

Practice Problems

Disclaimer


These practice problems are for your practice alone; I make no claims of their representation on the exam, or their relevance to exam material. Any and all exercises listed herein are simply my best guess of the type of problems you'll encounter and what you should be expected to perform on a test. They may or may not represent the difficulty of any question encountered on the exam. Practicing may not be right for you. Consult your doctor before using this practice exam.

Whew... sorry about that, my lawyers always make me cover my bases.

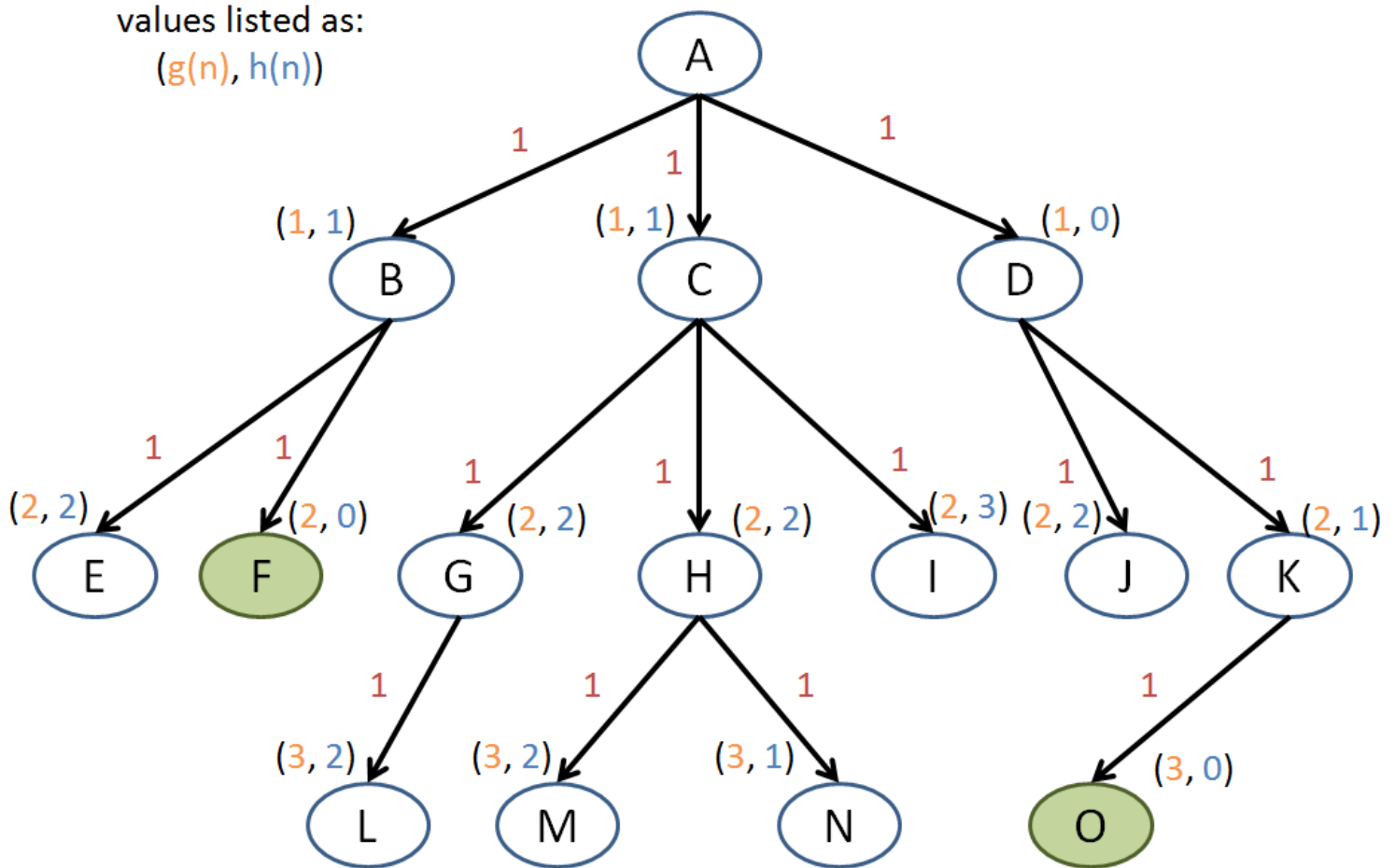
Here's a smattering of problems I think might be on the exam. Have at it, and let me know if you have any questions!

Search Problems

Example

 The following tree represents some imaginary search tree composed of lettered states, costs along edges, and evaluation function values at nodes in the form $(g(n), h(n))$, where $g(n)$ is the cost it took to reach that node, and $h(n)$ is some admissible heuristic's score of that node to some goal state. Goal states (F, O) are listed in green.

Each node has evaluation
values listed as:
 $(g(n), h(n))$



❓ Assuming a left-to-right node expansion and generation order (and not stopping at goal states), in what order will uninformed **Depth First Search** traverse this tree?

❓ If our uninformed DFS does stop at a goal state, which will it find? Is this an optimal goal? Is DFS guaranteed to give us an optimal goal if one exists?

❓ Assuming a left-to-right node expansion and generation order (and not stopping at goal states), in what order will uninformed **Breadth First Search** traverse this tree?

❓ If our uninformed BFS does stop at a goal state, which will it find? Is this an optimal goal? Is BFS guaranteed to give us an optimal goal if one exists?

❓ Assuming nodes of equivalent score generated earlier in the process will be expanded first, and that our search does not stop at goal states, in what order will **Greedy Search** traverse this tree (assuming an admissible heuristic and that lesser-depth nodes are expanded before greater ones in the event of a tie)?

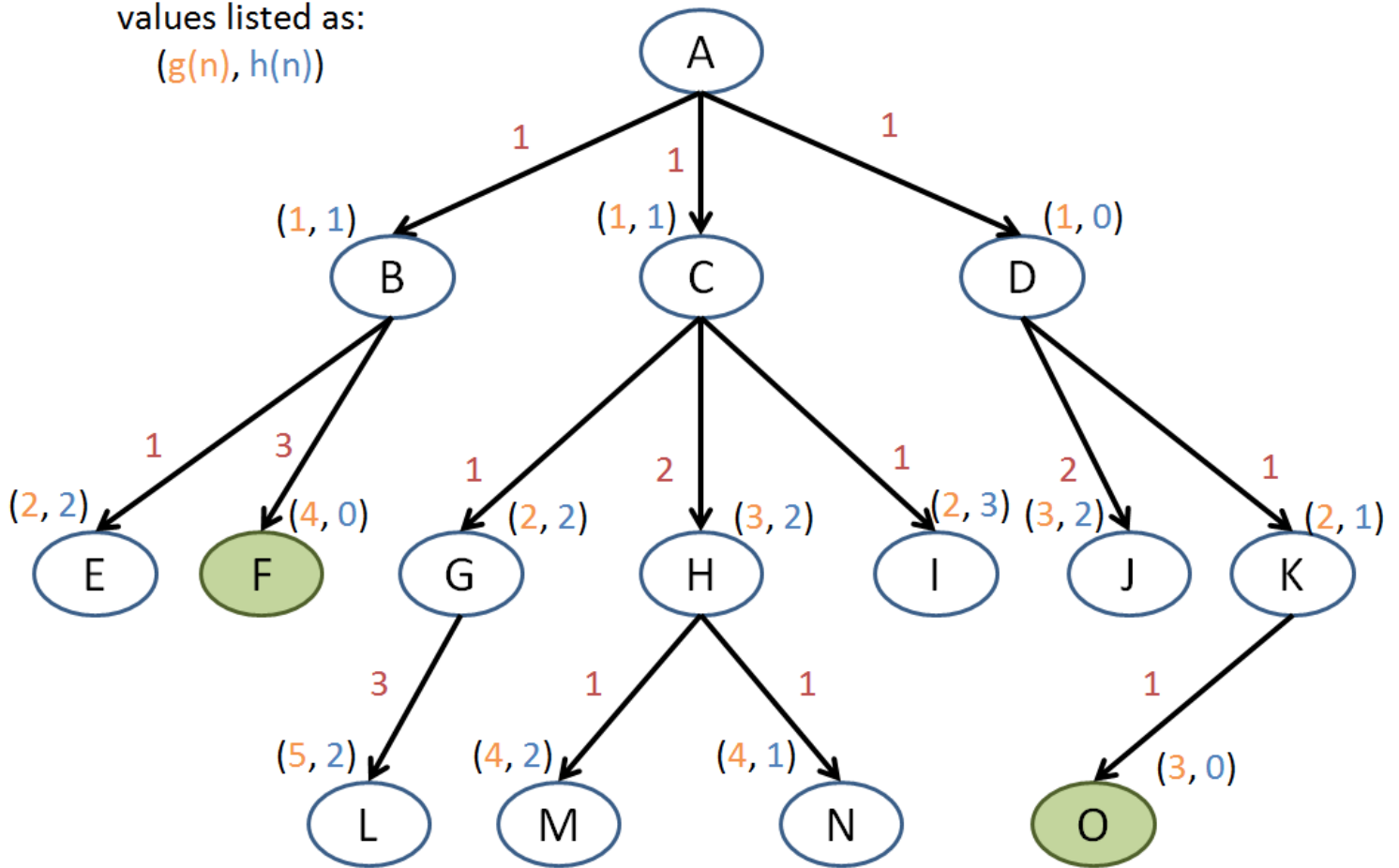
❓ If our greedy search does stop at a goal state, which will it find? Is this an optimal goal? Is greedy search guaranteed to give us an optimal goal if one exists?

❓ Assuming nodes of equivalent score generated earlier in the process will be expanded first, and that our search does not stop at goal states, in what order will **A* Search** traverse this tree (assuming an admissible heuristic)?

❓ If our A* search does stop at a goal state, which will it find? Is this an optimal goal? Is A* search guaranteed to give us an optimal goal if one exists?

Alright, let's switch up our search tree so that it's no longer uniform cost!

Each node has evaluation
values listed as:
 $(g(n), h(n))$



❓ If our greedy search does stop at a goal state, which will it find? Is this an optimal goal?

❓ If our A* search does stop at a goal state, which will it find? Is this an optimal goal?

❗ **Beam search** is a breadth-first-search optimization whereby some subset of nodes along the frontier of each level are expanded according to some heuristic score. The **beam width** defines the number of best (according to the heuristic) nodes along the frontier that will be expanded with each level.

Thus, the steps of beam search for beam width N are:

1. If our frontier is empty, we're done. Otherwise, expand the best N nodes in the current frontier, adding the nodes generated from that expansion to the queue.
2. Order those generated nodes by their heuristic cost.
3. Are any the goal state? If so, we're done, otherwise:
4. Remove all but the first N nodes from the queue and go back to (1).

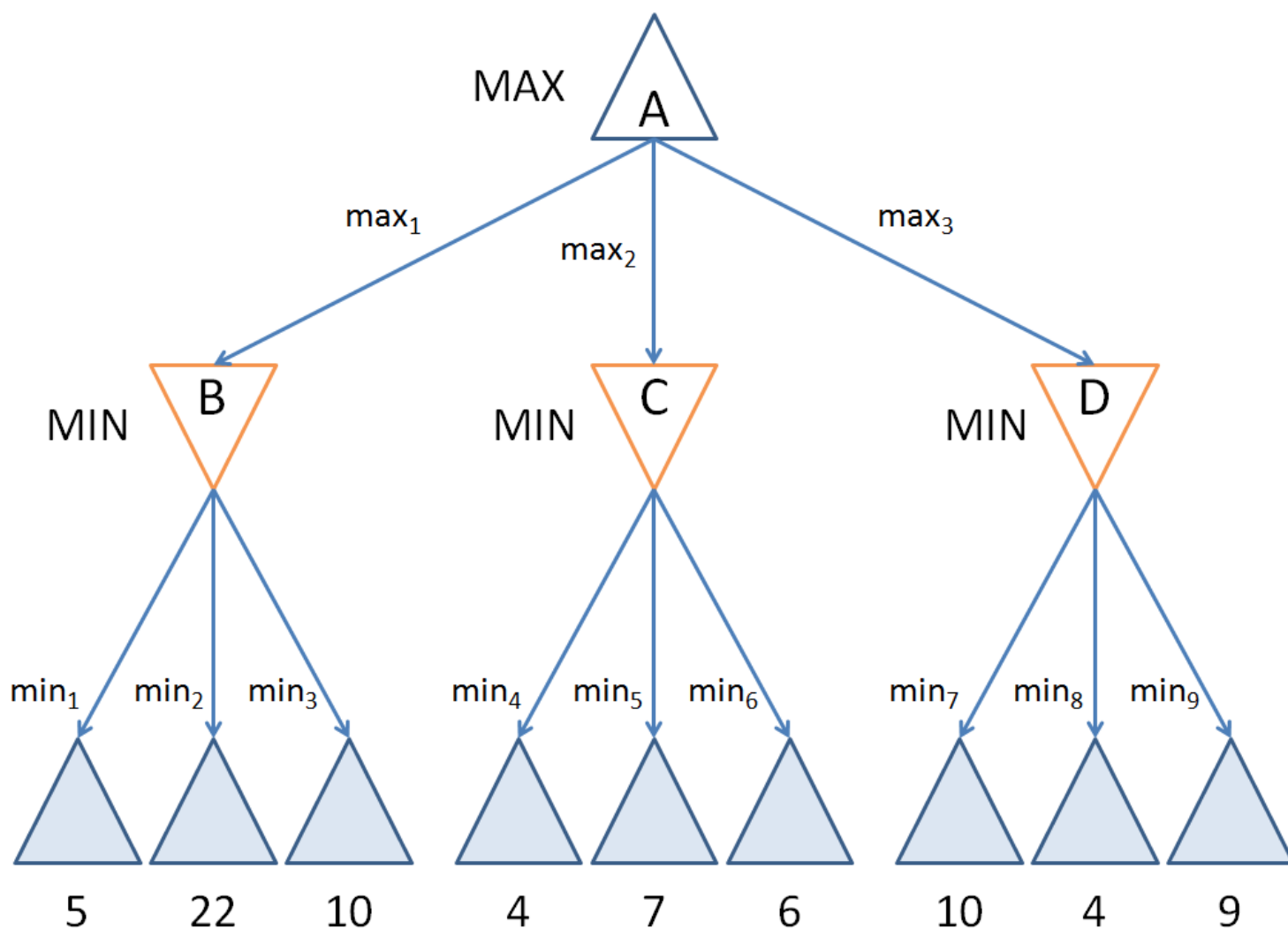
Thus, Beam Search reduces the memory and time complexity tax of a BFS at the cost of pruning some subtrees.

❓ Suppose our beam width is 2, we use the greedy heuristic in scoring nodes along the frontier (i.e., just $h(n)$), and in the event of a tie, we will expand in left-to-right order (nodes along the then-limited frontier will be expanded in order of cost, then left-to-right). What will be the order of expansion for beam search in the search tree above, assuming we do not stop at goal states?

❓ In general, is Beam Search complete?

Example

✔ Suppose we are playing as MAX in the following adversarial search tree, exploring nodes in a DFS, left-to-right order. By α - β pruning, what sub-trees will be trimmed from exploration? Why? [UNDER CONSTRUCTION: Will include more in-depth example for α - β pruning algorithm]



🔗 Click for solution.

Example

🔗 Now that you remember everything there is to know about search, why don't you fill in this table below:

Assume the following variable definitions:

- d = depth of shallowest solution
- b = branching factor
- m = manual depth limit

- L = iteratively increasing depth limit

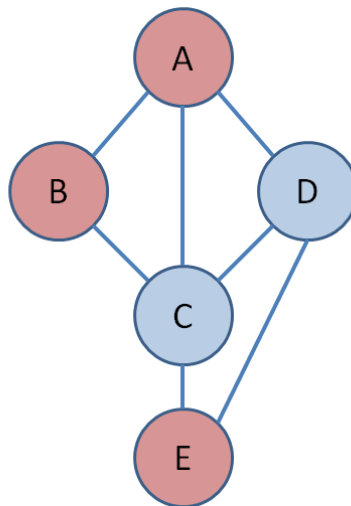
Strategy	Complete?	Optimal?	Time Complexity	Space Complexity
Breadth-first				
Depth-(limited)-first				
Iterative-Deepening-Depth-first				
A*				

🔗 Click here for the solution.

Local Search Problems

Example

🔗 Recall the map-coloring problem where we must find a coloring of each geographic body on a map such that no two bodies share the same color. The following adjacency map depicts such a problem, and represents some state along the solution path of a local search strategy.



Result of random, local search "guess"

A	B	C	D	E
---	---	---	---	---

❓ Suppose we had some state scoring function that counted the number of pairs of same-colored bodies on our map. The goal state is found when this score is 0. Give a possible sequence of actions that local search might take (starting from the above coloring) consisting of the following path. You may change no body's color more than once and may use a maximum of 3 colors:

1. Downward move
2. Upward move
3. Upward move and goal state reached

❓ Starting with the initial state and scoring function depicted and mentioned above, what would be an instance of a sideways move?

Example

☑ Suppose we are looking at a local search problem (whose parameters are unimportant for this example) and have landed on state S with neighbors $N1, N2, N3$ each scored by some function f . Scores are strictly positive and low scores are considered better than others. Thus, the closest to 0, the better.

State	Score (f(state))
S	5
N1	7
N2	6
N3	7

2 True or false: we are at a peak at state S.

Simulated annealing is a local search strategy where we may make downward or sideways moves with some probability proportional to the amount of time that we've been searching.

Typically, we refer to this probability as a function of our **temperature**, which (just like a metal cooling) will decrease over time, giving us much exploration of the search space at the beginning until we "settle" upon a local max near the end.

Now, for our purposes, suppose we have the following simulated-annealing pseudocode denoting the probability of moving to some neighbor; we take three parameters as values:

1. S = current state score (the score of the state we're currently occupying)
2. N = neighbor score (the score of the neighbor state we're considering)
3. T = temperature (higher values means we are more likely to explore worse neighbors)

```
annealingProbability (S, N, T)
  if (N < S)
    return 1.0
  return exp((S - N) / T)
```

2 Suppose our temperature starts out at $T = 100$, cools at a rate of 10% per action taken, and that we have taken 3 steps to reach state S in the table above. What will be the probability of moving to neighbor N2?

❓ Suppose our temperature starts out at $T = 100$, cools at a rate of 10% per action taken, and that we have taken 50 steps to reach state S in the table above. What will be the probability of moving to neighbor N1?

Now, take the following state and neighbors into consideration:

State	Score ($f(\text{state})$)
S	5
N1	7
N2	6
N3	4

❓ Suppose our temperature starts out at $T = 100$, cools at a rate of 10% per action taken, and that we have taken 50 steps to reach state S in the table above. What will be the probability of moving to neighbor N3?

Constraint Satisfaction Problems

❓ Given N variables each with a domain of V values, what is the time complexity needed in order to find a solution?

Example

☑ Given the following CSPs, determine whether or not the domains are arc consistent. If they are not, correct them to be arc consistent.

```
X = { A, B }  
D = { {1, 2, 3},  
      {1, 2, 3, 4, 5, 6, 7, 8, 9} }  
C = { B = A^2 }
```

🔗 Click for answer.

```
X = { S, T, V }  
D = { {1, 2, 3},  
      {1, 2, 3},  
      {1, 2, 3} }  
C = {  
      S != T,  
      T != V,  
      V != S  
}
```

🔗 Click for answer.

```
X = { S, T, V }  
D = { {1, 2, 3},  
      {1, 2, 3},  
      {1, 2, 3} }  
C = {  
      S < T,  
      T > V,  
      V = S  
}
```

🔗 Click for answer.

Propositional Logic

Example

✔ Use truth tables / worlds to show that the following pairs of sentences are logically equivalent.

1. $\text{sent1} = P \Rightarrow \neg Q$
 $\text{sent2} = Q \Rightarrow \neg P$

❓ Click for answer.

2. $\text{sent1} = P \Leftrightarrow \neg Q$
 $\text{sent2} = ((P \wedge \neg Q) \vee (\neg P \wedge Q))$

❓ Click for answer.

Example

✔ Determine if the following propositional logic sentences are valid, unsatisfiable, or neither.

Let S = Smoke
 Fi = Fire
 H = Heat

1. $(S \Rightarrow Fi) \Rightarrow (\neg S \Rightarrow \neg Fi)$
 2. $(S \Rightarrow Fi) \Rightarrow ((S \vee H) \Rightarrow Fi)$
 3. $((S \wedge H) \Rightarrow Fi) \Leftrightarrow ((S \Rightarrow Fi) \vee (H \Rightarrow Fi))$

❓ Click for answers TO ALL 3 of the above.

❓ Amongst the 3 sentences in the previous example, which entail one another?

Example

☑ Perform inference on the following propositional KB to answer the query: $\alpha = A \Rightarrow D$

KB =

1. $(A \vee \neg B) \Rightarrow C$
2. $C \Rightarrow (D \vee \neg E)$
3. $E \vee D$

$\alpha = A \Rightarrow D$

❓ Click for a sample derivation.

First Order Logic

Example

☑ Convert the following English sentences into FOL ones using the relation names specified in brackets.

1. John likes all food. [Food, Likes]
2. Apples are food! [Food]
3. Chicken is food! [Food]
4. Anything that someone eats, that does not kill them, is food. [Eats, Kills, Food]
5. For everyone in general, if something kills you, you are not alive. [Kills, Alive]
6. Bill eats peanuts and is very much alive! [Eats, Alive]
7. Anything Bill eats, Sue also eats. [Eats]

❓ Click for solutions to ALL of the above.

OK, now that we have this riveting story in FOL, let's put it in a usable format.

Example

☑ Convert the above FOL sentences into CNF.

❓ Click for solutions to ALL of the above.

Example

☑ Treating the above CNF sentences as our knowledge base, prove that $KB \models \alpha$ where $\alpha = \text{Likes}(\text{John}, \text{Peanuts})$

❓ Click for a sample derivation.

❓ Could you perform forward or backward chaining on the above KB as it is presently? Why or why not?

