1 | 9 | 5 | 3 | 4 | 8 |
| 1 | 9 | 8 | 3 | 4 | 2 | 5 | 6 | 7 |
| 8 | 5 | 9 | 7 | 6 | 1 | 4 | 2 | 3 |
| 4 | 2 | 6 | 8 | 5 | 3 | 7 | 9 | 1 |
| 7 | 1 | 3 | 9 | 2 | 4 | 8 | 5 | 6 |
| 9 | 6 | 1 | 5 | 3 | 7 | 2 | 8 | 4 |
| 2 | 8 | 7 | 4 | 1 | 9 | 6 | 3 | 5 |
| 3 | 4 | 5 | 2 | 8 | 6 | 1 | 7 | 9 |

(a) [2pts] How many neighbor states for this sample state?

(b) [2pts] What is the total size of the search space, i.e. how many possible states are there in total?

(c) [2pts] Suppose all black numbers are fixed. Only gray numbers can be swapped. How many neighbor states for this sample state? how many possible states are there in total?

Look at the state below. It is a simplified Sudoku problem that requires the same single integer may not appear twice in the same row or column. You decide to use a hill-climbing algorithm to solve this problem.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 7 | 1 | 8 | 6 | 5 | 1 | 1 |
| 2 | 3 | 4 | 2 | 1 | 7 | 3 | 6 | 2 |
| 6 | 4 | 2 | 3 | 3 | 5 | 6 | 9 | 3 |
| 3 | 2 | 3 | 4 | 9 | 4 | 7 | 2 | 7 |
| 4 | 6 | 9 | 9 | 5 | 1 | 4 | 3 | 4 |
| 8 | 8 | 5 | 6 | 7 | 2 | 8 | 5 | 5 |
| 5 | 5 | 6 | 7 | 6 | 8 | 9 | 4 | 9 |
| 7 | 7 | 8 | 8 | 2 | 3 | 1 | 7 | 6 |
| 9 | 9 | 1 | 5 | 4 | 9 | 2 | 8 | 8 |

(d) [4pts] Design a heuristic function for your hill-climbing algorithm. (Your h function for goal state should be 0.)

(e) [5pts] Base on your designed heuristic function, write an example of a valid action you may choose from this state.