Department of Computer Engineering

University of Peradeniya

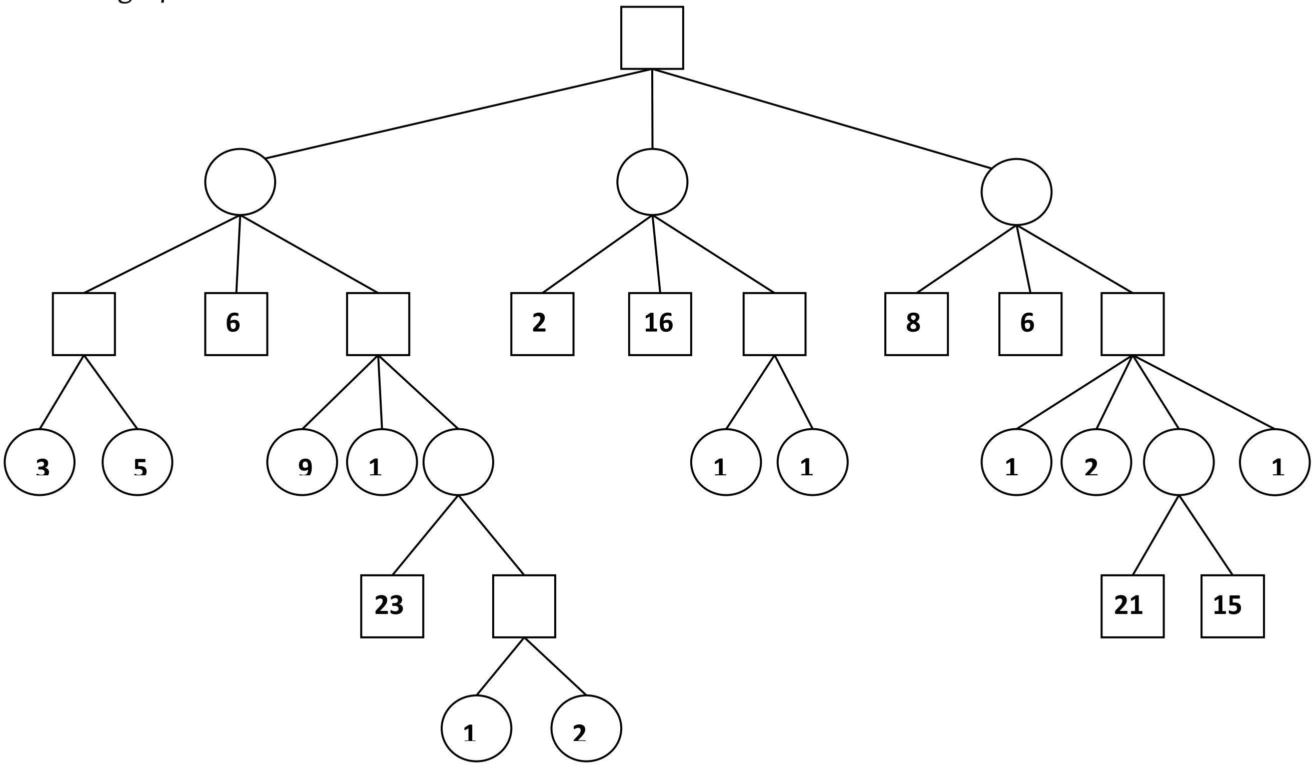
CO541: Artificial Intelligence

Assignment 4

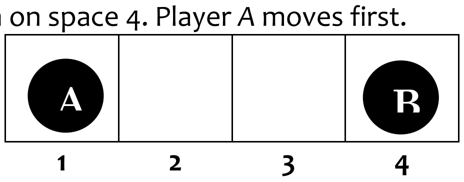
May 31, 2020

1. Draw the smallest possible game tree on which *alpha-beta* will prune at least one leaf node. Make sure to label the leaves with values, and circle the leaf (or leaves) that will be pruned.

2. Mark those leaf nodes of the following game-tree that will not be examined by a left-to-right scanning *alpha-beta* search.



3. Consider a two-player game featuring a board with four locations, numbered 1 through 4, and arranged in a line. Each player has a single token. Player A starts with his token on space 1, and player *B* starts with his token on space 4. Player *A* moves first.



The two players take turns moving, and each player must move his token to an open adjacent space in *either direction*. If the opponent occupies an adjacent space, then a player may jump over the opponent to the next open space, if any (e.g., if *A* is on 3 and *B* is on 2, then *A* may move back to 1). The game ends when one player reaches the opposite end of the board. If player *A* reaches space 4 first, then the value of the game is +1; if player *B* reaches space 1 first, then the value of the game is -1.

a. Draw the complete game tree, using the following conventions:
   - Write each state as (S_A, S_B) where S_A and S_B denote the token locations.
   - Put the terminal states in square boxes, and mark each with its game value in a circle.
   - Put *loop states* (states that already appear on the path to the root) in double square boxes. Since it's not clear how to assign values to loop states, annotate each with a "?" in a circle.

b. Now mark each node with its backed-up minimax value (also in a circle). Explain it in words how you handled the "?" values, and why.

c. Explain why the standard minimax algorithm would fail on this game tree and briefly explain how you might fix it, basing on your answer to (b). Does your modified algorithm give optimal decisions for all games with loops?

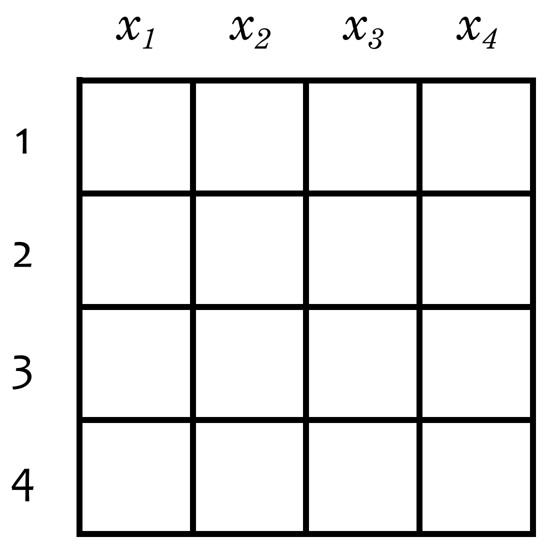
4. Define in your own words the terms constraint satisfaction problem, constraint, backtracking search, arc consistency and min-conflicts.

5. How many solutions are there for the map-colouring problem discussed in the class (i.e., of Australia)?

6. Explain why it is a good heuristic to choose the variable that is *most* constrained, but the value that is *least* constraining in a CSP search.

7. Use the AC-3 algorithm to show that arc consistency is able to detect the inconsistency of the partial assignment {$WA = red,\ V = blue$} for the map-colouring problem mentioned in ***Question* 5**.

8. **Confused Queen Problem (4 Queens):**



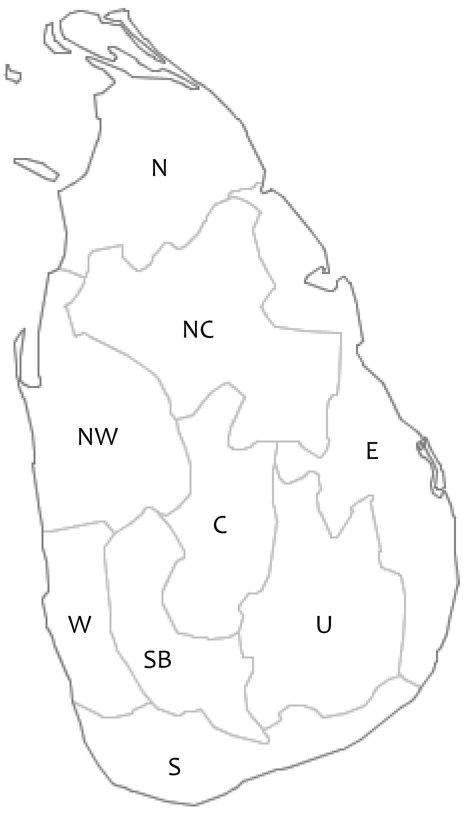
Place the Queens on the board such that each queen is in conflict with all the other Queens

a. Draw the constrained graph for this problem
b. Write the original domains of all variables and all R_{xy} relations
c. Use arc-consistency algorithm on the constrained graph to make it completely arc-consistent. Show the final domains for each x_i

9. Consider the map of Sri Lankan provinces given below.

   a) Draw the constraint graph for the above problem.

   b) Imagine you are asked to colour this map using three colours *Red*, *Blue* & *Green* (*RGB*), where adjacent provinces must have different colours. Show that this cannot be done (Hint: Use the Minimum Remaining Value (MRV) and Most Constraining Variable (MCV) heuristics to assign colours, and apply AC-3 algorithm to check for arc-consistency after each colour assignment).



DEADLINE: 10:00 p.m. on Monday, June 15th, 2017