Artificial Intelligence

3. Adversarial Search

K. Buza, Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute of Economics and Information Systems
& Institute of Computer Science
University of Hildesheim
http://www.ismll.uni-hildesheim.de

Szenario

- Mulitagent environemts
  - Cooperative
  - Competitive
- Contingencies (unpedictability of other agents)
- Agents goals are in conflict: adversarial search (game)

Game Theory

- Branch of mathematic/economics
- Game
  - Multiagent environment, significant impact on each other
- AI: special kind of games
  - Deterministic
  - Turn-taking (alternating agents)
  - Two player
  - Zero sum (utility values are equal but opposite)
  - Perfect information (full observable)

Game as search problem

- Initial state
- Successor function (legal moves)
- Terminal test (terminal states ≈ goal states)
- Utility function

- Two players: MAX, MIN
- Game tree: half-move = ply

MINIMAX-ALGORITHM

- MINIMAX-VALUE(n) =
  - = UTILITY(n), if n is a terminal state
  - = max(MINIMAX-VALUE of the successors of n), if n is a MAX node
  - = min(MINIMAX-VALUE of the successors of n), if n is a MIN node

- Recursive counting of the minimax-value
- Gives optimal decision in games
- Space Complexity of the algorithm: $O(m)$
- Time Compelxity of the algorithm: $O(b^m)$

Optimal decisions in multiplayer games

- Modified MINIMAX-algorithm
  - Vector of utilities

- Collaboration, Alliances

Alpha-Beta Pruning

- α = best choice we have found so far for MAX
- β = best choice we have found so far for MIN
- Subtrees not improving the utility are not visited
- Reduces complexity
  - „ideal“ ordering of child-nodes: $O(b^{m/2})$
  - random ordering: $O(b^{3m/4})$
  - Chess: the „ideal“ complexity almost reachable

Real-time Decisions

- Evaluation functions
  - Expected value of utility
  - Features
  - EVAL(s) = $w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$
- Cutting off search
  - At a given depth d
  - Quiescene search
  - Horizon effect
  - Singular extensions
  - Forwald pruning

Games with element of chance

- EXPECT-MINIMAX-VALUE(*n*) =

  = UTILITY(*n*), if n is a terminal state

  = max(MINIMAX-VALUE of the successors of *n*), if *n* is a MAX node

  = min(MINIMAX-VALUE of the successors of *n*), if *n* is a MIN node

  = ∑ P(*s*) EXPECT-MINIMAX-VALUE(*s*), for all successors *s* of *n*,

  if *n* is a chance node