Supervision Assignments in AI
Easter Term 2019
Set 2: Adversarial Seach, CSPs, Knowledge Representation and Reasoning

Supervisor : Dionysis Manousakas
dm754@cam.ac.uk

May 5, 2019

1. (*AIMA5.7*) Prove the following assertion: For every game tree, the utility obtained by MAX using minimax decisions against a suboptimal MIN will never be lower than the utility obtained playing against an optimal MIN. Can you come up with a game tree in which MAX can do still better using a suboptimal strategy against a suboptimal MIN?

2. (*AIMA5.12*) Describe how the minimax and alphabeta algorithms change for two-player, nonzero-sum games in which each player has a distinct utility function and both utility functions are known to both players. If there are no constraints on the two terminal utilities, is it possible for any node to be pruned by alphabeta? What if the players utility functions on any state differ by at most a constant k, making the game almost cooperative?

3. (*AIMA5.14*) Prove that alphabeta pruning takes time $O(2^{m/2})$ with optimal move ordering, where m is the maximum depth of the game tree.

4. (*AIMA5.18*) Prove that with a positive linear transformation of leaf values (i.e., transforming a value x to $ax + b$ where $a > 0$), the choice of move remains unchanged in a game tree, even when there are chance nodes

5. (*AIMA6.6*) Show how a single ternary constraint such as "$A + B = C$" can be turned into three binary constraints by using an auxiliary variable. You may assume finite domains. (Hint: Consider a new variable that takes on values that are pairs of other values, and consider constraints such as "X is the first element of the pair Y .") Next, show how constraints with more than three variables can be treated similarly. Finally, show how unary constraints can be eliminated by altering the domains of variables. This completes the demonstration that any CSP can be transformed into a CSP with only binary constraints.

6. (*AIMA6.12*) What is the worst-case complexity of running AC-3 on a tree-structured CSP?

7. (*AIMA6.13*) AC-3 puts back on the queue every arc (X_k, X_i) whenever any value is deleted from the domain of X_i, even if each value of X_k is consistent with several remaining values of X_i. Suppose that, for every arc (X_k, X_i), we keep track of the number of remaining values of X_i that are consistent with each value of X_k. Explain how to update these numbers efficiently and hence show that arc consistency can be enforced in total time $O(n^2d^2)$.

8. (*AIMA7.21*) Is a randomly generated 4-CNF sentence with n symbols and m clauses more or less likely to be solvable than a randomly generated 3-CNF sentence with n symbols and m clauses?

9. (*AIMA8.2*) Consider a knowledge base containing just two sentences: $P(a)$ and $P(b)$. Does this knowledge base entail $\forall x\ P(x)$?

10. (*AIMA8.18*) Write out the axioms required for reasoning about the wumpuss location, using a constant symbol Wumpus and a binary predicate At(Wumpus, Location). Remember that there is only one wumpus.

11 (*AIMA 8.22*) Write in first-order logic the assertion that every key and at least one of every pair of socks will eventually be lost forever, using only the following vocabulary: Key(x), x is a key; Sock(x), x is a sock; Pair (x, y), x and y are a pair; Now, the current time; Before(t1, t2), time t1 comes before time t2; Lost (x, t), object x is lost at time t.

12. (*AIMA 9.2*) From $Likes(Jerry, IceCream)$ it seems reasonable to infer $\exists x\ Likes(x, IceCream)$. Write down a general inference rule, **Existential Introduction**, that sanctions this inference. State carefully the conditions that must be satisfied by the variables and terms involved.

13. (*AIMA 9.18*) Suppose that successor(X,Y) is true when state Y is a successor of state X; and that goal(X) is true when X is a goal state. Write a definition for solve(X,P), which means that P is a path (list of states) beginning with X, ending in a goal state, and consisting of a sequence of legal steps as defined by successor. You will find that depth-first search is the easiest way to do this. How easy would it be to add heuristic search control?

14. (*AIMA 10.5*) A finite Turing machine has a finite one-dimensional tape of cells, each cell containing one of a finite number of symbols. One cell has a read and write head above it. There is a finite set of states the machine can be in, one of which is the accept state. At each time step, depending on the symbol on the cell under the head and the machines current state, there are a set of actions we can choose from. Each action involves writing a symbol to the cell under the head, transitioning the machine to a state, and optionally moving the head left or right. The mapping that determines

which actions are allowed is the Turing machines program. Your goal is to control the machine into the accept state. Represent the Turing machine acceptance problem as a planning problem. If you can do this, it demonstrates that determining whether a planning problem has a solution is at least as hard as the Turing acceptance problem, which is PSPACE-hard.

15. *AIMA 10.12* a. Would bidirectional state-space search be a good idea for planning?

    b. What about bidirectional search in the space of partial-order plans?

    c. Devise a version of partial-order planning in which an action can be added to a plan if its preconditions can be achieved by the effects of actions already in the plan. Explain how to deal with conflicts and ordering constraints. Is the algorithm essentially identical to forward state-space search?