

CSE 471/598 Intro to AI (Kambhampati) Fall 99 Midterm Total Marks: 75
Time: 75min (3:15-4:30)

Name: _____ **Student ID:** _____

Instructions: Do not open the exam until told to do so. If you make 70 points, it is considered maximum. Thus there is a 5 point "choice" on the exam.

*This exam is **open book and open notes**. Answer all the questions.*

If you feel you could not do well in the exam due to time and other pressures, you are welcome to do take a copy of the exam from the homepage work on it at home "BY YOURSELF" and submit it on Monday at the beginning of the class. It will be graded as an optional homework by the TA (you of course can use notes and textbook for the at-home version).

<p><i>An AI Limerick: (Kautz)</i></p> <p>If your thesis is utter vacuous Use first-order predicate calculus. With sufficient formality The sheerest banality Will be hailed by the critics: "Miraculous!"</p>	<p><i>An AI KOAN: (MIT AI Lab)</i></p> <p>In the days when Sussman was a novice, Minsky once came to him as he sat hacking at the PDP-6. "What are you doing?", asked Minsky. "I am training a randomly wired neural net to play Tic-Tac-Toe" Sussman replied. "Why is the net wired randomly?", asked Minsky. "I do not want it to have any preconceptions of how to play", Sussman said. Minsky then shut his eyes. "Why do you close your eyes?", Sussman asked his teacher. "So that the room will be empty." At that moment, Sussman was enlightened.</p>
---	--

<p>An AI Lightbulb joke (Rich & Knight)</p> <p>Question: How many AI people does it take to change a lightbulb? Answer: At least 67.</p> <p>(Partial) Explanation:</p> <p>5 - The Problem Space Group</p> <ul style="list-style-type: none"> One to define the goal state One to define the operators One to describe the universal problem solver One to hack the production system One to indicate about how it is a model of human lightbulb-changing behavior <p>7 - The Lisp Hackers</p> <ul style="list-style-type: none"> One to bring up the network One to order the Chinese food Four to hack on the Lisp debugger, compiler, window system, and microcode One to write the lightbulb-changing program 	<p>5 - The Game-Playing Group</p> <ul style="list-style-type: none"> One to design a two-player game tree with the robot as one player and the lightbulb as the other One to write a minimax search algorithm that assumes optimal play on the part of the lightbulb One to build special-purpose hardware to enable 24-ply search One to enter the robot in a human lightbulb-changing tournament One to state categorically that lightbulb changing is "no longer considered AI" <p>12 - The Logical Formalism Group</p> <ul style="list-style-type: none"> One to figure out how to describe lightbulb changing in predicate logic One to show the adequacy of predicate logic One to show the inadequacy of predicate logic One to show that lightbulb logic is nonmonotonic One to incorporate nonmonotonicity into predicate logic One to determine the bindings for the variables One to show the completeness of the solution One to show the consistency of the solution One to hack a theorem prover for lightbulb resolution One to indicate how it is a description of human lightbulb-changing behavior One to call the electrician
--	--

Qn I.[11x2=22] For each of the following statements below, indicate whether the statement is true or false, and give a brief but precise justification for your answer. Correct answers with correct justifications will carry 2points. **No points will be awarded for answers without correct justifications.**

Example qn: The time and memory requirements of IDA* can be improved by using A* algorithm to do search in individual iterations.

Answer: False. Because A* in the worst case can take as much memory as breadth-first, and thus using A* in the individual iterations will make IDA* require exponential memory (instead of linear memory).

- A. We can derive the empty clause by resolving the clauses $[\sim A \vee \sim B]$ and $[A \vee B]$ together

- B. A player using the full min-max search strategy will always be able to win games in smallest number of moves

- C. If we have a program that can solves CSP problems, we can use that program to check if certain fact F follows form a propositional knowledgebase K.

- D. The reason Herbrand interpretations provide a better foundation than Tarskian interpretations for computationally attractive semantics for first order logic is that there are only a finite number of Herbrand interpretations compared to an infinite number of Tarskian ones.

- E. Let $h_1(n)$ be an admissible heuristic for some search problem P. Let $R(n)$ be a function that returns a small random number between 0 and 1. Let $h_2(n) = h_1(n) + R(n)$. Suppose further than $h_2(n)$ is still admissible. An IDA* search using $h_2(n)$ in place of $h_1(n)$ would do more efficient search.

- F. Iterative Deepening Depth First search is a very appropriate search strategy for solving CSP problems, since it guarantees optimal solution with linear amount of memory.
- G. Quiscent search in game-tree evaluation involves checking if the evaluation value of the leaf nodes is indeed stable. This is done by expanding the leaf nodes of the game-tree a few additional ply and seeing if the backed-up evaluation values differ considerably from the values returned by the evaluation function.
- H. There is basically no difference between A* search with $f(n)=h(n)$ (i.e., $g(n)$ is ignored) and hill-climbing search—both, in essence try to follow the currently most promising node, disregarding the cost of getting to that node in the first place.
- I. The role of Alpha-beta pruning in Min-max is more similar to the role of the branch and bound technique in A* than the role of admissible heuristic in A* .
- J. As what we have done in the course till now clearly demonstrates, the goal of AI is to think the way humans think.
- K. The performance of CSP search strategies can be improved by doing a limited amount of duplicate-node detection.

Question II.[15] Consider the following simple grid-search problem, that may be faced by a mobile robot navigating a room with obstacles. The cells in the grid are named from A to F. If a cell is shaded, that means there is an obstacle in that cell—and so the robot cannot get into that cell. The only moves available to the robot are to go right, left, up or down (as permitted by the gridcell topology). Each move is considered to have a unit cost. When making a move takes the robot into a cell with obstacles, that move is considered infeasible. Suppose the robot is in the cell named A, and wants to get to the cell named F.

G	H	I
D	E	F **
A	B	C

Consider the two heuristics for this problem—the straight-line distance heuristic, and the manhattan distance heuristic. The straightline distance heuristic just counts up cells falling on the straightline from the current cell to the goal cell. The manhattan distance heuristic is as you used it in project 2. For example, the straightline distance between A and I is 2, while the manhattan distance is 4.

Part A.[2] Which of these heuristics is admissible for this problem? Consider in particular, whether the presence of obstacles changes your answer.

Part B.[2] Which heuristic would you rather use, if you are planning to use A* search on this problem.

Part C.[2] Would your answer to Part A change if the robot were allowed diagonal moves (i.e, to go from A to E, for example).

Part D.[6] Show how A* search with the heuristic you chose in part B will solve this problem (assuming only the up,down,left and right moves) and give you a path from A to F. You need to give enough details along with the search tree to convince me you understand A* .

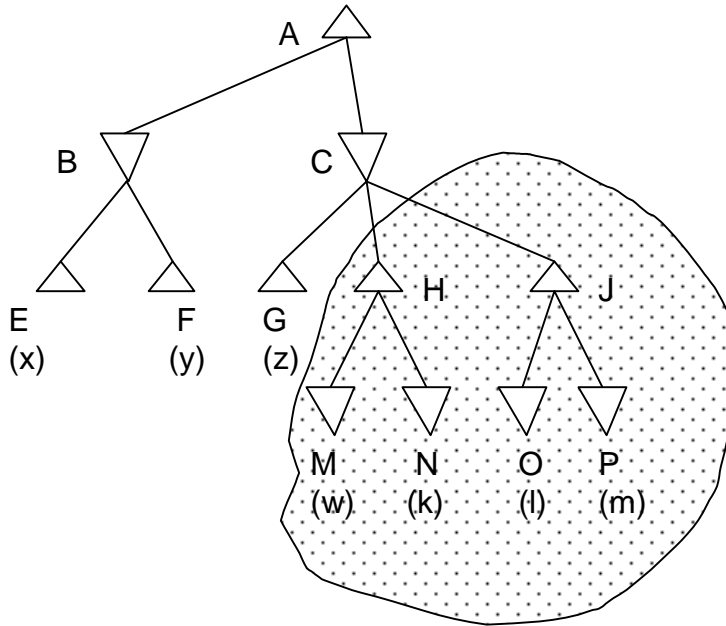
Part E.[3] On the search tree that you put up for A* in the previous question, draw the contours corresponding to the various iterations that IDA* search would take to solve the same problem. (Ask me if you have difficulty understanding this).

Qn III.

Part A. [8] Here is an extremely silly two-person game—it is even sillier than tic-tac-toe (if such a thing is possible). In this game, there are some number of stacks of coins on the table (each stack contains possibly different number of coins). When it is a player's turn, he/she needs to choose a stack of coins and split it into *two* unequal sized stacks (e.g., if you pick a stack of 7 coins, you can split it into two stacks of 5 and 2 or two stacks of 3 and 4 or two stacks of 6 and 1). When the game reaches a stage such that the player whose turn it is now cannot pick a stack from the table and split it this way, the player is considered to have **lost** the game (and the other wins).

Suppose **that initially there is a stack of 5 coins** on the table. You (MAX) is playing first. **PROVE** using min-max strategy, MAX will ALWAYS win this game, whoever the opponent (MIN) is. (You will have to explain what is the representation that you are using for the game-tree state, and generate the full min-max tree which is possible since it is such a small game).

Part B.[4] Consider the following game-tree, where x, y, \dots, m are the evaluation values of the leaf node.

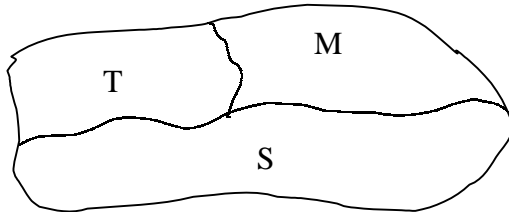


Part B.1. Assuming a left-to-right evaluation of the tree, give the most general relation between x, y, z, w, k, l and m that must hold such that the tree in the dotted part would be pruned (would never be considered) by alpha-beta pruning.

Part B.2. Now, give the most general relation that must hold so that node F will be pruned by Alpha-beta pruning (assuming left-to-right evaluation again). Interpret your result.

Question IV[11]

Consider the following (fictional) problem of coloring a small regional map with three cities—Tempe, Mesa and Scottsdale—which all share borders. We are trying to put colors on these cities. Clearly, the coloring should be such that no two adjacent cities have the same color. You have three colors with which to color these cities—*yellow, red and green*. Tempe due to its sundevil craziness wants only the color yellow. Mesa folks hate *green* (may be some latent anti-irish sentiment). Scottsdale folks are too laid back to have any strong preferences (they don't mind any of the colors).



- A. [3] Model this problem as a CSP problem. What are the variables? What are their domains? What are the constraints?
- B. [5] Show how this problem would be solved using a systematic CSP search strategy that uses *forward checking* and *dynamic variable ordering*. You need to show the search tree—explain why you pick each variable, show how forward-checking occurs, and show how a solution is found.
- C. [3] Suppose you are using a min-conflict based (hill-climbing) search instead of systematic search. Your current assignment puts yellow on tempe, yellow on mesa and red on scottsdale. Show how you would change the assignment in the next iteration using the min-conflict idea. You need to show which is the city whose color would be changed, and what would be the changed color. Show how you arrived at that answer.

QUESTION V

A.[3] Here is a piece of “Python logic” (from Monty python and the holy grail):

Things made of wood float on water.

Mary floats on water

Ergo, the Mary is made of wood.

Prove that this is logically unsound reasoning.

B. Consider the following piece of knowledge:

Everyone is loved by someone.

Anyone who is loved by someone is happy and chirpy.

We want to prove that *everyone is happy.*

B.1. [3] Show the knowledge and the goal in first order logic form

B.2. [3+4] Use *resolution refutation* to prove the goal. Note that you need to show the clausal form of the sentences, and the resolution trace that shows the proof.

B.3. [2] Could this proof have been done using Prolog? Explain your answer.