

Games

Agenda

- game-theory
- minimax algorithm
- alpha-beta pruning
- ... but first, our homework exercise

Game Definition

- initial state
- successor function: lists legal (move, state') pairs
- terminal test: when is game over?
- utility function: value of terminal states

Optimal Decisions in 2-player games

- search for sequence of moves that lead to terminal state with positive utility (winning state)
- but opponent might not be so cooperative!
- solution: find strategy that leads to winning state regardless of what opponent does

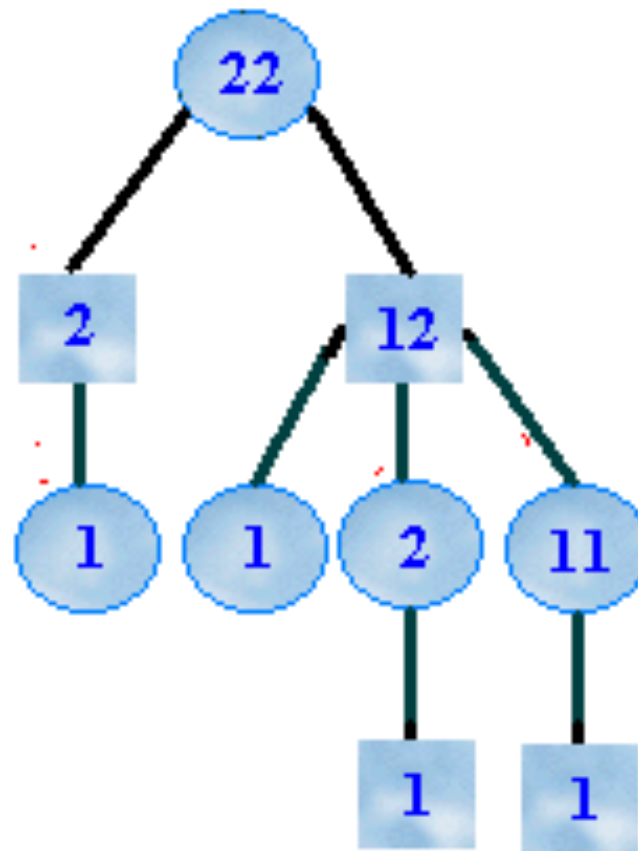
Minimax strategy for 2-player games

- generate whole game tree down to terminal nodes
- find value of each terminal state using **utility function**
- repeat
 - determine utility of parent nodes from children
 - **MIN** chooses move that minimizes utility
 - **MAX** chooses move that maximizes utility
- until we reach root
- choose move that leads to “best” value

MiniMax algorithm

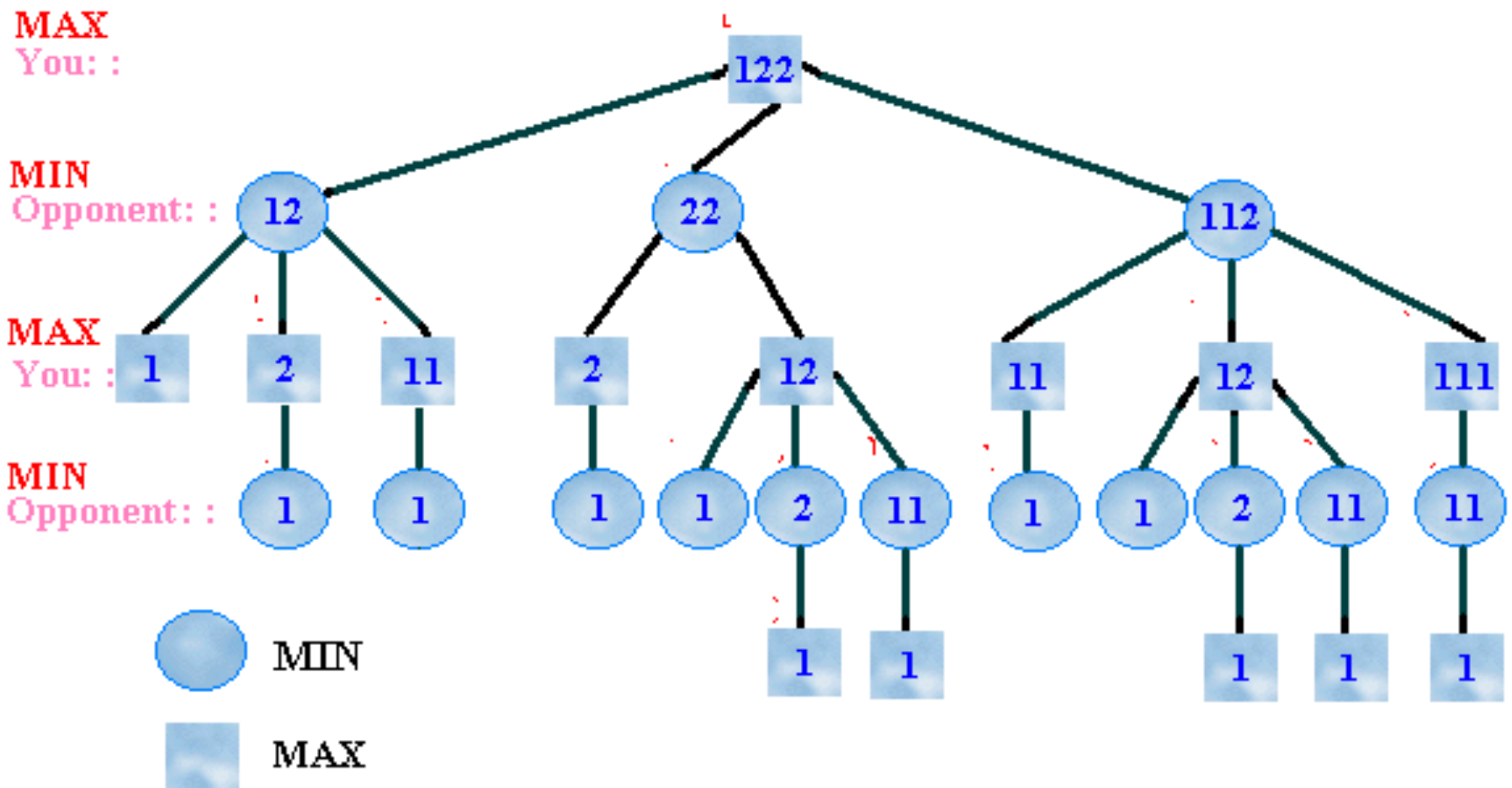
```
minmax(u) { // u is node to evaluate
  if u is a leaf return score of u;
  else if u is a min node
    for all children of u: v1, .. vn
      return min{ minmax(v1),..., minmax(vn)}
  else // u is a max node
    for all children of u: v1, .. vn
      return max{ minmax(v1),...,minmax(vn)}
}
```

NIM - player to take last stick loses



Exercise: build tree for NIM 122...

Expanded tree

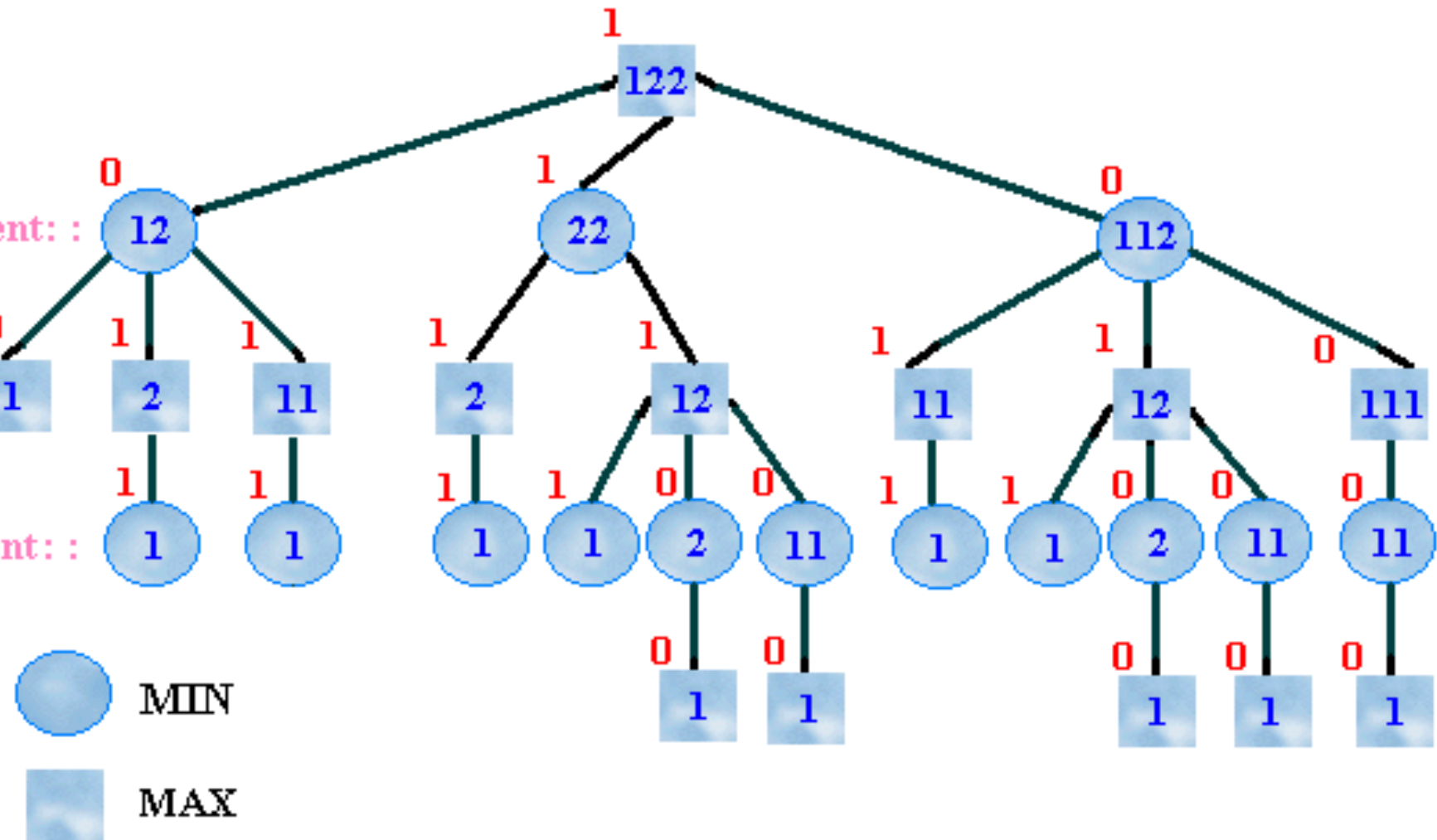


MAX
You: :

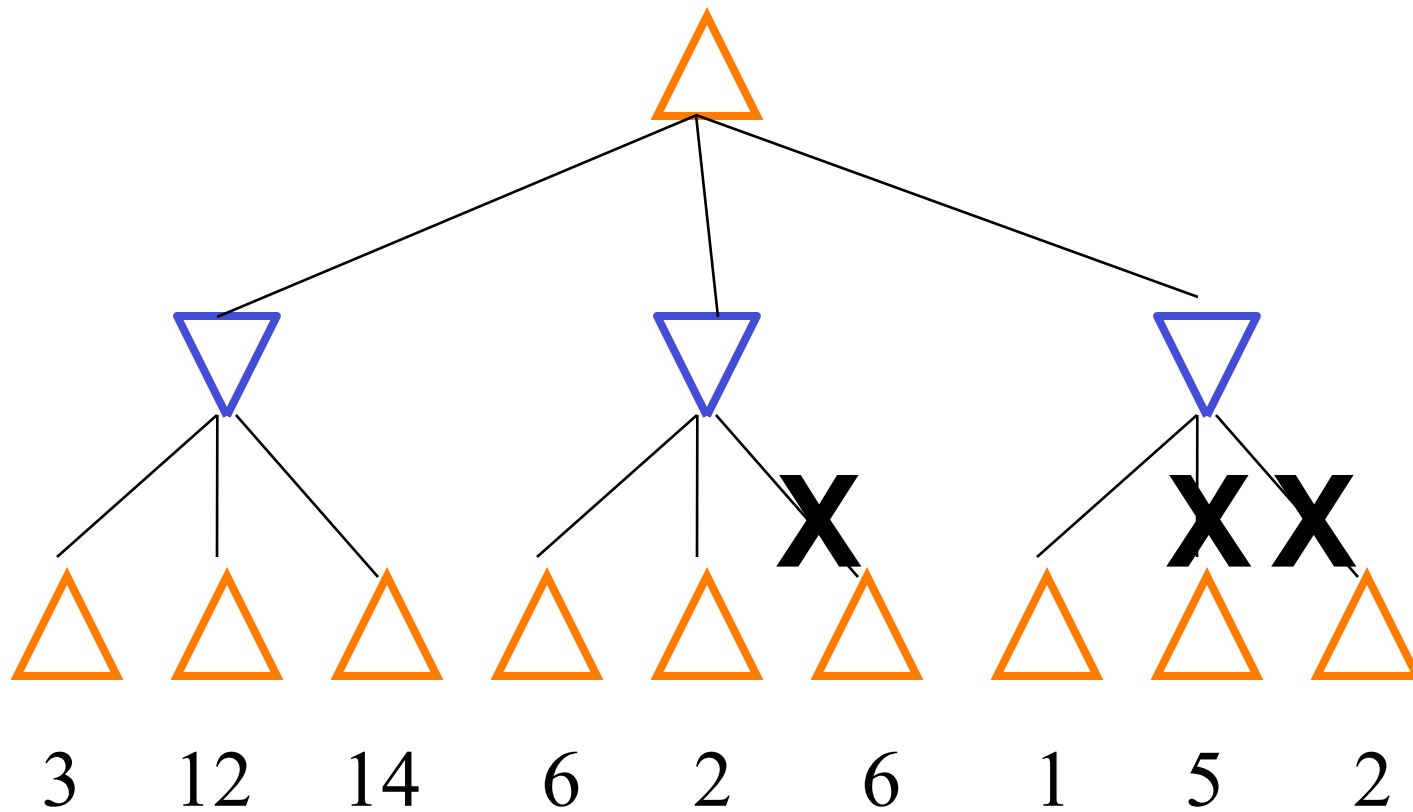
MIN
Opponent: :

MAX
You: :

MIN
Opponent: :



Improving Search: Alpha-Beta Pruning



Alpha-Beta definitions

α *minimal score **MAX** is guaranteed*

β : *maximum score **MAX** can hope to obtain (against a sensible opponent)*

Alpha-Beta pseudocode

```
function MaxV(state,  $\alpha$ ,  $\beta$ )  
if CUTOFF-TEST(state) then  
    return EVAL( state )  
for each s in SUCC(state) do  
     $\alpha \leftarrow$  MAX( $\alpha$ , MinV(s,  $\alpha$ ,  $\beta$ ))  
    if  $\alpha \geq \beta$  then return  $\beta$   
return  $\alpha$ 
```

```
function MinV(state,  $\alpha$ ,  $\beta$ )  
if CUTOFF-TEST(state) then  
    return EVAL( state )  
for each s in SUCC(state) do  
     $\beta \leftarrow$  MIN ( $\beta$ , MaxV(s,  $\alpha$ ,  $\beta$ ))  
    if  $\alpha \geq \beta$  then return  $\alpha$   
return  $\beta$ 
```

Problem

- usually impossible to explore entire state space (e.g., chess search tree has approx. 35^{100} nodes)
- infeasible to make optimal decision
- solution: use estimate of utility of states based on insight

Position Evaluators

- Game specific
 - have to be creative!
- Othello [IAGO: Rosenbloom 1988]
 - edge/corner control
 - mobility
 - stability