132

# 情報認識論
# (第6回 ゲーム木探索 II)

九州大学大学院システム情報科学研究院
知能システム学部門
横尾 真
E-mail: yokoo@is.kyushu-u.ac.jp
http://lang.is.kyushu-u.ac.jp/~yokoo/

133

## 出席カードへの回答(4/28)

Q: じゃんけんゲームの均衡の求め方がよく分からない．そもそもゼロサムゲーム?

A: 対称なゼロサムゲームなら，混合戦略も許せば，均衡での期待利得は0のはず．また，均衡で，グー，チョキ，パーのどれかが他より有利なら，その手の確率を増やした方が期待利得が良くなるので，どれも期待利得は同じで0のはず --- 厳密には，ナッシュ均衡で使われない手もあるが，じゃんけんの場合はすべて使われている

実際は微妙にゼロサムゲームと違う(あと三歩以内で勝てるなら何で勝ってもよい)

Q: 宝石の分配で，同点は可決にしたらどうなる?

A: 自分で考えましょう．多分，結果はそんなに変わらない (DとEが逆転?)．

Q: 繰り返し囚人のジレンマで，どうして裏切り続けるのが "合理的"? 効用は減少してしまう．

A: 裏切り続けるのが反復支配戦略，反復支配戦略という "合理的"な戦略を取るプレイヤがパレート効率的な状態に到達できないから気持ち悪い/paradoxical.

## 具体的なアルゴリズム

- 関数 Vmax(n, α, β)を定義する．nはノード，α，βはそれぞれ下界値，上界値，この関数はノードnの評価値を返す．
- ルートのMAXノードnに関して，Vmax(n, －∞, ＋∞) を実行するとnの評価値が得られる．

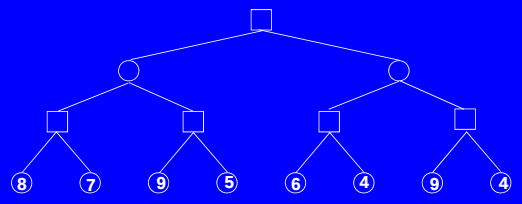
## 関数の (再帰的な) 定義

Vmax(n, α, β)
1. If n が端点 then return 静的評価関数の値
else set $n_k$ for each child $n_1$, $n_2$, ..., $n_b$
2. set α=max(α, Vmin($n_k$, α, β))
3. If α≧β, return β
4. If k=b, return α, otherwise, goto step 2.

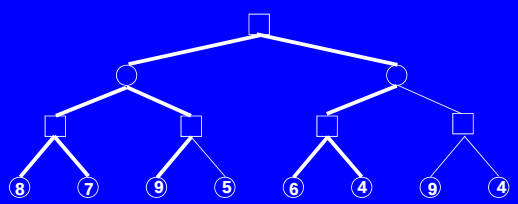
Vmin(n, α, β)
1. If n が端点 then return 静的評価関数の値
else set $n_k$ for each child $n_1$, $n_2$, ..., $n_b$
2. set β=min(β, Vmax($n_k$, α, β))
3. If β≦α, return α
4. If k=b, return β, otherwise, goto step 2.

## 例題: アルファ・ベータ探索

- アルファ・ベータ探索で探索されるノードはどれか?

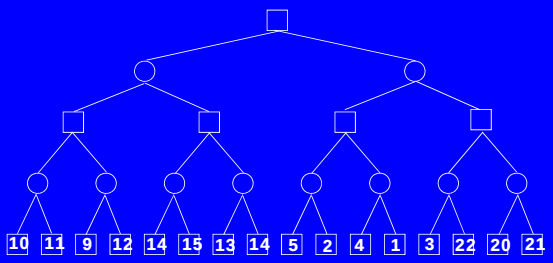



## 答え

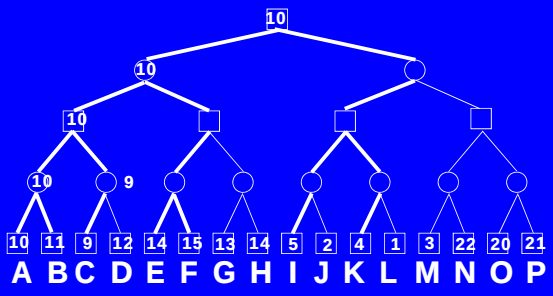


## 演習：アルファ・ベータ探索

- アルファ・ベータ探索で探索されるノードはどれか?




## 答え






## アルファ・ベータ探索の効果

- ノードを展開する順序に依存する
- MAXノードに関しては，なるべく大きな評価値となる子ノード，MINノードに関しては，なるべく小さな評価値となる子ノードから展開する方が良い
- 運が悪いと展開するノード数はミニマックス探索と同じ（ぜんぜん枝刈りができない）

## アルファ・ベータ探索の効果

- 運が良ければ，深さd，分岐bとして，探索される端点の個数は
  - MIN-MAX: $b^d$
  - アルファ・ベータ: $2b^{d/2}-1$
    - 祖先のMINノードが，すべてその親のMAXノードの最初の子供であるノードは必ず展開される（αが－∞）．
      - 数は $b^{d/2}$
    - 祖先のMAXノードが，すべてその親のMINノードの最初の子供であるノードは必ず展開される（βが＋∞）
      - 数は $b^{d/2}$
    - 一番左は二重に数えているので，合計 $2b^{d/2}-1$



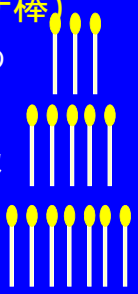
## アルファ・ベータ探索の効果

- 運が良ければ，深さd，分岐bとして，探索される端点の個数は
  - MIN-MAX: $b^d$
  - アルファ・ベータ: $2b^{d/2}-1$
- 効果は莫大であることに注意
  - b=2として，
    - $2^2=4$
    - $2^4=16$
    - $2^8=256$
    - $2^{16}=65536$
    - $2^{32}=4.29\times10^9$



## ゲームの例：ニム(マッチ棒)

- マッチ棒で，3本，5本，7本の三つの山を作る
- 交互にマッチ棒を取っていく
- 一つの山を選んで，好きな数だけ取る（全部取っても良い）
- 最後の一本を取ったほうが負け
- 前のコインバージョンと違って，山が分割されることはない



144

## 練習問題

- ニムの評価関数を作ってみよう
  - 自分の手番に関して評価関数を定義
  - 分からない状態は適当で良い
- その評価関数を使って，以下の状態でのMAXプレイヤの最適な手を求めよう (二手先を読む)
- 先読みの深さを変えると最適な手はどう変わるか？

145

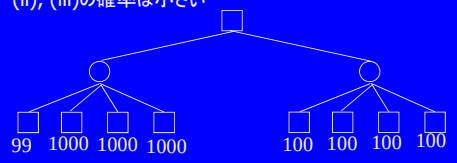
## 先読みの効果

- 基本的には，深く読めば読むほど強い
  - 終盤の方が静的評価関数の値が信用できる
  - そうでない場合は，先読みの効果は必ずしも自明ではない
  - 静的評価関数の値が，ノイズを含む値だとすると，MIN-MAX法で集計した値はノイズを増幅している可能性がある

146

## 評価関数がエラーを含む場合

- 先読み2で，MIN-MAXは右を選ぶ
- 本当に右が良いか?
- 静的評価関数の値が確率1/3で正しく，確率1/3で±10とする
- 本当に右が良いのは，(i) 左が過大評価の場合(1/3), (ii) 左が正しく，右で過大評価が一つもない場合，(iii)全部が過小評価の場合
- (ii), (iii)の確率は小さい




147

## 水平線効果

- 先読みの深さが一定だと，将来の損失が明らかな場合に，本質的でない先延ばしの手を選んでしまう可能性がある
  - ほぼ負けが決定の状態で，無意味な王手を繰り返す
  - 頭を砂に埋めるダチョウみたいなもの

148

## (とりあえずの)まとめ

- 二人，完全情報，決定的ゲームはゲームの木で記述される
- 原理的には先手必勝／後手必勝／引き分け
  - ゲーム木を完全に展開すれば分かる
- 完全に展開できない場合は，静的評価関数を用いて，一定の先読みでMIN-MAX法を用いる

149

## ゲームプログラムの歴史(1)

- ゲームをプレイするプログラムの作成は，人工知能のfruit fly (ショウジョウバエ) と呼ばれていた.
- 1950年 Shannon，Turingがコンピュータチェスの可能性を示す論文
- 1950年代 初めてチェスを指すプログラムが作成される
- 1950年代 Herbert Simonが10年で世界チャンピオンに勝つと予想

150

## ゲームプログラムの歴史（2）

- 1960年代　哲学者のヒューバート・ドレイファスがチェスのプログラムは「永久に世界チャンピオン」に勝てないと予想
- 1960年代　Arthur Samuelのチェッカープログラム
  - 静的評価関数の学習（強化学習の一種）
  - 強い!

151

## ゲームプログラムの歴史（3）

- 1980年代
  - チェス専用コンピュータ
  - スーパーコンピュータ
- Deep Thought　CMU
  1秒間に70万局面
  人間のベスト100に到達

152

## Deep Blue

- IBMが1989年から開発を開始
- 1990年世界チャンピオンのカスパロフと対戦　2戦2敗
- 1996年再度カスパロフと対戦
  6戦　1勝3敗2分け
- 1997年　ニューヨーク
  6戦　2勝1敗3引き分け
- 1秒間に2億個の状態を評価
  - 3分で14手先読み

  スーパーコンピュータ
  ＋チェス専用の論理回路512台

153

## 将棋

- 難しさの要因
  - 持ち駒制度
    - 平均分岐数の大きさ
  - 勝負の長さ
  - 静的評価関数のむずかしさ
    - 小駒が多い

154

## 将棋（続き）

- 平均分岐数の多さから，α-β探索を使うことは困難で，従来は，あらかじめ有望な手を絞り込む手法が中心
  - 最近の強いソフト（ボナンザ）は，絞込みをあまり行わないことが特徴
  - 評価関数の自動学習を頑張っているらしい

155

## 二人ゲーム以外への応用

- 一人での意思決定だが，偶然の要素がある場合：
  - 自然というもう一人のプレイヤがいると考える
  - 自然がどう行動しても（自分に取って最悪の手を打っても），自分が勝てるような手を選ぶようにする

156

## 偽金貨を見つける

- 12個の見た目は全く同じ金貨がある
- 一つだけ偽の金貨があり，本物よりわずかに重い
- 天秤秤を三回だけ使って，偽金貨を見つけられるか?

157

## 偽金貨を見つける（続き）

- 例えば，金貨を一つずつ選んで秤にのせた場合，起こりうる可能性は三通り
  - つりあう
  - 左が重い
  - 右が重い
- どれが起こるかは分からない（自然の選択）
- どれが起こっても大丈夫なように計画を作っておく

158

## 偽金貨を見つける （答え）

- まず4つずつ比べる
- つりあわなかったら重かった方，つりあったら残りの4個の中に偽金貨がある
- うたがわしい4個から，2個ずつ比べる
- 重いほうの2個のどちらかが偽金貨
- 2個を比べて重いほうが偽金貨
- 他の解も多数存在

159

## 偽金貨を見つける（続き）

- 一つだけ偽金貨があることは分かっているが，それが本物より重いか軽いか分からない場合
  - うまく工夫するとやはり3回で十分
  - それが重いか軽いかも分かる
  - ヒント：最初は4つずつ比較
  - 二回目がポイント，うまくまぜる

160

## 偽金貨を見つける（続き）

- 1, 2, 3, …, 11, 12の金貨がある
- (1, 2, 3, 4) と (5, 6, 7, 8)を比較
- If (1, 2, 3, 4) = (5, 6, 7, 8)：偽は9,…, 12の中
  - (1, 9) と (10, 11)を比較
  - If (1, 9) = (10, 11), 12が偽
    - (1)と(12)を比較，(1)<(12)なら重い，(1)>(12)なら軽い
  - If (1, 9) < (10, 11)
    - (10)と(11)を比較，(10)=(11)なら9が軽い，(10)<(11)なら11が重い，(10)>(11)なら10が重い
  - If (1, 9) > (10, 11)
    - (10)と(11)を比較，(10)=(11)なら9が重い，(10)<(11)なら10が軽い，(10)>(11)なら11が軽い

161

## 偽金貨を見つける (続き)

- If (1, 2, 3, 4) > (5, 6, 7, 8)：9,…,12は本物
  - (1, 2, 5)と(3, 6, 12)を比較
  - If (1, 2, 5) = (3, 6, 12) --- 4, 7, 8のどれか
    - (7)と(8)を比較，(7)=(8)なら4が偽で重い，(7)<(8)なら7が軽い，(7)>(8)なら8が軽い
  - If (1, 2, 5) < (3, 6, 12) --- 5が軽いか3が重い
    - (3)と(12)を比較，(3)=(12)なら5が軽い，(3)>(12)なら3が重い
  - If (1, 2, 5) > (3, 6, 12) --- 1, 2が重いか6が軽い
    - (1)と(2)を比較，(1)=(2)なら6が軽い，(1)<(2)なら2が重い，(1)>(2)なら1が重い
- If (1, 2, 3, 4) < (5, 6, 7, 8) --- 省略

162

## 偽金貨を見つける (続き)

- コインの個数nに対して，3回で偽金貨を発見できる最大のnはいくつか?
  - 重いことが分かっている場合
  - 重いか軽いか分からない場合
  - 重いか軽いかも判定しないといけない場合
- nと，最小の秤の使用回数の関係は?