Lehr- und Forschungsgebiet Informatik 5
RWTH Aachen

Prof. G. Lakemeyer, Ph.D.
J. Claßen, M. Liebenberg
22.11.2016

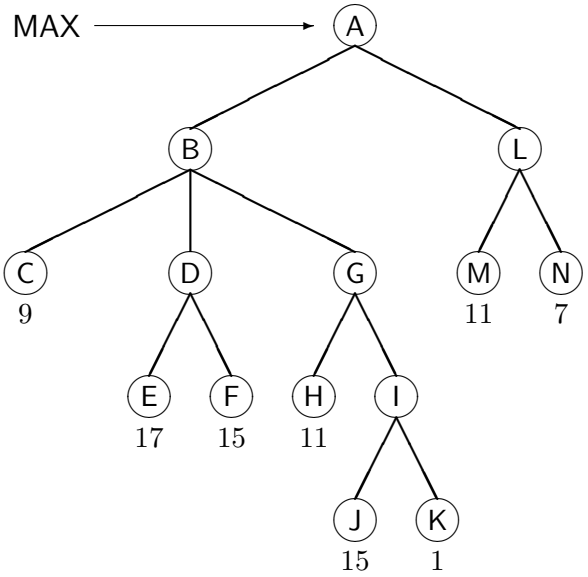
Introduction to Artificial Intelligence (Winter 2016)
3. Assignment

Submit your solution electronically via the L2P until 06.12.2016.

Homework assignments are optional but strongly recommended.

Exercise 3.1 (25 points)

Consider the MAX-MIN game tree shown on the right where the numbers underneath the leaves of the tree are utility values from the first player's point of view (MAX).



(a) Perform minimax on this tree (i. e., compute the minimax values of all nodes).

(b) Perform a left-to-right alpha-beta prune. Which nodes are not examined?

(c) Perform a right-to-left alpha-beta prune. Which nodes are not examined?

Exercise 3.2 (25 points)

The minimax algorithm assumes that players take turns moving, but in card games such as whist and bridge, the winner of the previous trick plays first on the next trick.

(a) Modify the algorithm to work properly for these games. You may assume that a function `Winner(s)` is available that reports which player won the trick just completed (if any).

(b) Draw the game tree for the first pair of hands as shown below. Use the number of won tricks as the utility for the terminal states.

Imagine two players, `MAX` and `MIN`, playing some practice hands of two-card two handed bridge with all the cards showing. The hands are as follows, with `MAX` to play first:

`MAX`: ♣10 ♠3 ♡4 `MIN`: ♣8 ♡7 ♢6

Rules: The leading player may choose any of her cards; the other player has to follow suit (i.e. she has to play a card of the same suit). If the following player does not have a card of the same suit she must throw away some card.

Exercise 3.3 (30 points)

Consider the following two-player game (players A and B):

A			B
1	2	3	4

The starting position of the simple game is shown on the right:
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. (For example, if A is on 3 and B is on 2, then A may move back to 1 or forward to 4.) 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 to A is +1; if player B reaches space 1 first, then the value of the game to A is −1.

(a) Draw the complete game tree using the following conventions:

- Annotate each terminal state with its game value in a circle.
- Treat loop states as terminal states. Since it is not clear how to assign values to loop states, annotate each with a question mark in a circle.

  Loop states are states that already appear on their path to the root at a level in which it is the same player's turn to move.

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

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

(d) This 4-square game can be generalized to n squares for any $n > 2$.
Give a *formal* proof that A wins (A has a winning strategy) if n is even and loses (B has a winning strategy) if n is odd.

Exercise 3.4 (20 Points)

Assume a variation of the two-player game as it is described in Exercise 3.3. Before a player moves she flips a fair coin (i.e., $P(head) = P(tails) = 0.5$). If the coin shows head the player may move one square in the one or the other direction; if the coin shows tails she may move two squares. In case that no move is possible (e.g., the only possible square to move to is alreay occupied by the other player) the game is over and the other player wins. (We assume that "jumping" now requires the coin toss to show tails.)

(a) Draw the complete game tree for the first two rounds (each player moves twice). In the initial state (depicted below), player A is on square 2, player B is on square 4, and player A is allowed to move one square. Label the leaf nodes with +1 if player A wins the game, with -1 if player B wins the game, and with 0 if the game isn't decided, yet.

	A^1		B
1	2	3	4

(b) Perform the EXPECTIMINIMAX-algorithm on the game tree. What is the best action for player A to take in the initial state? Left or right?