Artificial Intelligence (CS303)

Lecture 3: Adversarial Search

Hints for this lecture

- The ***environment*** may, in many cases, involves more than one agents.
- ***Adversarial*** means agents compete to maximize their own benefits.

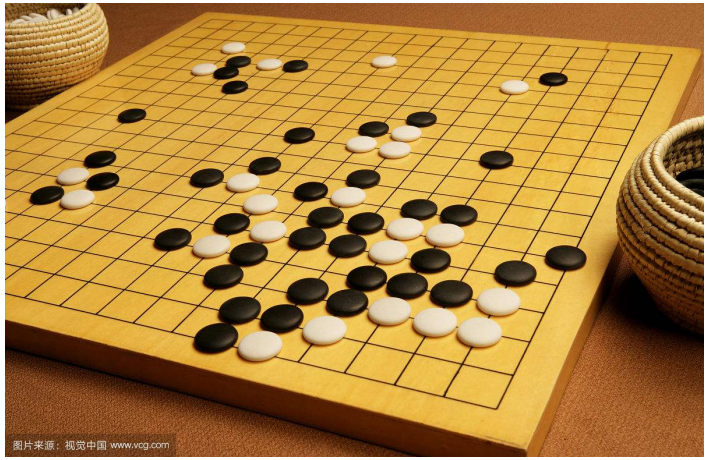
Outline of this lecture

- **Games**
- **Alpha-Beta Pruning**
- **Generalization of Minimax Algorithm**

I. Games

What is Game?

- More than 1 agent, with conflicting goals.
- We consider two players, zero-sum games.

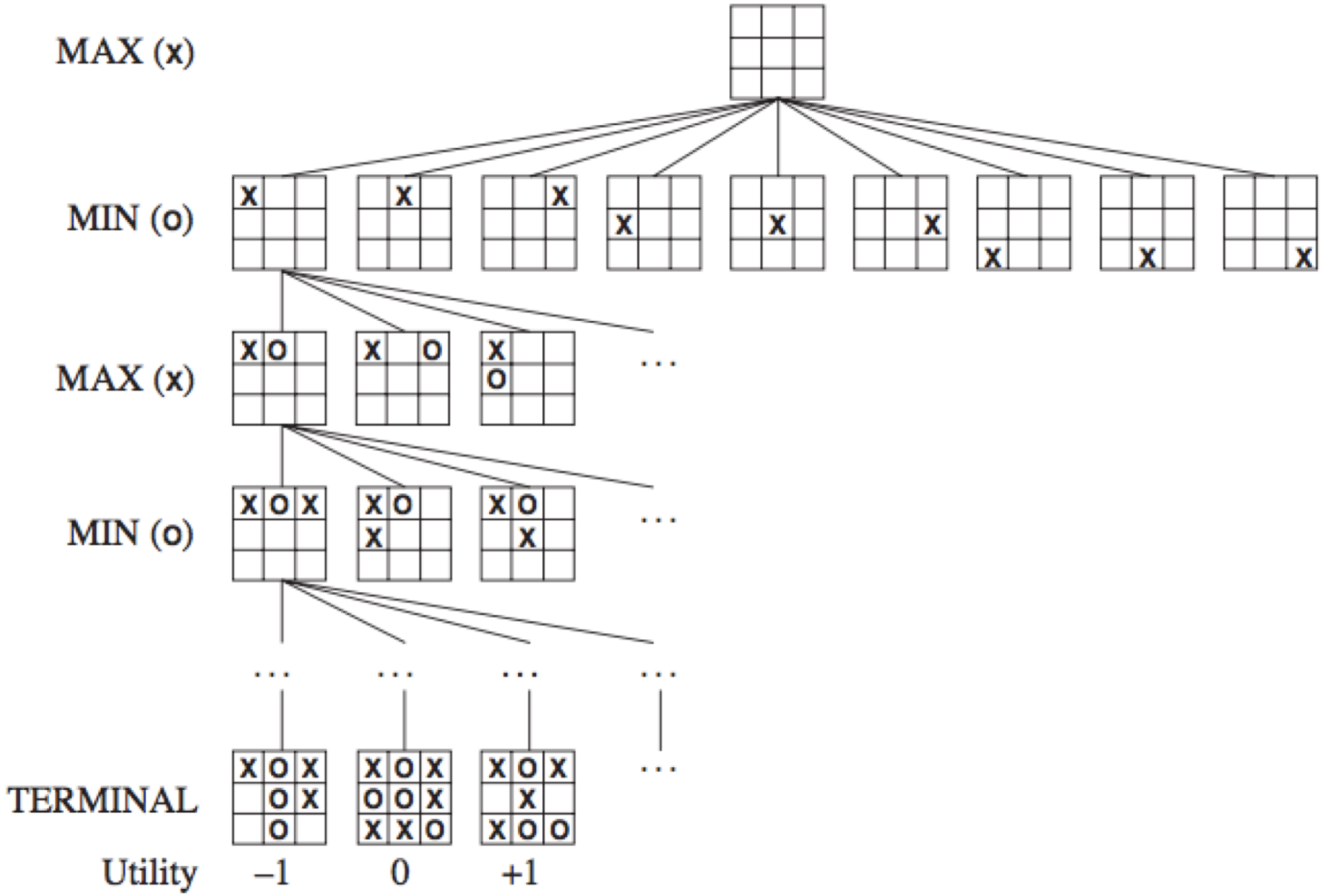


Definition of a game

- S_0: The **initial state**, which specifies how the game is set up at the start.
- PLAYER(s): Defines which player has the move in a state.
- ACTIONS(s): Returns the set of legal moves in a state.
- RESULT(s, a): The **transition model**, which defines the result of a move.
- TERMINAL-TEST(s): A **terminal test**, which is true when the game is over and false otherwise. States where the game has ended are called **terminal states**.
- UTILITY(s, p): A **utility function** (also called an objective function or payoff function),

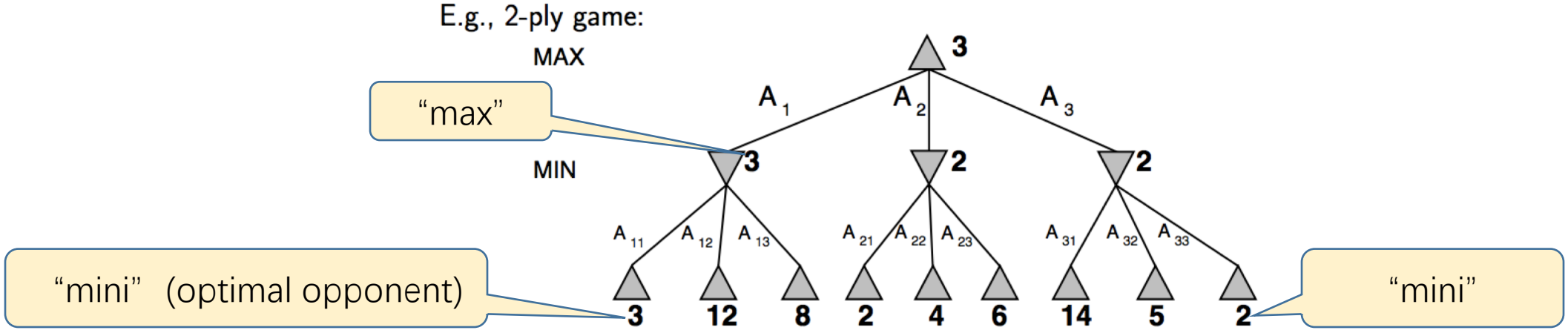
Two players: MAX and MIN

Tic-Tac-Toe Search Tree



Optimal decision in games

- **Assume the game is deterministic and perfect information is available**
- **Idea (for MAX): choose the move to position the highest minimax value**



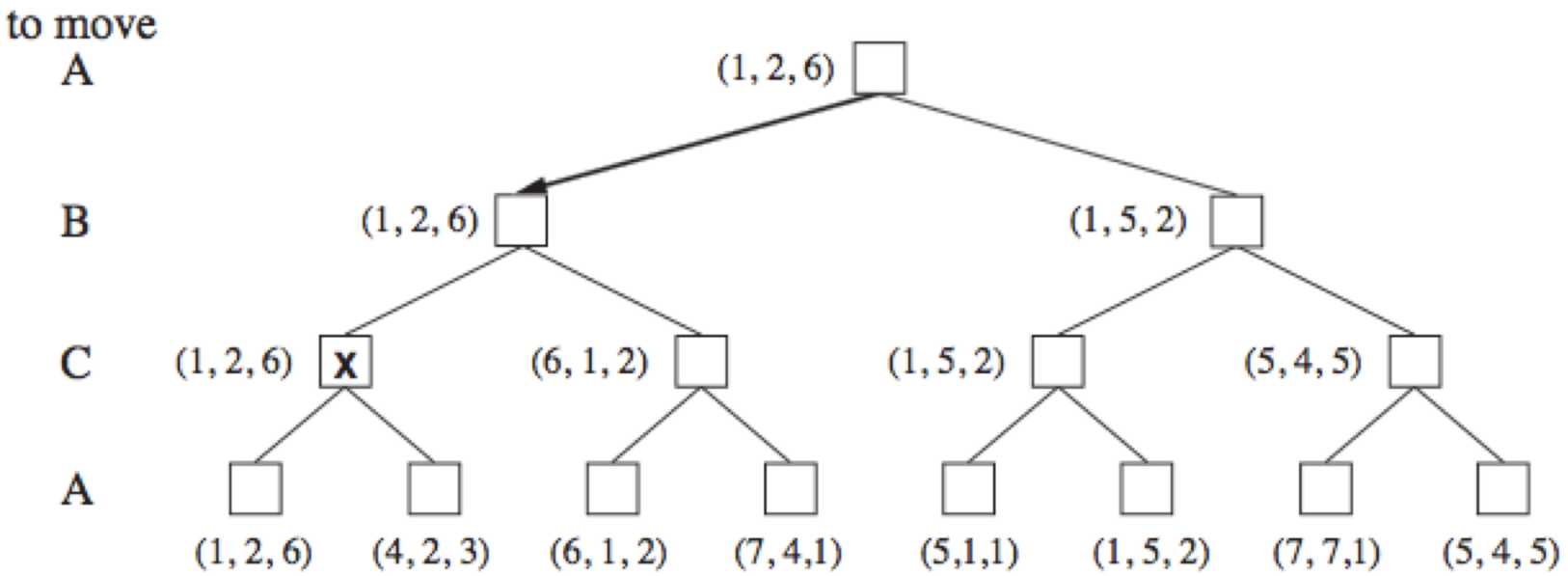
$$\text{MINIMAX}(s) = \begin{cases} \text{UTILITY}(s) & \text{if TERMINAL-TEST}(s) \\ \max_{a \in Actions(s)} \text{MINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MAX} \\ \min_{a \in Actions(s)} \text{MINIMAX}(\text{RESULT}(s,a)) & \text{if PLAYER}(s) = \text{MIN} \end{cases}$$

Minimax Algorithm

- **Perform a complete depth-first search of the game tree.**
- **Recursively compute the minimax values of each successor state.**
- **Maximize the worst-case outcome for MAX.**

Minimax Algorithm

- **Use a vector to represent the utility in case of more than 2 players.**

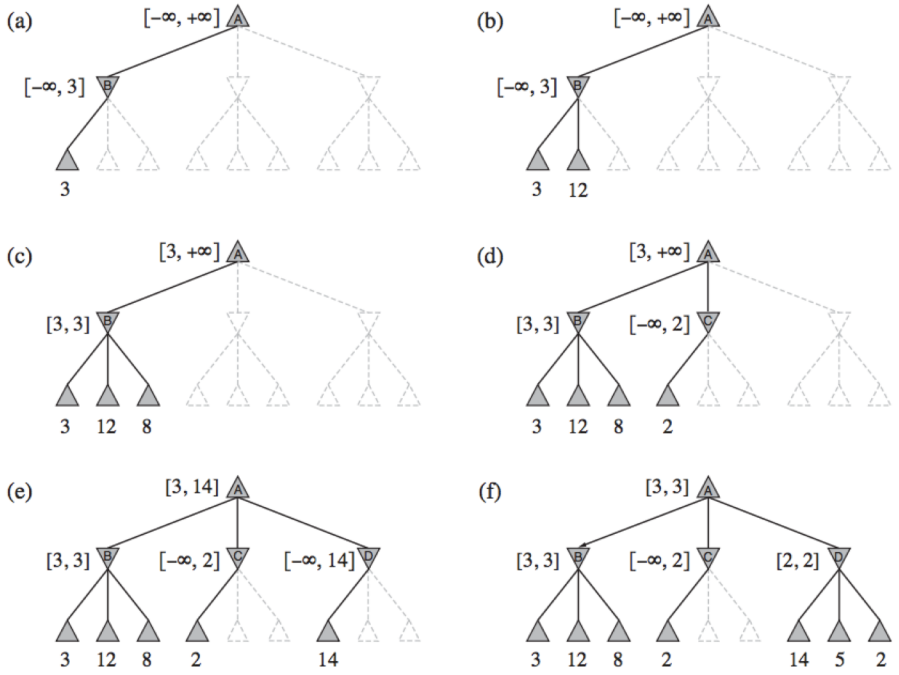


Is minimax algorithm still optimal in this case?

II. Alpha-Beta Pruning

Alpha-Beta Pruning

- **A simplification of minimax algorithm.**
- **Remove (unneeded) part of the minimax tree from consideration.**



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**.

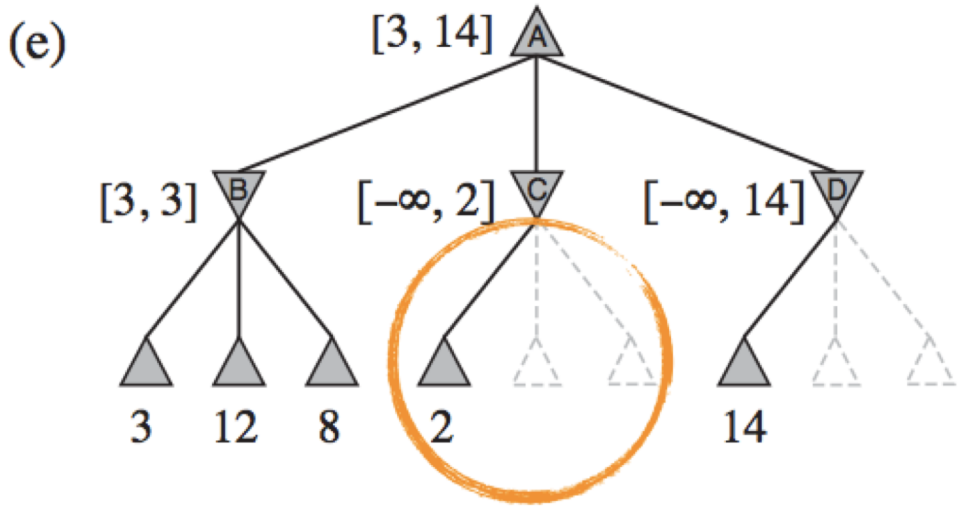
Beta: the value of the best (i.e., **lowest**-value) choice we have found so far at any choice point along the path for **MIN**.

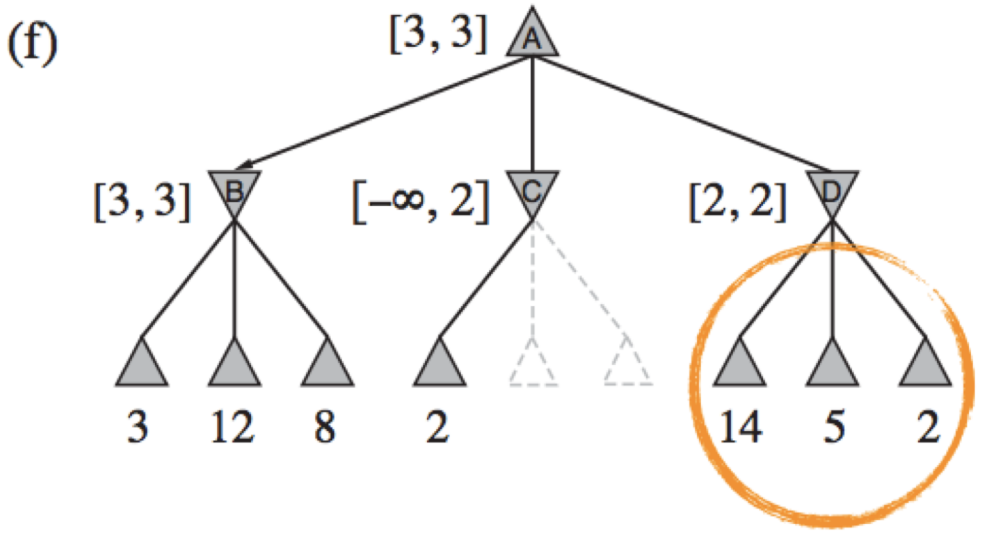
Alpha-Beta Pruning

```
function ALPHA-BETA-SEARCH(state) returns an action
  v ← MAX-VALUE(state, −∞, +∞)
  return the action in ACTIONS(state) with value v

function MAX-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s,a), α, β))
    if v ≥ β then return v
    α ← MAX(α, v)
  return v

function MIN-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← +∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s,a) , α, β))
    if v ≤ α then return v
    β ← MIN(β, v)
  return v
```

The search order is important

- It might be worth while to examine first the successors that are likely to be the best.





Further acceleration?

- Alpha-beta pruning does **not** affect the final result. (good!)
- With perfect ordering, the time complexity is still high, i.e., O($b^{m/2}$). (bad…)

In practice:

- Replace the **UTILITY** function with a heuristic evaluation function **EVAL**
- Use **CUTTOFF-Test** instead of **TERMINAL-Test**

$$\text{H-MINIMAX}(s,d) = \begin{cases} \text{EVAL}(s) & \text{if CUTOFF-TEST}(s,d) \\ \max_{a \in Actions(s)} \text{H-MINIMAX}(\text{RESULT}(s,a), d+1) & \text{if PLAYER}(s) = \text{MAX} \\ \min_{a \in Actions(s)} \text{H-MINIMAX}(\text{RESULT}(s,a), d+1) & \text{if PLAYER}(s) = \text{MIN}. \end{cases}$$

III. Generalization of Minimax Algorithm

Stochastics Game

Deterministic game: Scale of the payoff does not matter.

Stochastic game: Scale does matter

Partially Observable Games

Texas Holdem

Starcraft

With partial observability, value of an action depends on the information state or belief state the agent is in.

Minimax Optimization

$$\max_{x \in X} \min_{u \in U(x)} f(x, u)$$

- Beyond the tree representation (again).
- An agent can be a general search method.
- Commonly encountered in robust optimization (design, transportation, logistics, etc).

To be continued