

# 敵対探索

Chapter 6

Section 1 – 4

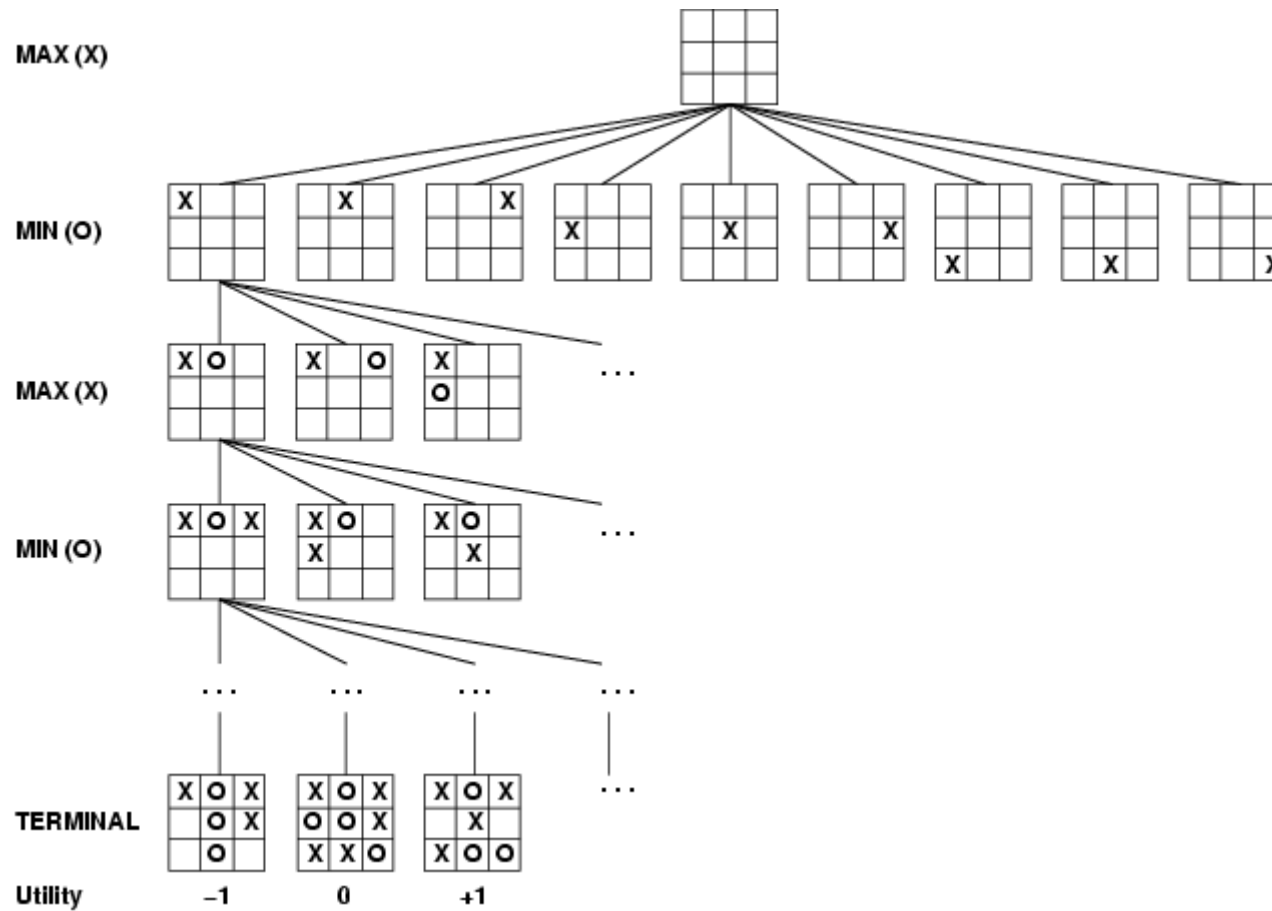
# 概要

- 最適決定
- $\alpha$ - $\beta$  刈込み
- 不完全な実時間の決定

# ゲームvs探索問題

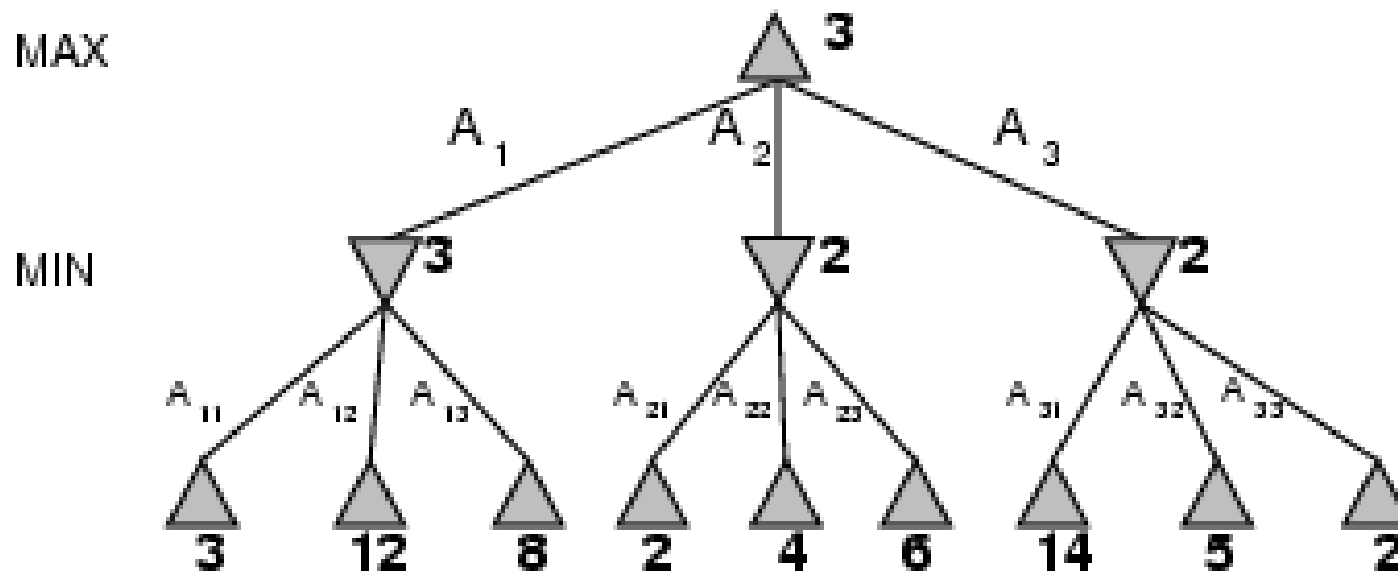
- “予想できない” 相手 → 可能な敵の打ち手に対して自分の打ち手を示すことができる
- 時間制限 → ゴールを見つけることはなかなかないので、近似となる

# ゲーム木 (2プレイヤー, 決定的, 交互)



# Minimax

- 決定的なゲームに対する完全なプレー
- 考え方: 最も高得点の **minimaxの値** を選択する  
= 敵の最善のプレイに対して最も報いのある打ち手
- 例えば2手先を読むゲーム:



# Minimaxアルゴリズム

**function** MINIMAX-DECISION(*state*) *returns an action*

$v \leftarrow \text{MAX-VALUE}(state)$

**return** the *action* in SUCCESSORS(*state*) with value *v*

---

**function** MAX-VALUE(*state*) *returns a utility value*

**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)

$v \leftarrow -\infty$

**for** *a, s* in SUCCESSORS(*state*) **do**

$v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(s))$

**return** *v*

---

**function** MIN-VALUE(*state*) *returns a utility value*

**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)

$v \leftarrow \infty$

**for** *a, s* in SUCCESSORS(*state*) **do**

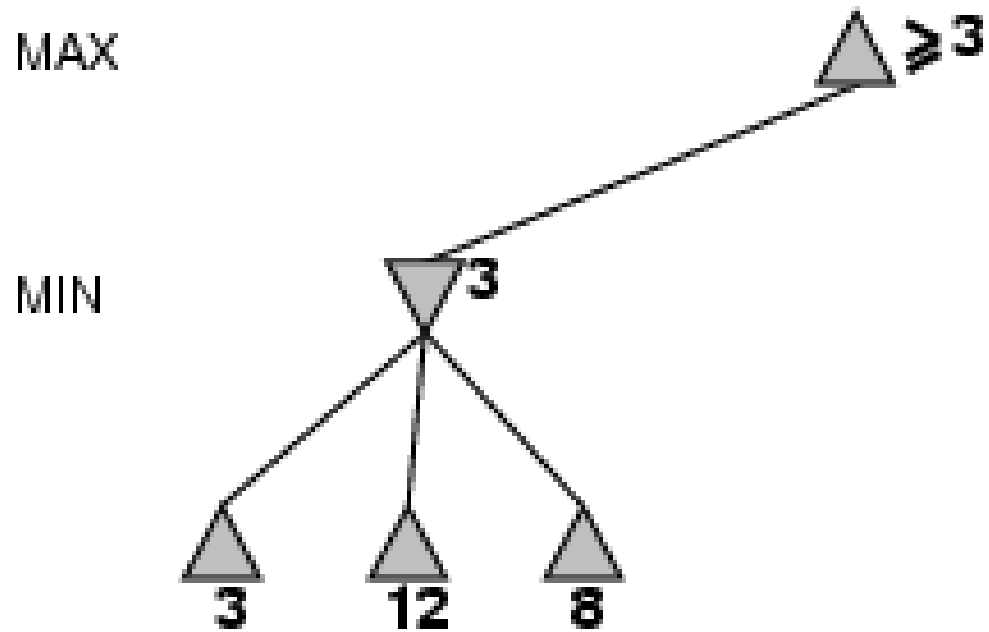
$v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(s))$

**return** *v*

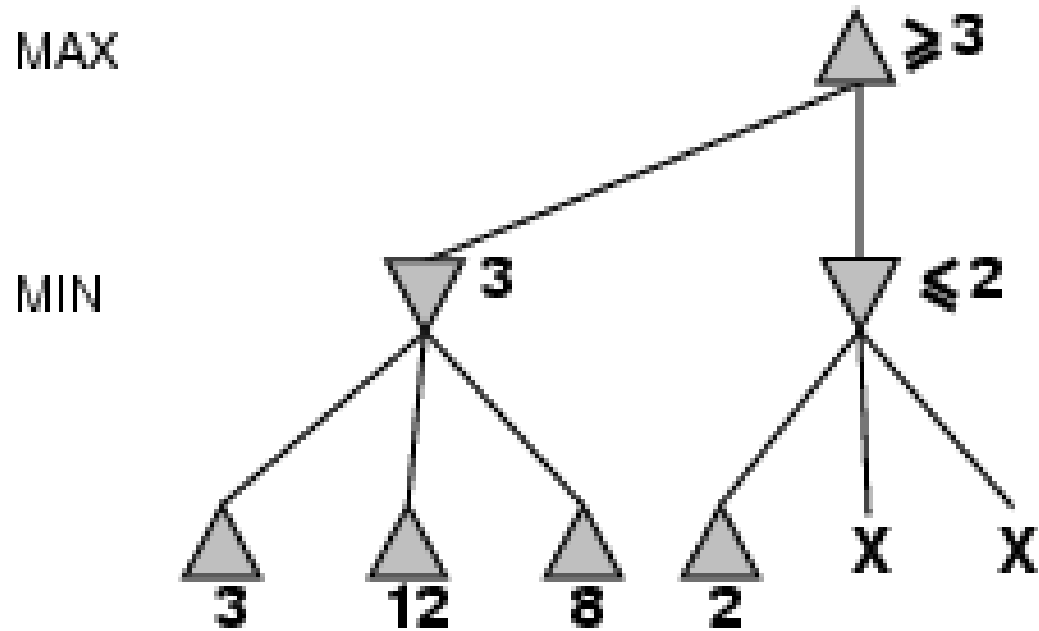
# minimaxの性質

- Complete? Yes (if tree is finite)
- Optimal? Yes (against an optimal opponent)
- Time?  $O(b^m)$
- Space?  $O(bm)$  (depth-first exploration)
  
- 例えばチェスでは, “ほどほどの”ゲームに対しては $b \approx 35$ ,  $m \approx 100$   
→ 本当に全ての解に対して探索しなければならないか。これは不可能

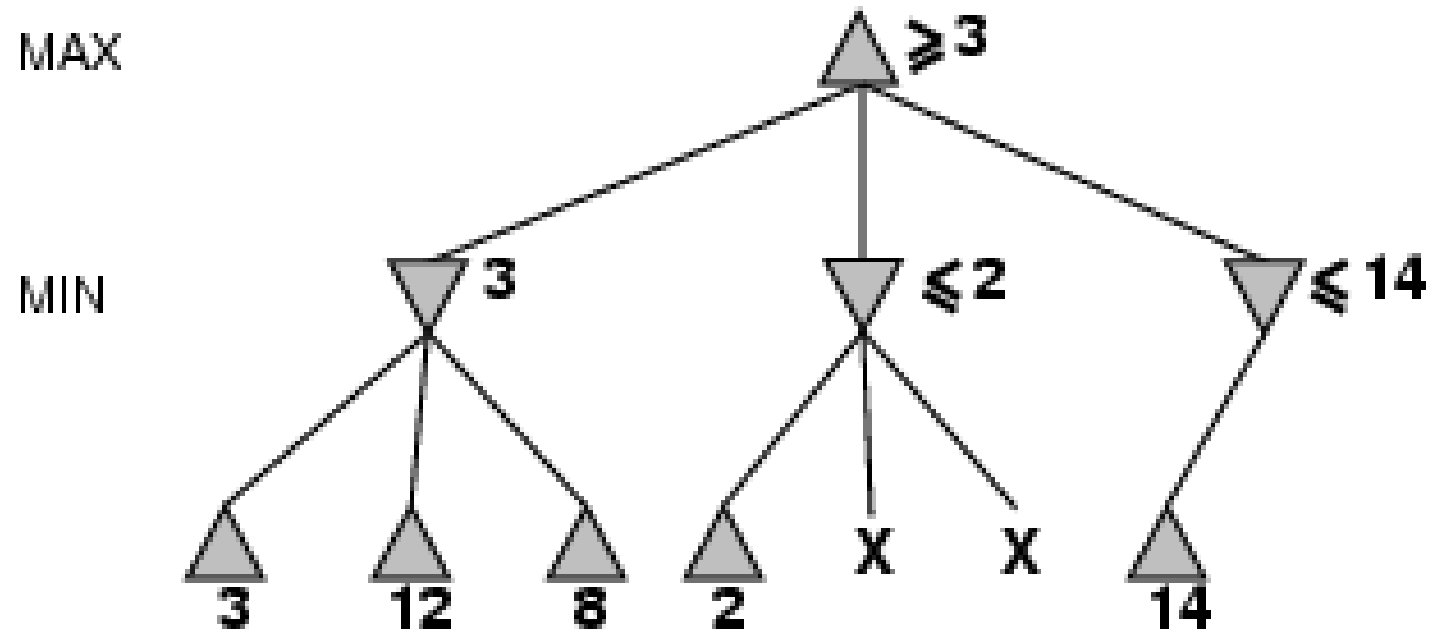
# $\alpha - \beta$ 刈込みの例



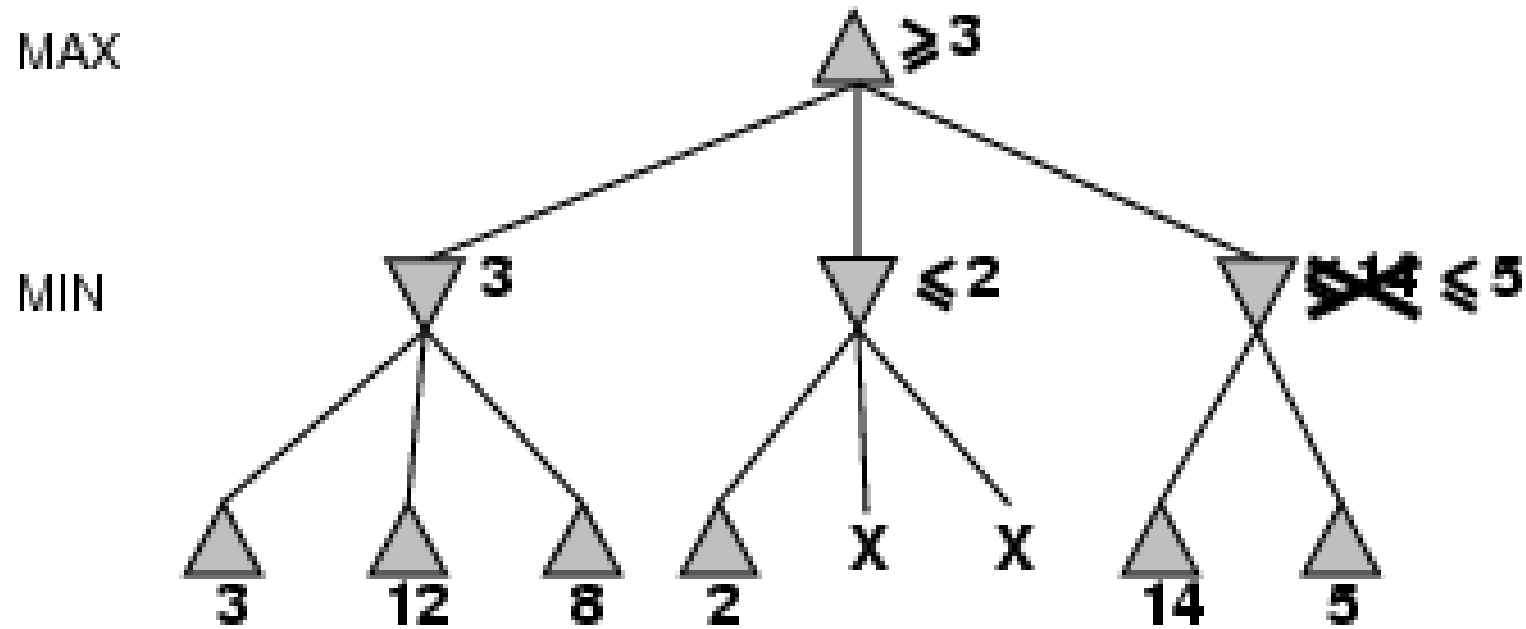
# $\alpha - \beta$ 刈込み



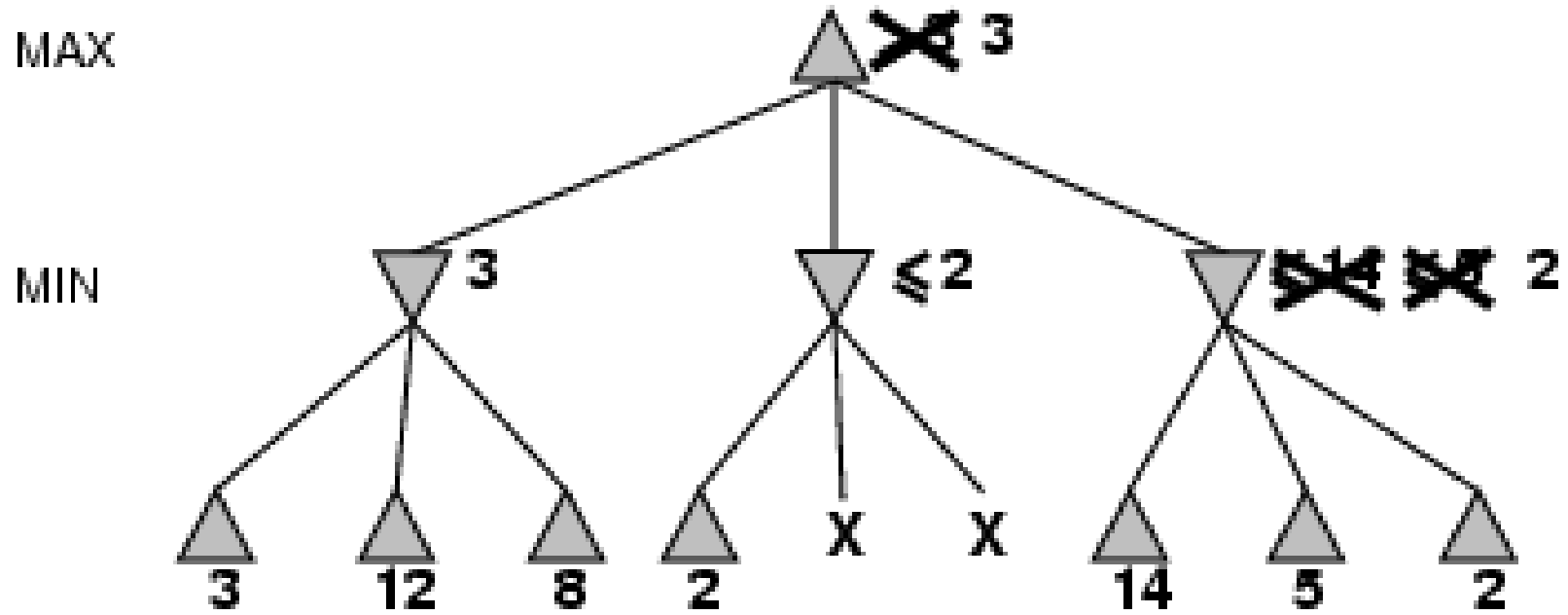
# $\alpha - \beta$ 刈込み



# $\alpha$ - $\beta$ 刈込み



# $\alpha$ - $\beta$ 刈込み

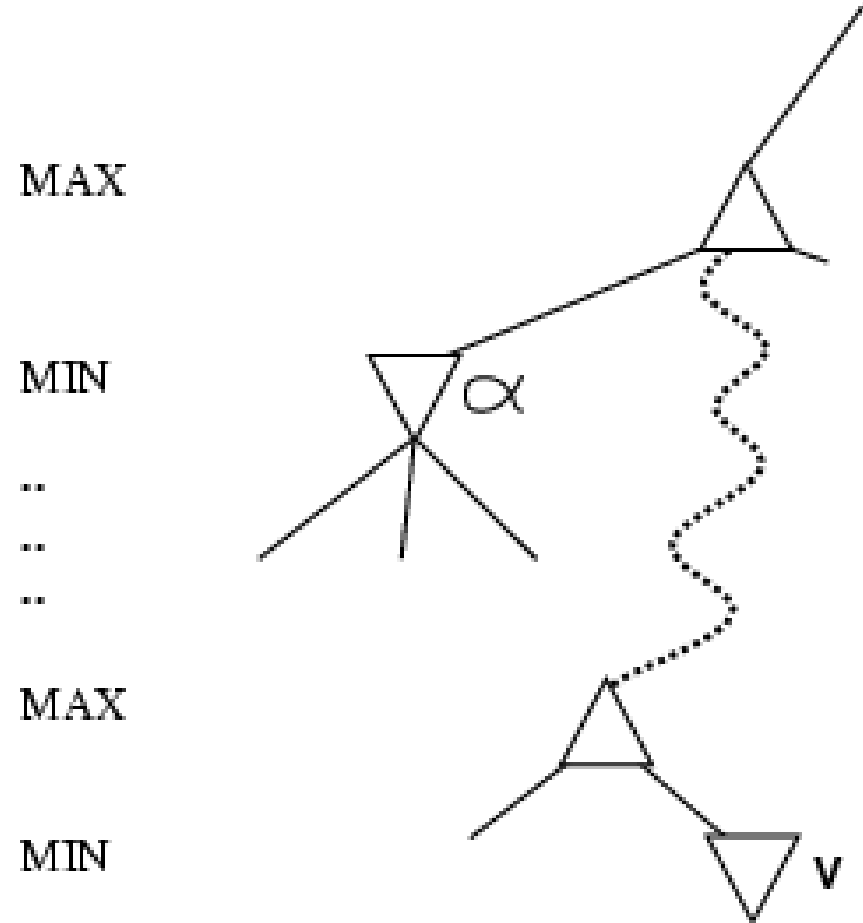


# $\alpha - \beta$ 刈込みの性質

- 刈込みは最終結果に影響しない
- 変な順番で探索すると、まったく刈り込むことができない。うまい順番でやるとかなり効果が上がる
- “完璧な順番付け”では時間の複雑性は  $= O(b^{m/2})$   
→ 深さ探索の2倍
- $\alpha - \beta$  刈込みはどの計算が解を求めるのに関係しているのかを推論する例となっている

# なぜ $\alpha$ - $\beta$ 刈込みと呼ばれるか

- $\alpha$  は、 $max$ の経路に沿った選択の点で、それまでに選択された中での最善の値
- $v$ が  $\alpha$  よりも悪ければ  $max$  はそれを避ける  
→ その枝を刈込む
- $\beta$  を同様に  $min$  に対して定義する



# $\alpha$ - $\beta$ 刈込みのアルゴリズム

**function** ALPHA-BETA-SEARCH(*state*) *returns an action*

**inputs:** *state*, current state in game

$v \leftarrow$  MAX-VALUE(*state*,  $-\infty$ ,  $+\infty$ )

**return** the *action* in SUCCESSORS(*state*) with value  $v$

---

**function** MAX-VALUE(*state*,  $\alpha$ ,  $\beta$ ) *returns a utility value*

**inputs:** *state*, current state in game

$\alpha$ , the value of the best alternative for MAX along the path to *state*

$\beta$ , the value of the best alternative for MIN along the path to *state*

**if** TERMINAL-TEST(*state*) **then return** UTILITY(*state*)

$v \leftarrow -\infty$

**for**  $a, s$  in SUCCESSORS(*state*) **do**

$v \leftarrow$  MAX( $v$ , MIN-VALUE( $s$ ,  $\alpha$ ,  $\beta$ ))

**if**  $v \geq \beta$  **then return**  $v$

$\alpha \leftarrow$  MAX( $\alpha$ ,  $v$ )

**return**  $v$

# $\alpha$ - $\beta$ 刈込みのアルゴリズム

```
function MIN-VALUE(state,  $\alpha$ ,  $\beta$ ) returns a utility value  
  inputs: state, current state in game  
            $\alpha$ , the value of the best alternative for MAX along the path to state  
            $\beta$ , the value of the best alternative for MIN along the path to state  
  
  if TERMINAL-TEST(state) then return UTILITY(state)  
   $v \leftarrow +\infty$   
  for  $a, s$  in SUCCESSORS(state) do  
     $v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(s, \alpha, \beta))$   
    if  $v \leq \alpha$  then return  $v$   
     $\beta \leftarrow \text{MIN}(\beta, v)$   
  return  $v$ 
```

# 資源制限

100 秒あったとし、1秒に $10^4$ ノード探索できる  
→ 一動作に $10^6$ ノード

標準的な探索:

- 打ち切りのテスト:

深さの限界 : 静止期 (状態は近い将来では変化しない) 探索を加える

- 評価関数

打ち切られた局面でどの程度有利であるのかを見積もる

# 評価関数

- チェスに対しては、特徴の線形の重み付けられた和

$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \dots + w_n f_n(s)$$

- 例:  $w_1 = 9$  で

$f_1(s) = (\text{number of white queens}) - (\text{number of black queens}), \text{ etc.}$

# 打切り探索

*MinimaxCutoff* は *MinimaxValue* と同じ

1. *Terminal* は *Cutoff* で置き換えられる
2. *Utility* は *Eval* で置き換えられる

実際の場面で機能するか

$$b^m = 10^6, b=35 \rightarrow m=4$$

4手先を読むチェスのプレイヤーには望みがない

- 4手先  $\approx$  素人
- 8手先  $\approx$  典型的なPC, 人間界でのマスター
- 12手先  $\approx$  Deep Blue, Kasparov

# 実用レベルでの決定的ゲーム

- Checkers: Chinook ended 40-year-reign of human world champion Marion Tinsley in 1994. Used a precomputed endgame database defining perfect play for all positions involving 8 or fewer pieces on the board, a total of 444 billion positions.  
»
- Chess: Deep Blue defeated human world champion Garry Kasparov in a six-game match in 1997. Deep Blue searches 200 million positions per second, uses very sophisticated evaluation, and undisclosed methods for extending some lines of search up to 40 ply.
- Othello: human champions refuse to compete against computers, who are too good.
- Go: human champions refuse to compete against computers, who are too bad. In go,  $b > 300$ , so most programs use pattern knowledge bases to suggest plausible moves.

# まとめ

- ゲームは対象として面白い
- AIについてのいくつかの重要な点を示した
- 完全ではありえない → 近似にならざるを得ない
- 何を考えるべきかを考えるための良い概念となる