CSCI 4150: Intro. to Artificial Intelligence
Midterm Examination

Tuesday October 17, 2000

Name: ____

1 Assorted search questions

The following questions explore differences in the number of nodes evaluated for different search implementations.

1. (points) Two students each write a correct implementation of breadth first search. They then each write correct implementations of the `goal?` and `get-children` functions to solve a problem. However, after running their programs on the same start state, they find that their programs have evaluated different numbers of nodes.

   How can they both have correct implementations yet evaluate different numbers of nodes? Be specific in your answers.

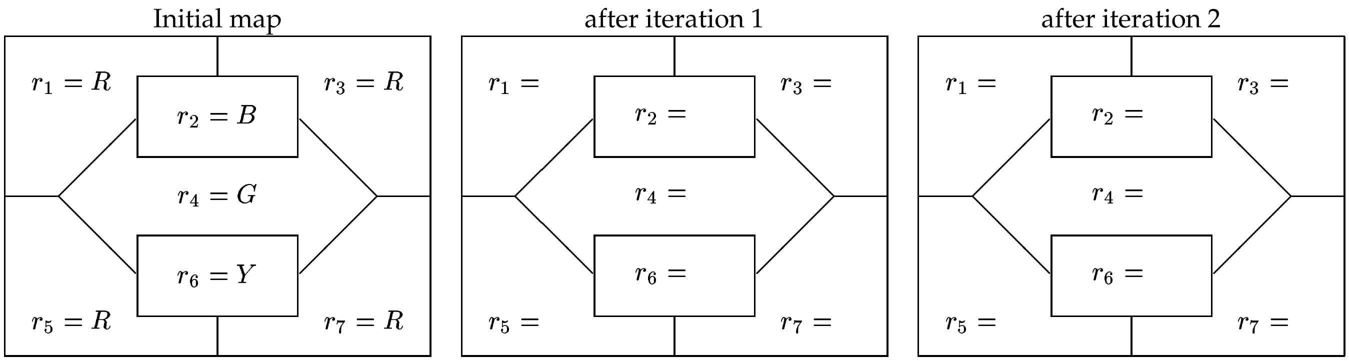
2. (points) Two students each write a correct implementation of the A* search using the OPEN/CLOSED list formulation. They then run the A* search on the same problem using the same heuristic. Is it possible for them to evaluate different numbers of nodes? Explain why or why not.

3. (points) Assume we have a search tree with a uniform branching factor b. What is the minimum and maximum number of nodes that breadth first search will evaluate to find a solution at depth d? What is the minimum and maximum number of nodes that depth first search will evaluate? Express your answers in terms of the depth d of the solution and the maximum depth m of the tree.

2 Constraint Satisfaction and Iterative Improvement search

(points) The map coloring problem is a good example of a constraint satisfaction problem. The goal is to assign one of four colors to each region of the map so that no two adjacent regions have the same color. The initial assignment of the colors to regions is given below. The four colors (abbreviated on the map below) are Red, Green, Blue, and Yellow. The color of each region is assigned to the variable r_i.

For problem, you are to do two iterations of heuristic repair using the "min-conflicts" heuristic. You can choose variables randomly, and you should also break any ties randomly. Show the color assignments after each iteration and explain your work.



3 Informed search

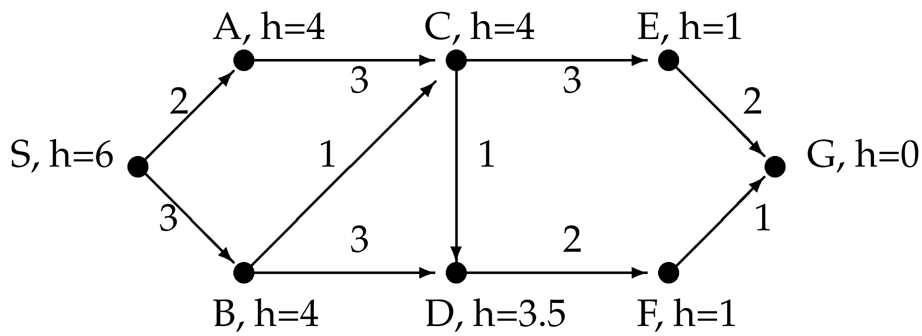
1. (points) The following questions explore heuristics and informed search.

   (a) What is an admissible heuristic?

   (b) Is A* search with an admissible heuristic complete? Is it optimal?

   (c) Is A* search with an inadmissible heuristic complete? Is it optimal?

2. (points) For this problem you will apply the A* algorithm to the graph below. Note that the edges in this graph are directed.



For this problem:

- Perform the A* algorithm on this graph, filling in the table below. You should not need all the lines in the table. Indicate the f, g, and h values of each node on the queue as shown in the first two rows of the table.

  Assume that if you find a path to a node already on the queue that you update its cost (using the lower f value) instead of adding another copy of that node to the queue.
- Show the path found by the A* algorithm on the graph above.

iteration	node expanded	new queue/OPEN list
0		S = 0 + 6 = 6 (i.e. S = g(S) + h(S) = f(S))
1	S	A = 2 + 4 = 6; B = 3 + 4 = 7
2		
3		
4		
5		
6		
7		
8		
9		
10		

4 Game search

1. (points)

   (a) Explain the horizon problem for game tree search.

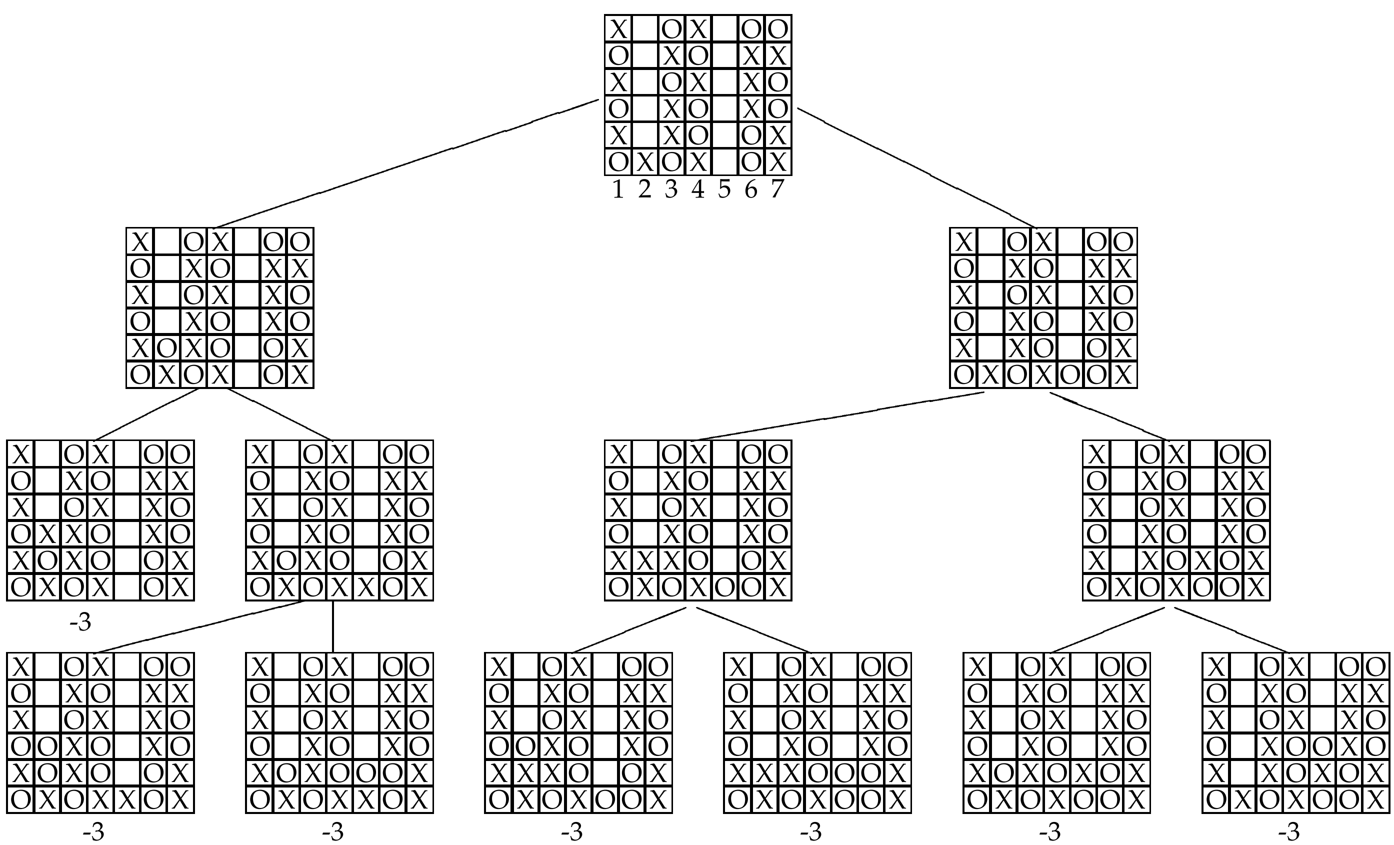
   (b) What is the space and time complexity of minimax search?

   (c) What is the space and time complexity of minimax search with alpha-beta pruning?

2. (points) Suppose we are using minimax search (without alpha-beta pruning) with a depth cutoff of 2 to play a game of Connect 4.

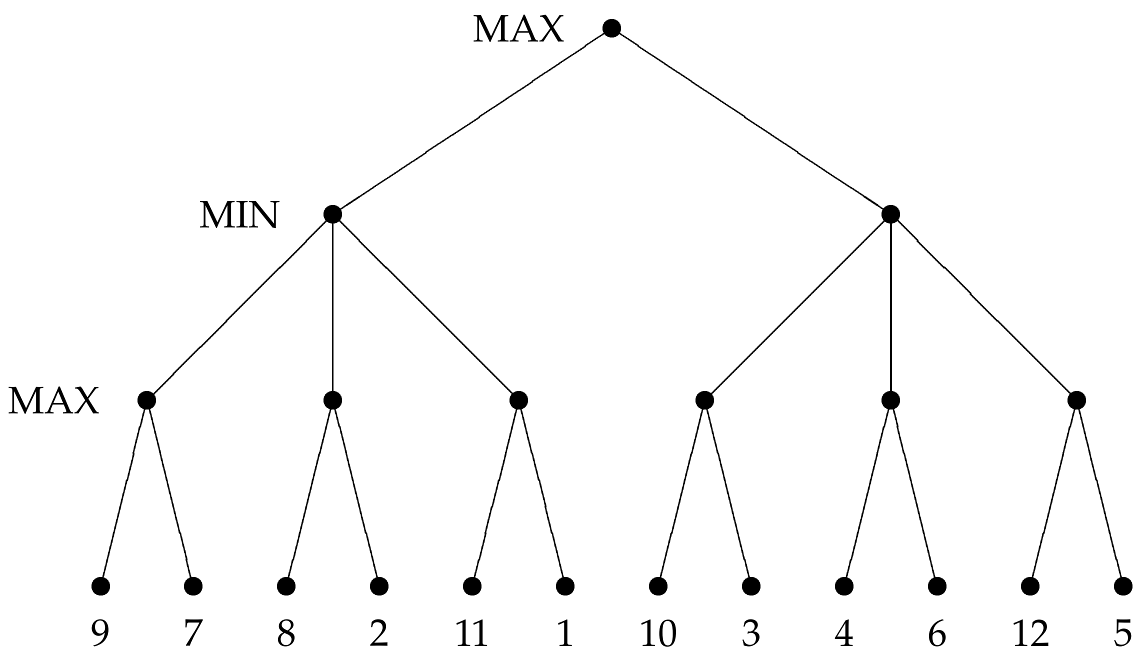
   It is O's turn to move; the resulting game tree appears below. The value of the evaluation function for each leaf node is shown.

   - What value does minimax return on this game tree?
   - Indicate O's best move on the tree below.



3. (points) Perform alpha-beta minimax search on the tree below.

   - Use the cutoff test $\alpha \geq \beta$
   - Indicate which leaf nodes are evaluated by circling the value of the leaf node.
   - What is the value of this tree?
   - Which is the best move from the root node?



5 Logic

Here are some sentences in first order logic that encode knowledge about the "kinship domain:"

1. Father(x, y) → Parent(x, y)
2. Mother(x, y) → Parent(x, y)
3. Wife(a, b) → Married(a, b)
4. Husband(a, b) → Married(a, b)
5. Married(j,k) ∧ Parent(k, m) → Parent(j, m)
6. Sister(s, t) ∧ Parent(t, v) → Aunt(s, v)
7. Brother(s, t) ∧ Parent(t, v) → Uncle(s, v)

Remember the convention for these two argument predicates is as follows: Father(x, y) means "x is the father of y," Wife(a, b) means "a is the wife of b," etc.

1. Write additional sentences *in Horn normal form* for this knowledge base to add definitions for the following new predicates. Use only the predicates introduced above and ones that you will write below. Make your sentences as general as possible!

   (a) Sibling(g, h): g is a sibling of h

   (b) Cousin(g, h): g is a cousin of h

   (c) Grandparent(g, h): g is a grandparent of h

2. For this problem, do *either* Part A *or* Part B. Use the chart on the following page to show each step of forward/backward chaining. You should only show the successful attempts at matching rules.

   (a) Using the original knowledge base (1-7), use forward chaining to add the following sentences:

   - Father(Homer, Lisa)
   - Mother(Marge, Bart)
   - Wife(Marge, Homer)
   - Sister(Patty, Marge)

   As an example of filling out the chart for this problem, here's how you'd start forward chaining on the sentence Husband(Bob, Alice):

sentence number	Term added to KB, or Premise sought	matches sentence #	with substitution	premise(s) to match conclusion produced
8	Husband(Bob, Alice)	4	a/Bob, b/Alice	Married(Bob, Alice)

   (b) Using the original database (1–7) and the following sentences:

   9. Father(Homer, Lisa)
   10. Mother(Marge, Bart)
   11. Wife(Marge, Homer)
   12. Sister(Patty, Marge)

   do backward chaining to show Aunt(Patty, Lisa).

   As an example of filling out the chart for this problem, here's how you might start backward chaining on the sentence Parent(Alice, Bob):

	Premise sought, or Conclusion to match	matches sentence #	with substitution	premise(s) to match
	Parent(Alice, Bob)	5	j/Alice, m/Bob	Married(Alice, k), Parent(k, m)

(FC) sentence number	Term added to KB (FC) Premise sought (FC/BC) Conclusion to match (BC)	matches sentence #	with substitution	premise(s) to match (FC/BC) conclusion produced (FC)