CSE 473

Instructions: Please read through all the questions before you start. We will likely separate the exam into different pages and grade them individually, so it is important to keep all work localized to the page with the question; if you use additional sheets, please: one problem per page.

If any problem is underspecified, feel free to make assumptions. Just state and justify your assumptions. Be neat, concise and clear – you will be graded on the presentation, understandability & clarity of your answer. The correct answer mixed in with a morass of irrelevant facts will likely receive a lower score than the answer presented simply.

There are 4 pages with 6 problems totaling 50 points; make sure you have every page!

1. [5 points] Write your name on the top of every page.

2. [10 points] Suppose that \( h_1 \) and \( h_2 \) are both admissible heuristics to a search problem which you need to solve. Is \( \max(h_1, h_2) \) also admissible? Prove or provide a counterexample.

Let \( f = \max(h_1, h_2) \)

Assume not. Thus there exists some node, \( n \), such that \( f(n) \) overestimates the minimum cost of reaching the goal from \( n \). But then either \( h_1 \) or \( h_2 \) must overestimate this same cost. Thus \( h_1 \) or \( h_2 \) is inadmissible, but this violates our assumption that \( h_1 \) and \( h_2 \) were admissible.
3. Consider the following adversarial search trees. Note that underneath the leaves (e.g., nodes c, d, f, & g in Tree I) we have written the value returned by the heuristic evaluation function.

**Tree I**
```
     a
    / \
   2   e
  /     |
 c   d   f  g
```

**Tree II**
```
     a
   /   |
  b 4   c
 /     |
3   d   i
```

a. [2 points] Using min-max search on Tree I, what is the value of node a?

3

b. [4 points] Suppose Tree I is searched using alpha-beta search in a depth first fashion. E.g., if all nodes were explored, they would be visited in alphabetic order. Would alpha-beta visit all nodes, or would it prune one or more nodes? Answer “no pruning” or for each node which does get pruned, say “node X pruned immediately after visiting node Y”.

No pruning

c. [2 points] Using min-max search on Tree II, what is the value of node a?

4

d. [4 points] Answer question b for Tree II.

Node g pruned at f (because the value of b is greater than the value of f so no value of g could affect the root node’s decision to go left or right). The values discovered at h and i, on the other hand, do still affect that decision so they cannot be pruned.
4. [7 points] If Fred is rich, he is happy. But if he’s not rich, then he’s an unhappy liar. If Fred is either happy or a liar, then he is funny. Fred smells bad when he’s funny. Let \( R \) denote that Fred is rich, \( H \) happy, \( L \) liar, \( F \) funny, and \( S \) smells bad. Using propositional logic, can you prove that Fred is both rich and smelly? Provide either a proof or a succinct argument explaining why no such proof exists.

This problem fooled almost everyone (sorry!). Many people tried to generate a proof, but either failed or made a mistake. In fact, \( R \land S \) doesn’t follow from the premises. In other words its not the case that every truth table which satisfies \( R \Rightarrow H \) (and the other premises), is guaranteed to satisfy \( R \land S \). Put another way, if one adds the negation of the desired conclusion (i.e. \( \neg R \lor \neg S \)) into the knowledge base and try to find a contradiction (e.g. using resolution) you will fail. In full first order logic with functions, you may loop forever trying to generate this contradiction, but since this is a propositional problem you will eventually do every possible resolution (without getting the contradiction) and know that you can stop. Unfortunately, checking to make sure you’ve done all possible resolutions, while easy for a computer to do, is hard for a person to book keep. So a better strategy is to use DPLL to try and find a satisfying assignment for the KB + negated goal. If you find a satisfying assignment, then clearly there can’t be a contradiction.

The first step is to convert to clausal form, just as in resolution, but recall that with resolution we add each newly resolved clause to the clause set. In contrast, with DPLL, the set of clauses gets smaller over time (and clauses tend to get shorter). The objective is either no clauses (which denotes a satisfying assignment) or the presence of an empty clause (which means that the original set o clauses was unsatisfiable).

We start with the following knowledge base:

1. \( R \Rightarrow H \) which we can write in clausal form as: \((\neg R \lor H)\)
2. \( \neg R \Rightarrow (\neg H \land L) \) thus \((R \lor \neg H) \land (R \lor L)\)
3. \((H \land L) \Rightarrow F \) thus \((\neg (H \land L) \lor F)\) thus \((\neg H \lor \neg L \lor F)\)
4. \( F \Rightarrow S \) thus \((\neg F \lor S)\)
5. Let’s now add the negated goal \((\neg R \lor \neg S)\)

Let’s write this on one line:

\[
(\neg R \lor H) \land (R \lor \neg H) \land (R \lor L) \land (\neg H \lor \neg L \lor F) \land (F \lor S) \land (\neg R \lor \neg S)
\]

\( F \) is pure so we set it to true (This is called pure literal elimination + it gives

\[
(\neg R \lor H) \land (R \lor \neg H) \land (R \lor L) \land (\neg R \lor \neg S)
\]

Now \( L \) & \( S \) are pure, so \( L := true, S := false.\)

\[
(\neg R \lor H) \land (R \lor \neg H)
\]

At this point, DPLL has to start searching, suppose it sets \( R := true \) then we have

\[
(H)
\]

Since \( H \) is pure, we know it must be true, we delete the clause, and we have no more clauses.

This means that \{\( F=\text{true}, L=\text{true}, S=\text{false}, R=\text{true}, H=\text{true} \}\} is a satisfying assignment for the KB+negated goal. So there isn’t a contradiction, so \( R \land S \) doesn’t follow from the KB (indeed \( S=\text{false} \)).
5. [7 points] Encode the sentence “Politicians can fool some of the people all the time, and all of the people some of the time, but they can’t fool all of the people all of the time.” Use the following predicates: \( P(x) \) is true when \( x \) is a politician. \( H(y) \) is true when \( y \) is a human being (i.e. a person). \( T(t) \) is true when \( t \) is a time. \( F(x, y, t) \) is true when \( x \) fools \( y \) at time \( t \).

\[
\forall p \exists h \forall t \ (P(p) \land H(h) \land T(t)) \Rightarrow F(p, h, t) \] \land \\
\[
\forall p \exists t \forall h \ (P(p) \land H(h) \land T(t)) \Rightarrow F(p, h, t) \] \land \\
\[
\forall p \exists h \exists t \ (P(p) \land H(h) \land T(t)) \Rightarrow \neg F(p, h, t) 
\]

I think this is the best answer, but other answers are also possible, e.g. switching the order of the quantifiers (\( \exists h \forall t \)) in the first line and the order of (\( \exists t \forall h \)) in the second line. These switches change the meaning of the overall sentence, but one might argue legitimately that the meaning matches that of the English sentence. It’s interesting that logic forces you to be more precise than English (e.g. distinguishing between there is one gullible person who is fooled time after time, or whether each time someone (maybe a different person each time) falls for the lie.
6. Suppose that you decide to create a startup that will produce a filter-service for Internet Service Providers so that their customers (who enable your filter) will be guaranteed\(^1\) never to browse to an X-rated Website. Having just taken CSE 473, you decide to build the filter using machine learning techniques. Your first step is to

a. [4 points] Describe the problem of identifying adult sites as a function approximation problem. What is the function you wish to approximate?

\[ F: \text{website-attributes} \rightarrow \{\text{adult}, \neg\text{adult}\} \]  
\[ F: \text{websites} \rightarrow \{\text{adult}, \neg\text{adult}\} \]

b. [5 points] What features of Web pages would you choose to use as attributes for describing the domain of this function? (This choice is much like the choice of features when building a heuristic function in a game-playing program)

There are many possible answers here, and indeed choice of features (while the single most important decision in building a ML system) is an art, not a science. Here are a few:

- Keywords (e.g. is the word “sex” present)
- Presence of images (perhaps of a certain size. Perhaps with certain hues (skin tone) or certain features (if one had a good computer vision system).
- Linkage structure (does the site point to other known adult sites, to pages that have certain characteristics (e.g. keywords or images, or...) or do other known adult sites point to this one.)

\(^1\) You’ll only want to offer this guarantee, if you can really get your filter to work perfectly, of course.