**Northeastern University – Silicon Valley**

CS 5100 Foundations of AI

**Midterm Exam 8/1/20** [100 points]

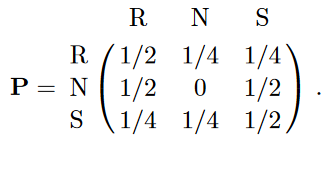
This is an open book exam. Pen and paper preferred. Scan answers to PDF. Duration: 150 min.

Answer all questions, with brief but complete explanations. K*eep your answers concise and to the point. Please write legibly. Write your name on Page 1. Hint: Try the larger credit questions first. Answer on separate papers.*

1. **State Space Representation** [8 points]
2. What is State space representation of simple AI problems? What are the key components of such representation?
3. Provide a diagram which shows the search space for the Missionaries and Cannibals problem.
4. **Agents Worlds** [12 points]

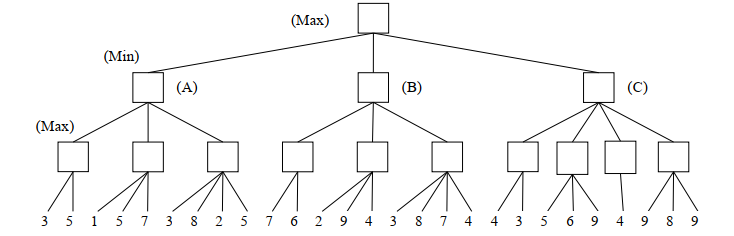
LILBANK is a bank with four rooms, every adjacent room is connected with the door. All rooms also have an external door.

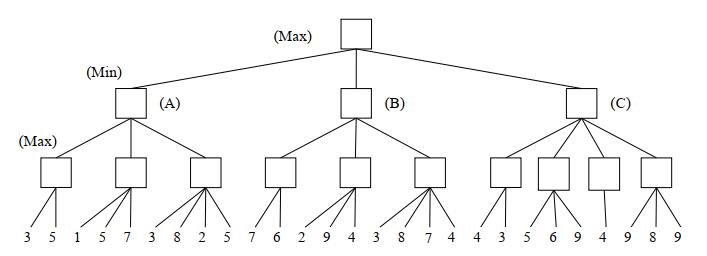
There is a security agent (Guard G) in the bottom leftmost corner. There is a thief T somewhere in the bank. Guard’s job is to move from one room to the next, check if the external door is locked are not, and relay a message for the police as follows: (OK if the door is locked and there is no thief; NAK if it orders phone unlocked and HELP if there is a thief found in the same room as the agent).

1. Draw a diagram to fully represent the above as an agent based AI world problem.
2. Provide an agent program (pseudocode) for a simple reflex agent first. Then modify this program to represent a model based agent.
3. What is AND-OR search? Give an example where and AND-OR Search tree function would be useful in solving the Missionaries and Cannibals problem from Question 1 above. Show with a figure. [7 points]
4. In an island, they never have two nice days in a row. If they have a nice day, they are just as likely to have snow as rain the next day. If they have snow or rain, they have an even chance of having the same the next day. If there is change from snow or rain, only half of the time is this a change to a nice day. With this information we form a Markov chain as follows. We take as states the kinds of weather R, N, and S. From the above information we determine the transition probabilities in a square matrix below: [8 points]
5. **Draw** a Transition Diagram for this MDP.
6. Explain if you need an HMM to make predictions about this weather. Why or why not?
7. MINI-MAX SEARCH IN GAME TREES:

The game tree below illustrates a position reached in the game. Process the tree left-to-right. It is Max's turn to move. At each leaf node is the estimated score returned by the heuristic static evaluator. [25 points]

1. Fill in each blank square with the proper mini-max search value.
2. What is the best move for Max? (write A, B, or C)
3. What score does Max expect to achieve?



1. **ALPHA-BETA PRUNING:** Process the tree left-toright. This is the same tree as above. You do not need to indicate the branch node values again. [25 points]  
   
2. For each English sentence below, write the letter corresponding to its best or closest FOPC (FOL) sentence (wff, or well-formed formula). The first one is done for you, as an example. [15 points]

**“Every butterfly likes some flower.”**A. ∀x ∀y Butterfly(x) ∧ Flower(y) ∧ Likes(x, y)  
B. ∀x ∃y Butterfly(x) ∧ Flower(y) ∧ Likes(x, y)  
C. ∀x ∀y Butterfly(x) ⇒ ( Flower(y) ∧ Likes(x, y) )  
D. ∀x ∃y Butterfly(x) ⇒ ( Flower(y) ∧ Likes(x, y) )

**“All butterflies are insects.”**  
A. ∀x Butterfly(x) ∧ Insect(x)  
B. ∀x Butterfly(x) ⇒ Insect(x)  
C. ∃x Butterfly(x) ∧ Insect(x)  
D. ∃x Butterfly(x) ⇒ Insect(x)

**“For every flower, there is a butterfly that likes that flower.”**  
A. ∀x ∃y Flower(x) ∧ Butterfly(y) ∧ Likes(y, x)  
B. ∀x ∃y [ Flower(x) ∧ Butterfly(y) ] ⇒ Likes(y, x)  
C. ∀x ∃y Flower(x) ⇒ [ Butterfly(y) ∧ Likes(y, x) ]  
D. ∀x ∀y Flower(x) ∧ Butterfly(y) ∧ Likes(y, x)

**“Every butterfly likes every flower.”**  
A. ∀x ∀y [ Butterfly(x) ∧ Flower(y) ] ⇒ Likes(x, y)  
B. ∀x ∀y Butterfly(x) ⇒ [ Flower(y) ∧ Likes(x, y) ]  
C. ∀x ∀y Butterfly(x) ∧ Flower(y) ∧ Likes(x, y)  
D. ∀x ∃y [ Butterfly(x) ∧ Flower(y) ] ⇒ Likes(x, y)

**“There is some butterfly in Irvine that is pretty.”**  
A. ∀x Butterfly(x) ∧ In(x, Irvine) ∧ Pretty(x)  
B. ∃x Butterfly(x) ∧ In(x, Irvine) ∧ Pretty(x)  
C. ∀x [ Butterfly(x) ∧ In(x, Irvine) ] ⇒ Pretty(x)  
D. ∃x Butterfly(x) ⇒ [ In(x, Irvine) ∧ Pretty(x)]