Artificial Intelligence

Lecture 4

Ph. D. Lect. Horia Popa Andreescu

2017-2018 3rd year, semester 5

- The slides for this lecture are based (partially) on chapter 6 of the Stuart Russel Lecture Notes [R, ch6], and on the same chapter from Russel & Norvig's book [RN, ch. 6]

Adversarial search

- In a multiagent environment, the agents can cooperate for attending the goal or can compete for the same purpose.
- Competitive environments, in which the agents goals are in conflict, give rise to **adversarial search problems**, also known as **games**.
- A branch of mathematics – **game theory** studies these type of problems, the applications of that are mainly in economics.

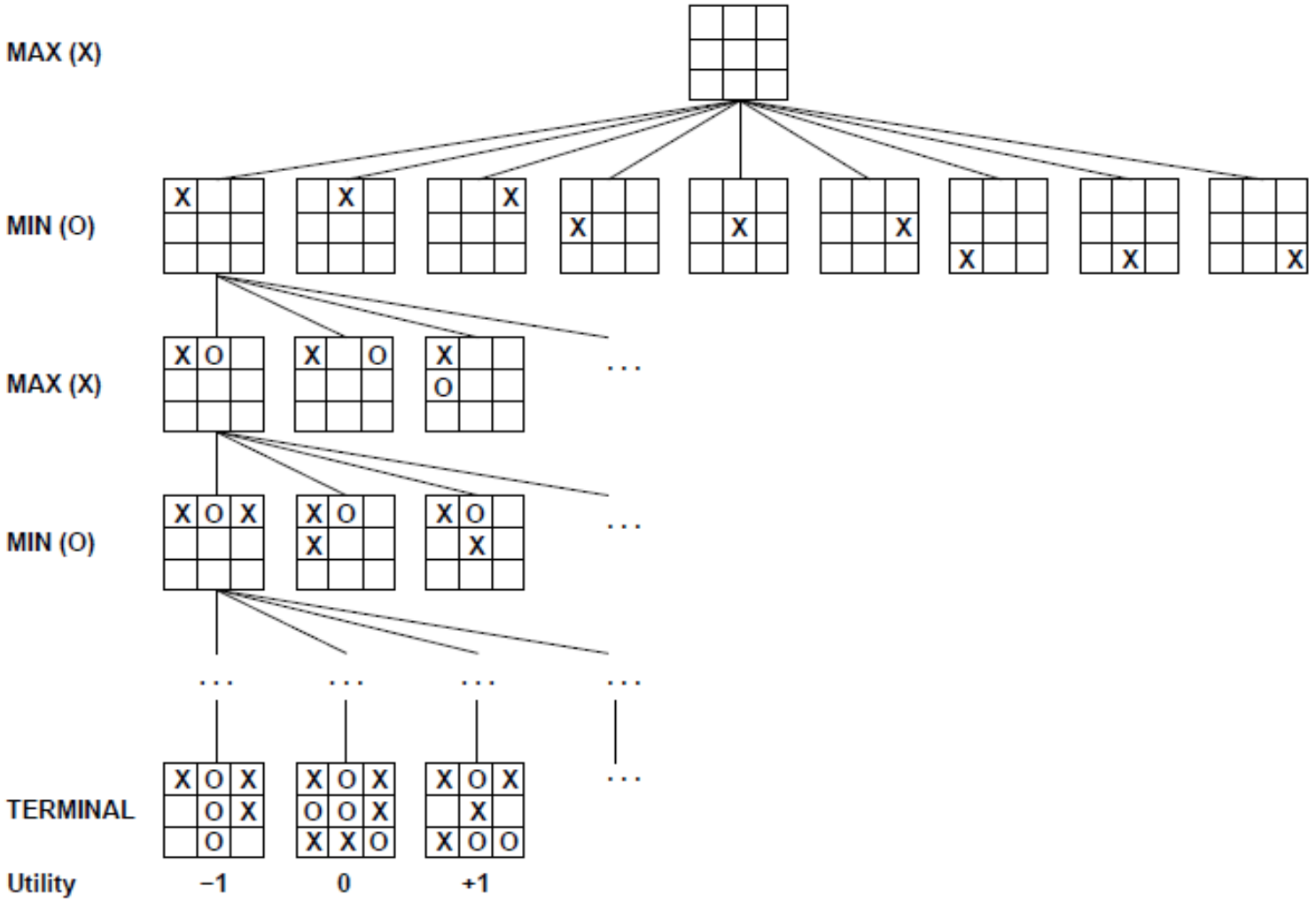
- AI games are of the following types:
  - Deterministic
  - Turn-taking
  - Two-player
  - Zero-sum games (e.g. Chess)
  - Imperfect information games (e.g. Bridge)
  - Other categories:
    - Multiplayer games
    - Non-zero-sum games (participants can all gain or suffer together)
    - Stochastic games (the transition between the game states is determined by player decisions + random)

MIN-MAX GAMES

- MIN-MAX games:
- Are two-player games in which one user tries to maximize the utility function, the other to minimize it.
- The game consists of:
  - The initial state – the board position, who's turn is to move
  - A successor function – it returns a list of (move, state) pairs
  - A terminal test – determines whether the game is over
  - A utility function (or objective function, or payoff function) – gives a numeric value for the terminal states (e.g. for backgammon, the values range are from -192 to +192)

- The initial state and the legal moves for each side form the **game tree.**
- For the tic-tac-toe game, for example, the root node contains the empty play table, on the first level being the nine possible moves that the first player can play, on the second level the responses of the opponent, and so on.
- The terminal states are those with one of the sides having three in a line, thus a win situation. Also there are terminal states in which neither of the opponents has won – draws.

Example. Tic-tac-toe and min-max algorithm (R-6-5)



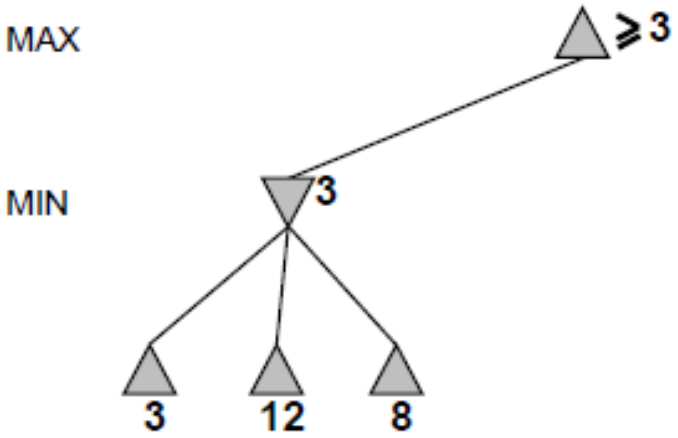
Aplha-beta pruning

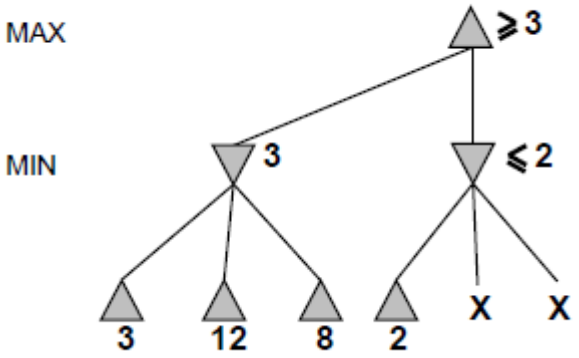
- Used generaly for two player zero sum games.
- The ideea is to prune from the tree nodes, not developing all of them. The algorithm uses 2 values
  - alpha – the value of the best (i.e., highest-value) choice we have found so far at any choice point along the path for MAX
  - beta – the value of the best (i.e., lowest-value) choice we have found so far at any choice point along the path for MIN
  - The current value V, a value obtained from the child nodes as the best value for that node (a maximum or a minimum), depending if it is a maximum or a minimum node

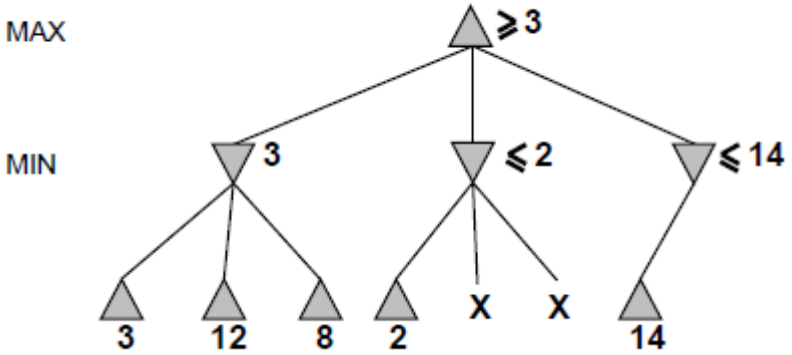
The alpha-beta algorithm [R-6-19]

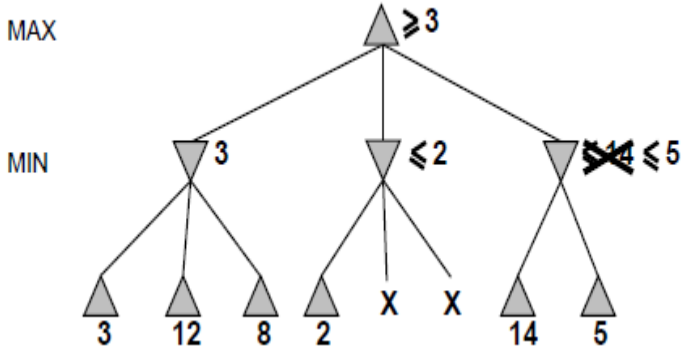
```
function ALPHA-BETA-DECISION(state) returns an action
    return the a in ACTIONS(state) maximizing MIN-VALUE(RESULT(a, state))

---

function MAX-VALUE(state, α, β) returns a utility value
    inputs: state, current state in game
            α, the value of the best alternative for MAX along the path to state
            β, the value of the best alternative for MIN along the path to state
    if TERMINAL-TEST(state) then return UTILITY(state)
    v ← −∞
    for a, s in SUCCESSORS(state) do
        v ← MAX(v, MIN-VALUE(s, α, β))
        if v ≥ β then return v
        α ← MAX(α, v)
    return v

---

function MIN-VALUE(state, α, β) returns a utility value
    same as MAX-VALUE but with roles of α, β reversed
```

Alpha-beta pruning example [R-6-12]









- [for now the lecture continues with the slides from [R, ch. 6, slide 20], until the final form of the lecture, which will be completed in the future]

Bibliography

- [RN] Russel S., Norvig P. – Artificial Intelligence – A Modern Approach, 2nd ed. Prentice Hall, 2003 (1112 pages)
- [R] Stuart Russel – Course slides (visited oct. 2012 at http://aima.cs.berkeley.edu/instructors.html#homework)