ゲーム情報学特論　探索 第3回
MINMAX探索　αβ探索

北陸先端科学技術大学院大学
情報科学系
ゲーム情報学分野　准教授　池田　心

2018-05-01

前回のおさらい

- 一人ゲームでゴールまでの経路を見つける
- 「探索木」に対するさまざまな探索法
- それぞれ特徴があり，一長一短
- コスト関数（given）とヒューリスティック関数（ゴールへの距離を推定する，解く側が与えるヒント）

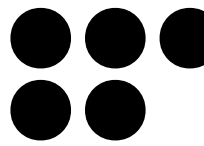
名称	コスト関数	ヒューリスティック関数	最適性	特徴
縦型探索				メモリ小，深いものが得意
横型探索			○	メモリ大，浅い正解を容易に発見
反復深化			○	メモリ小．重複あり
ダイクストラ法	○		○	横型に似る，効率の悪さもある
ヒューリスティック関数の最良優先探索		○		h'の設計により性能が大きく変動
Aアルゴリズム	○	○	※	同上．h'=0のとき分岐限定法． ※ h’(n) ≦ h(n) のとき最適保証

今日のお話：二人ゲーム

- ゲームと一口に言っても，二人/三人/四人以上，離散/連続，同時/交互/非同期，完全情報/不完全情報，ゼロサム/非ゼロサム等ある
- 今回は**二人**・交互・完全情報・ゼロサムを考える
- 囲碁，将棋，チェス，オセロなど
- 一人ゲーム（●○●○●○，TSP，15パズル，ルービックキューブ等）では，全ての着手（edge）を自分が決められた
- 敵プレイヤーは概ねこちらの邪魔をするように（一番やってほしくないように）行動する

やってみよう： 石とりゲーム

- ２人で対戦
- （1） 8個の石を置き，交互に1～3個とれる．最後に石をとった方が勝ち
- （2）2個，3個，4個の石のグループ（山）を作り，交互に，一つの山からだけ好きなだけとれる．最後に石をとった方が勝ち． ●● ／ ●●● ／ ●●●●
- （3）平面的に石を置き，交互に上下左右連続した石をとれる．最後にとった方が勝ち．



※別添資料も参照（中学生向け）

読み切りと評価関数

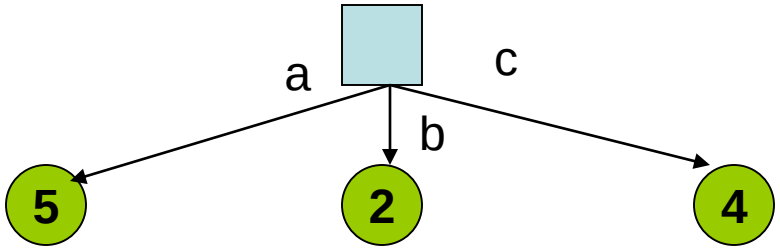
- 前述の石取りゲームくらいなら，勝ち負けがはっきりするまで読み切る（展開しきる）ことができる
- 将棋や囲碁では絶対無理

→ある程度の深さまで展開したら，その局面の良さを表す「**状態評価関数**（state evaluation function）」を使う

  - AlphaGoの論文ではこれを value network で表す

- オセロなら駒数・着手可能箇所数・カド数など
- 将棋なら駒得・利きの多さ・王の安全度など
- ○○なら××など（考えてみよう）
- 評価関数は本来性能に直結する重要な研究対象だが，今日の授業ではこれは与えられているとする

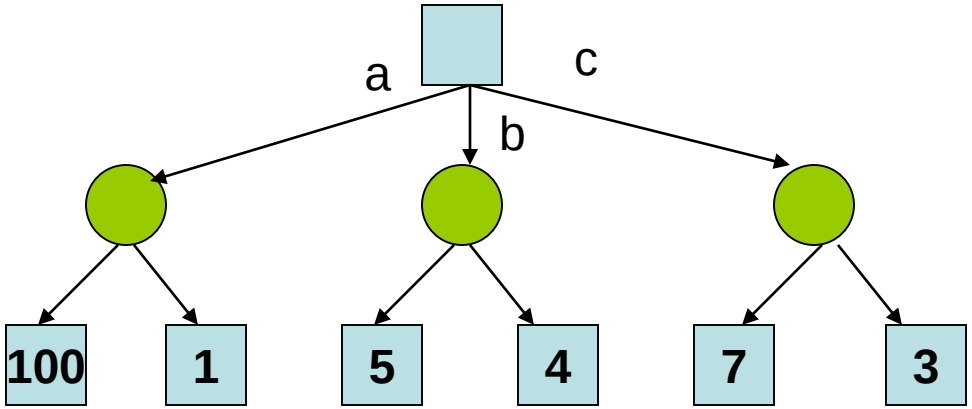
例題:1手の場合



自分の手番

相手からもらえる金額

例題:2手の場合
どの手を選びますか？

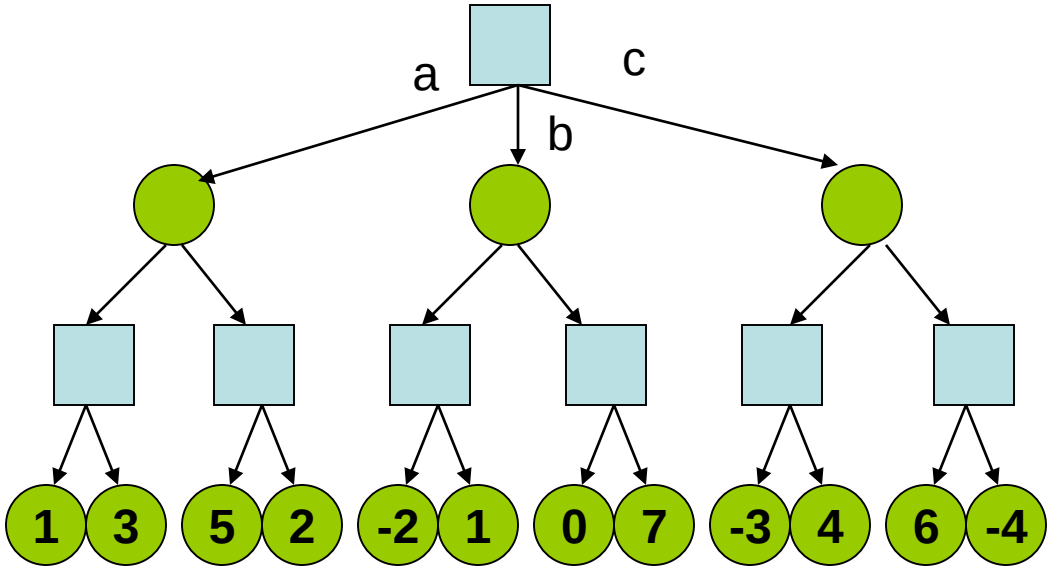


自分の手番

相手の手番

相手からもらえる金額

例題:3手の場合



自分の手番

相手の手番

自分の手番

MINMAXとは

- 自分の手番→ 評価値最大の手を選ぶ
- 相手の手番→（自分にとって）評価値最小の手が選ばれる
- 評価値の定まった葉ノードから上向きに評価値が徐々に定まってくる
- MINMAX法の考え方は，ほとんどの二人零和完全情報ゲーム（チェス、将棋、オセロ等）の探索で使われる

- 自分の手番のノードを MAX node
- 相手の手番のノードを MIN node と呼ぶことにする

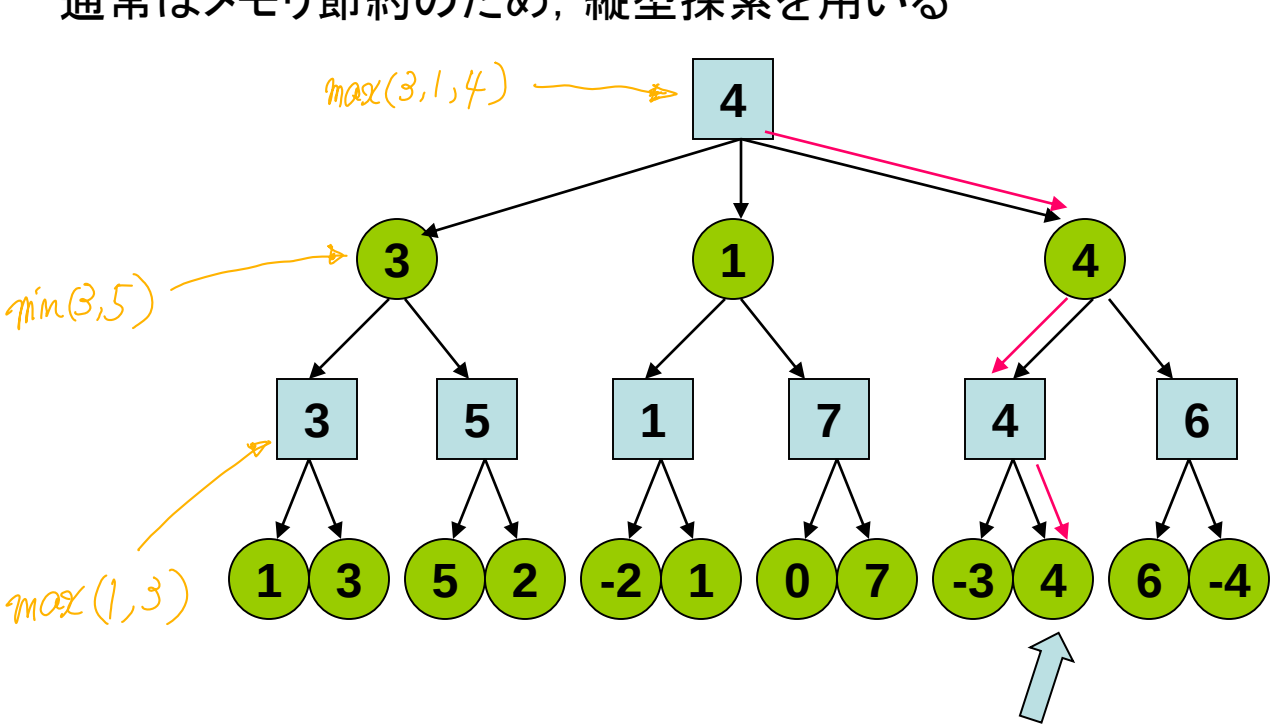
MAX node

MIN node

★重要

MINMAX tree

直感的には葉ノードが深さ3なら次は深さ2をMAXで，
次は深さ1をMINで・・・と階層的に計算したいところだが
通常はメモリ節約のため，縦型探索を用いる



MAX node

MIN node

→ : 最善手順　best route, principal variation
お互いに最善を尽くした結果選ばれる手順

MINMAX法　特徴

- （探索をした範囲内で，）（状態評価関数の評価値を元に，） お互いが最善を尽くした結果を求める
- 相手が十分賢いことを想定．そうでない場合は「もっとがめつい手」があるかもしれない
  - 相手モデルに基づく戦略選択
  - 現実世界では重要．たぶん飯田先生が講義する
  - P7の例で言うと aを選ぶなど
- 分岐因子数を b ,探索深さを d とすると b^d のノードを評価する　⇒ 深さに従い指数的に増大
- 例えば4×4オセロだと 12!（約5億）

αβ法

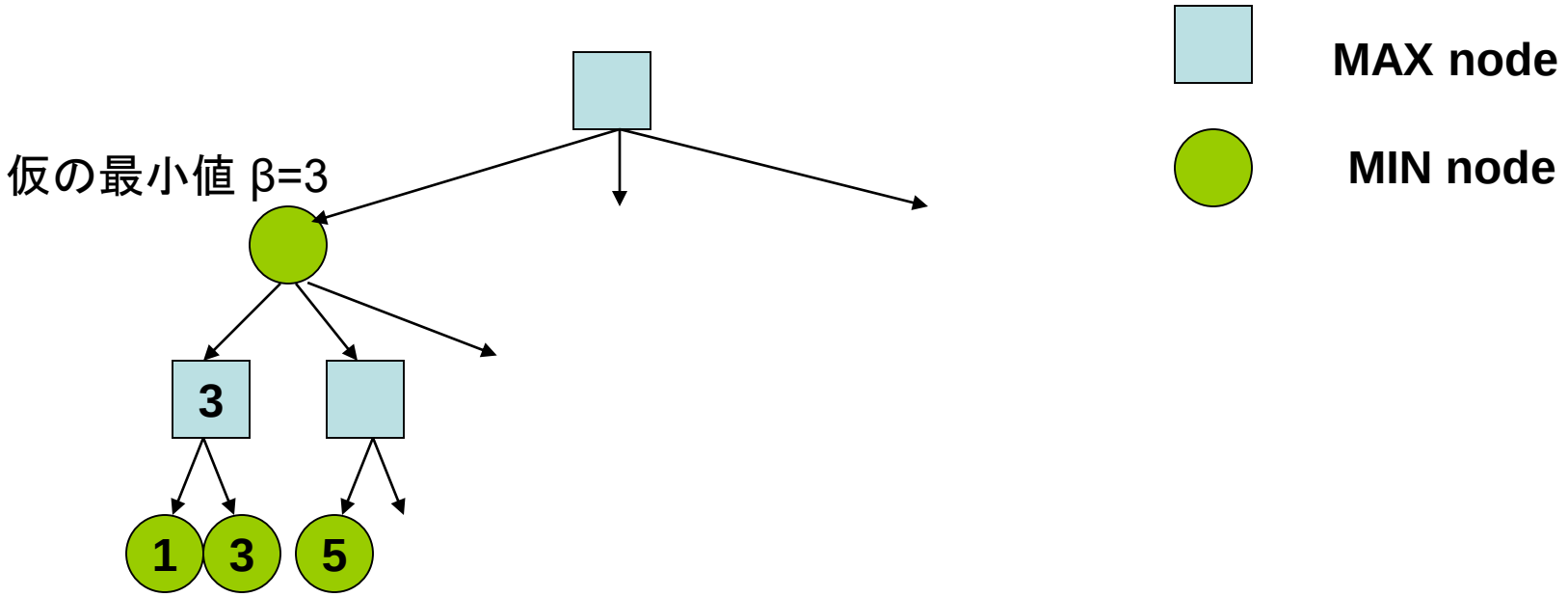
- 葉ノードの状態評価関数値を計算するのは高コストなことが多く，回数を減らしたい
- αβ法（αβ pruning）によって無駄な探索を省略

1. 基本は縦型探索のminmax探索
2. MAXnodeに仮の最大値α．MINnodeに仮の最小値βを保持する
3. MINnodeでは，β（ある子ノードの評価値）が親のα以下になったら，以下の子ノードは読まない
4. MAXnodeでは，α（ある子ノードの評価値）が親のβ以上になったら，以下の子ノードは読まない

言葉による定義だけでは意味が分からんが・・・

αβ法（αβpruning）実例 β cut

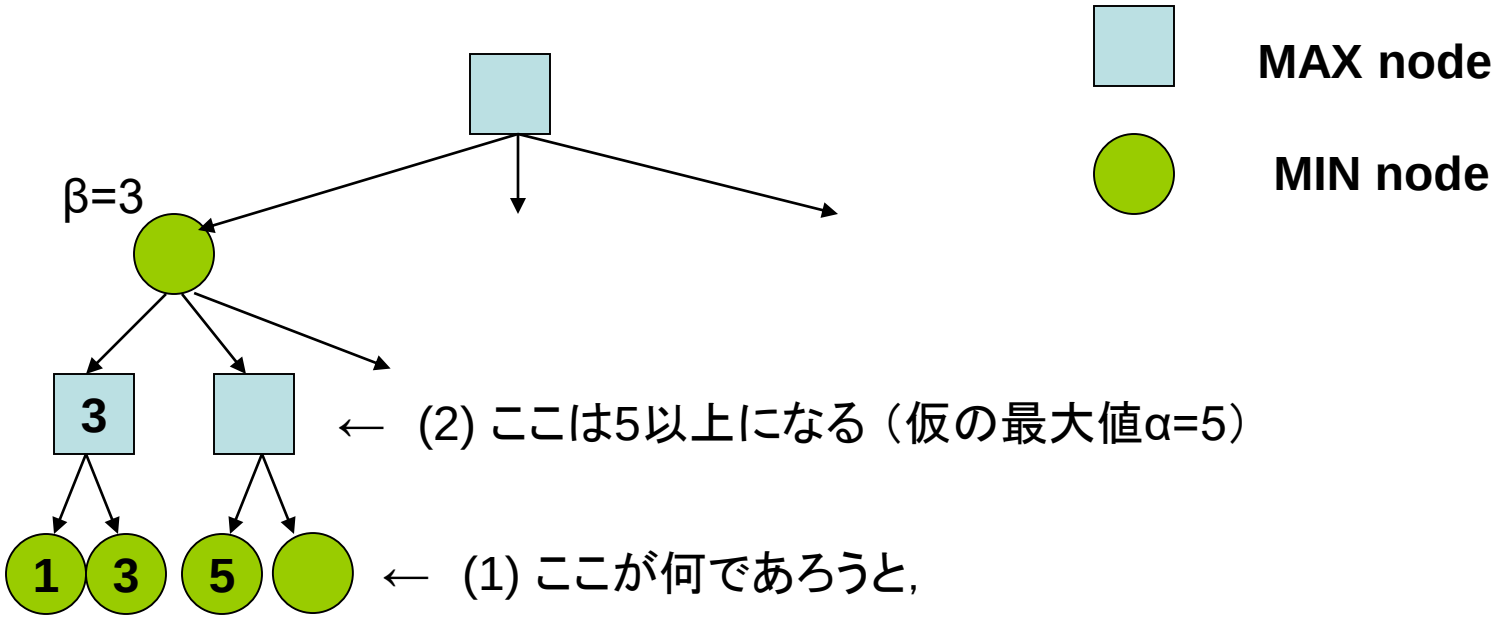
基本，縦型の順で読んでいく



ここまで探索したあと考えることは・・・

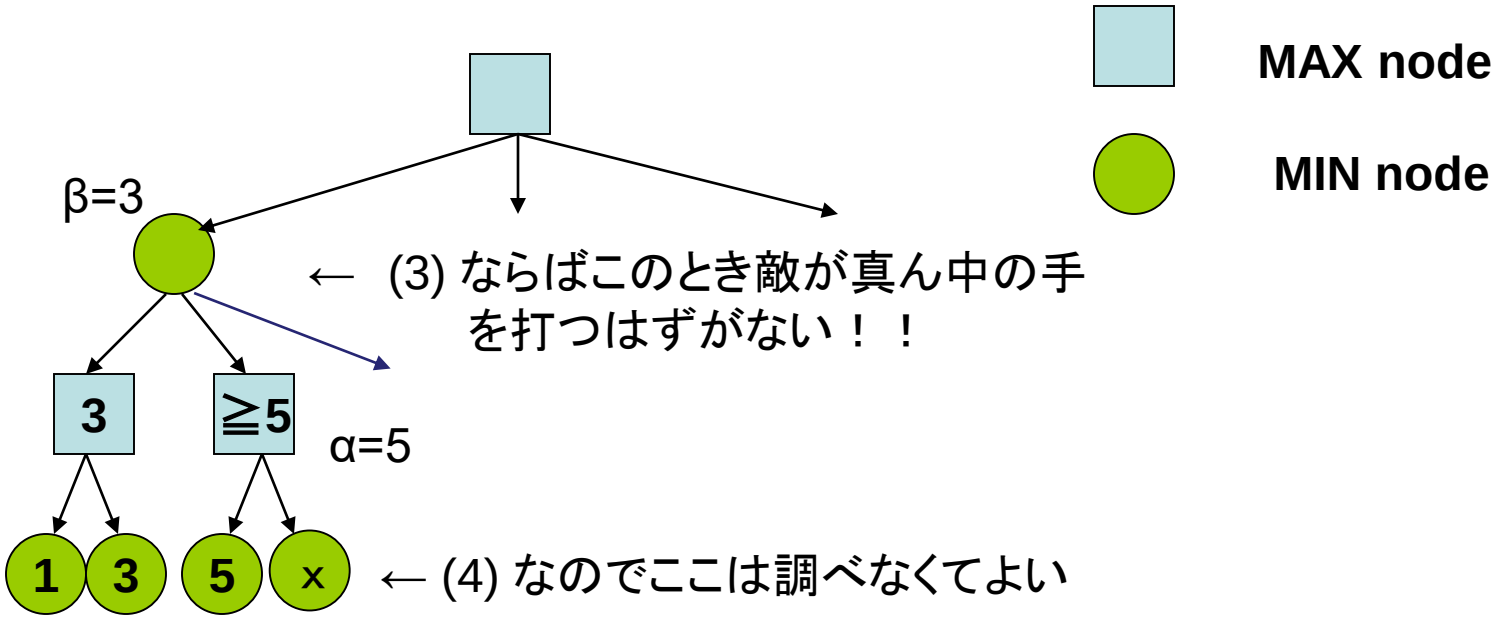
αβ法（αβpruning）実例 β cut

基本，縦型の順で読んでいく



αβ法（αβpruning）実例 β cut

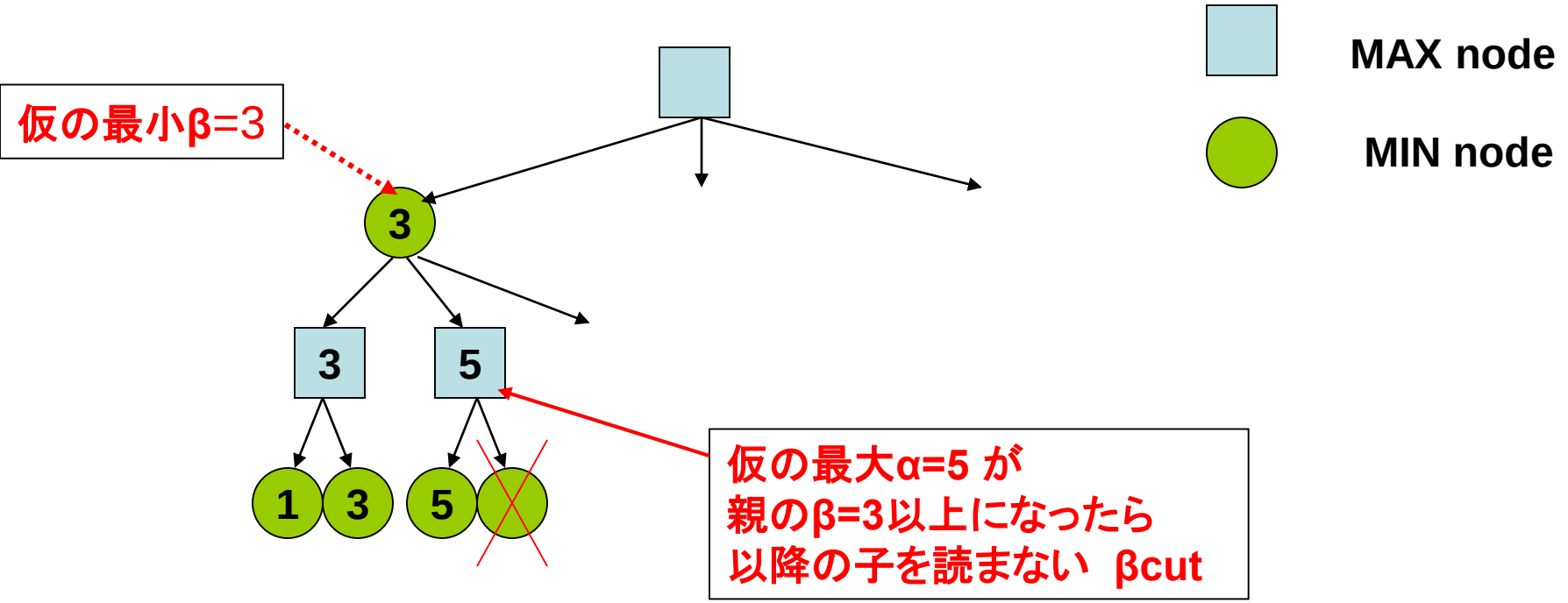
基本，縦型の順で読んでいく



※ β=3はまだ未確定，
敵の右の手はまだ調べる

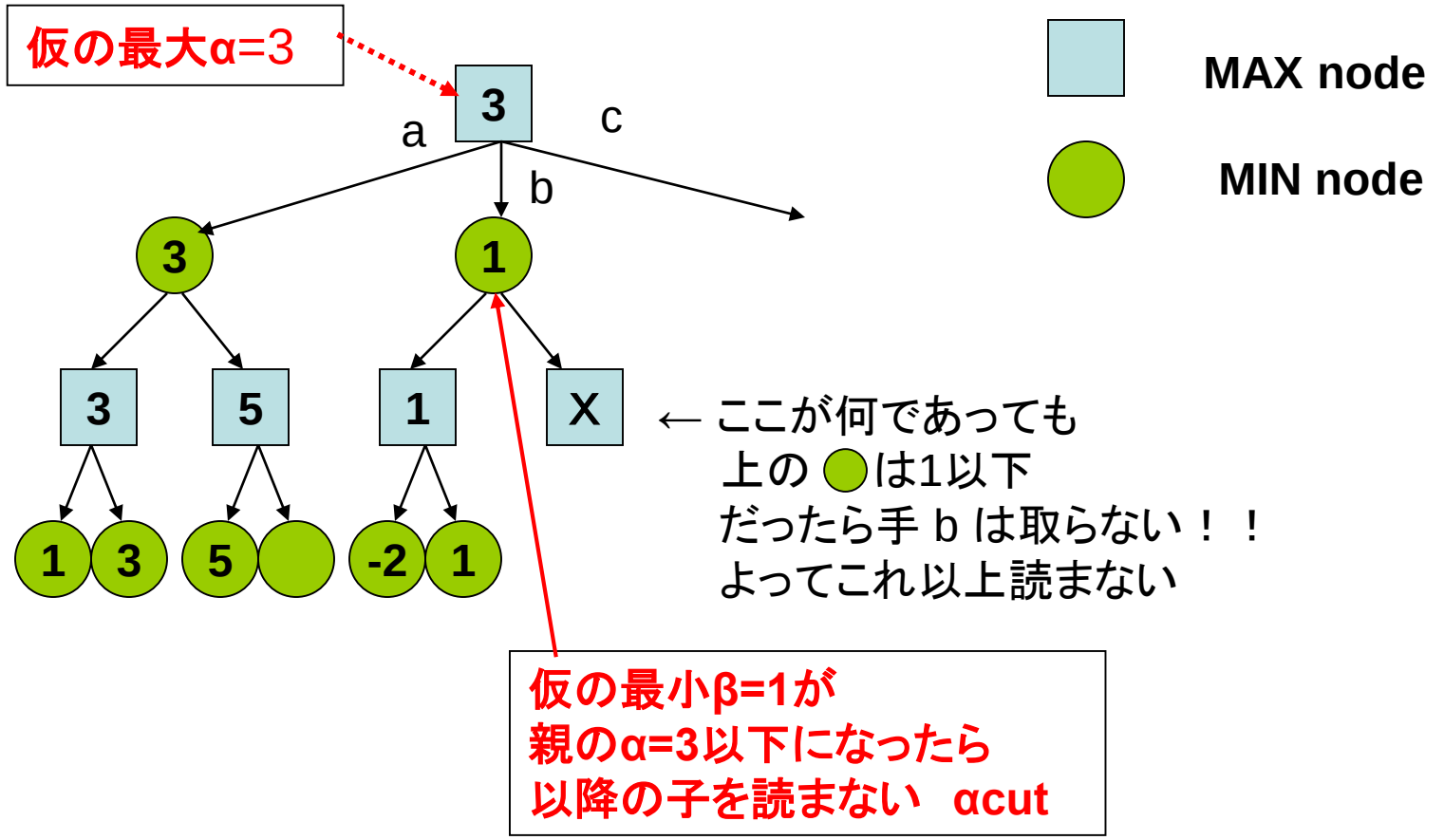
αβ法（αβpruning）実例　β cut

p12【 MAXnodeでは，α（ある子ノードの評価値）が
親のβ以上になったら，以下の子ノードは読まない】

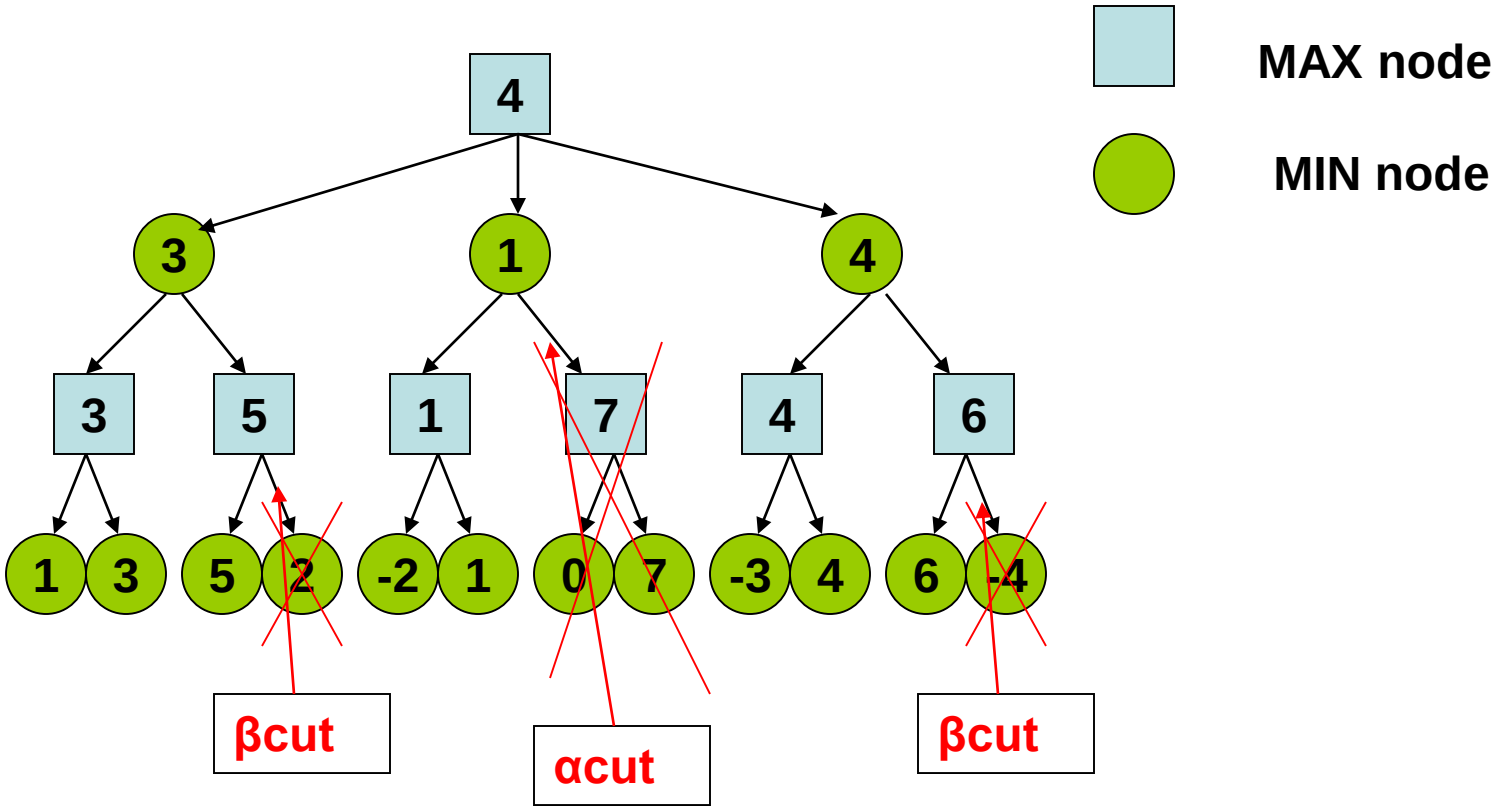


αβ法（αβpruning）実例　α cut

【MINnodeでは，β（ある子ノードの評価値）が
親のα以下になったら，以下の子ノードは読まない】



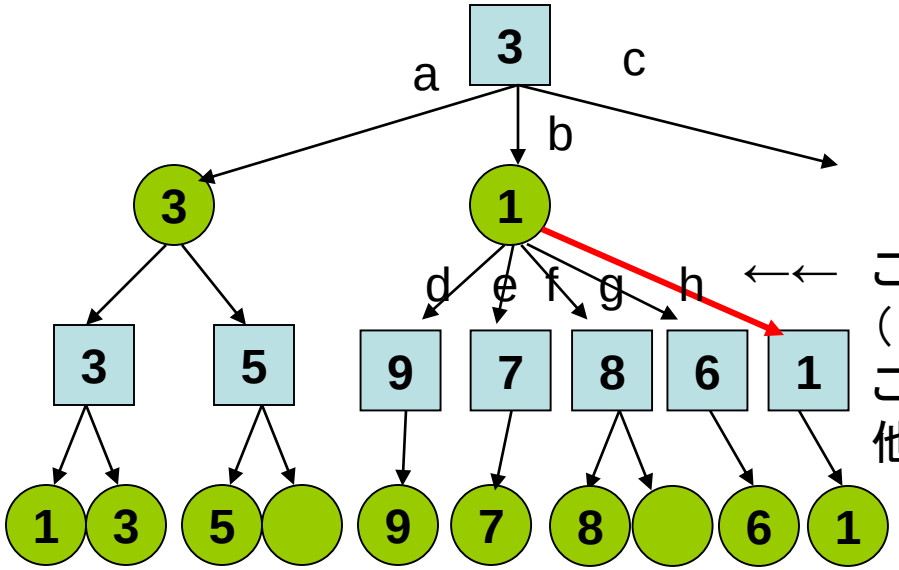
αβ法（αβpruning）実例



12回の評価　→ 8回の評価で済んだ
・・・・そんなもの？

αβ法の効率化

αβ法の効率を上げる為には，いい手を早めに読みたい



←← この手hは相手の良い手
（自分の都合の悪い手）
これが早めに見つかれば
他は読む必要がなかった！

良さそうな手（この場合a）を先に読んでおき，
他の手（この場合b）に対してはそれがダメな手
であると証明できるような手（この場合h）から読みたい

★重要

αβ法の効率化

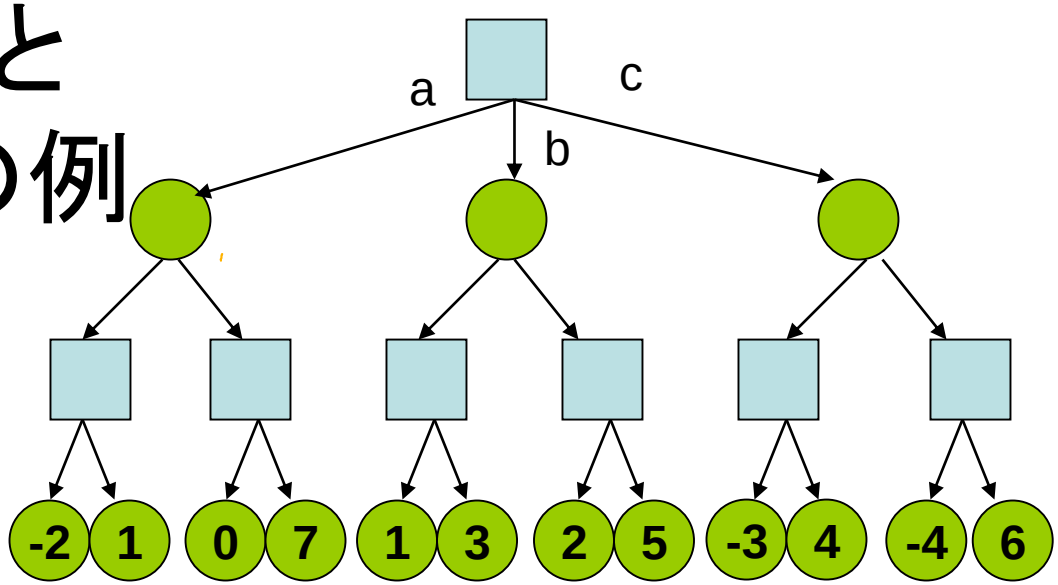
- αβ法の効率を上げる為には…
  ⇒ いい手を早めに読むとCutが起こりやすい
  ⇒ いかに「よさそう」な手を早めに読ませるか
  が重要なテーマになる

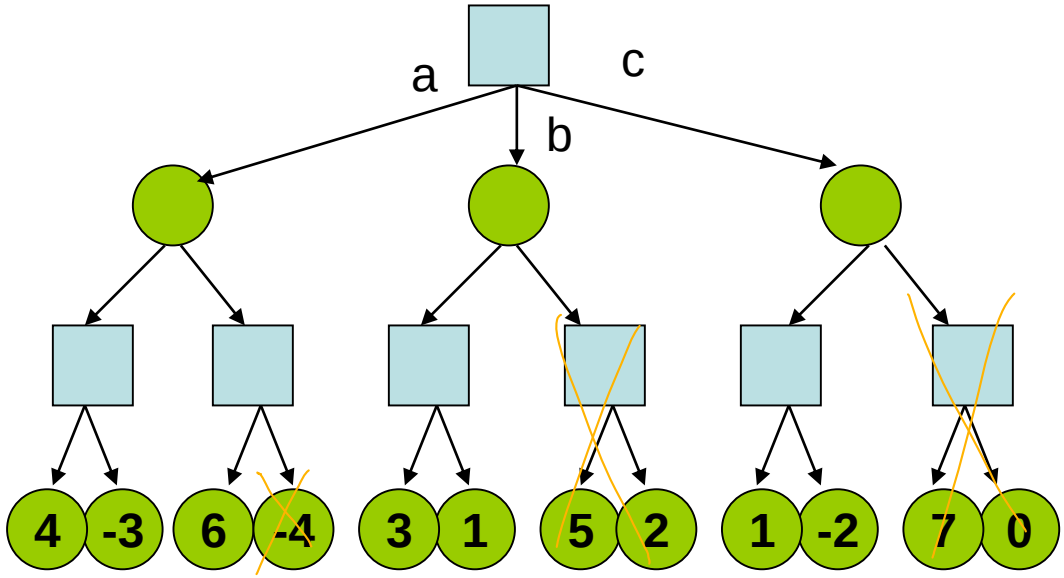
指し手の順序付け（Move ordering）が重要！

たいていの場合，Move ordering にも評価関数を使う
→ 微妙な違いだが，「**行動評価関数**」を使うことが多い．
- AlphaGoの論文では policy networkで表す

[Remi2007]Computing Elo Ratings of Move Patterns in the Game of Go
囲碁プログラム界の必読論文だったもの（フォルダに日本語訳あります）

最良ケースと最悪ケースの例



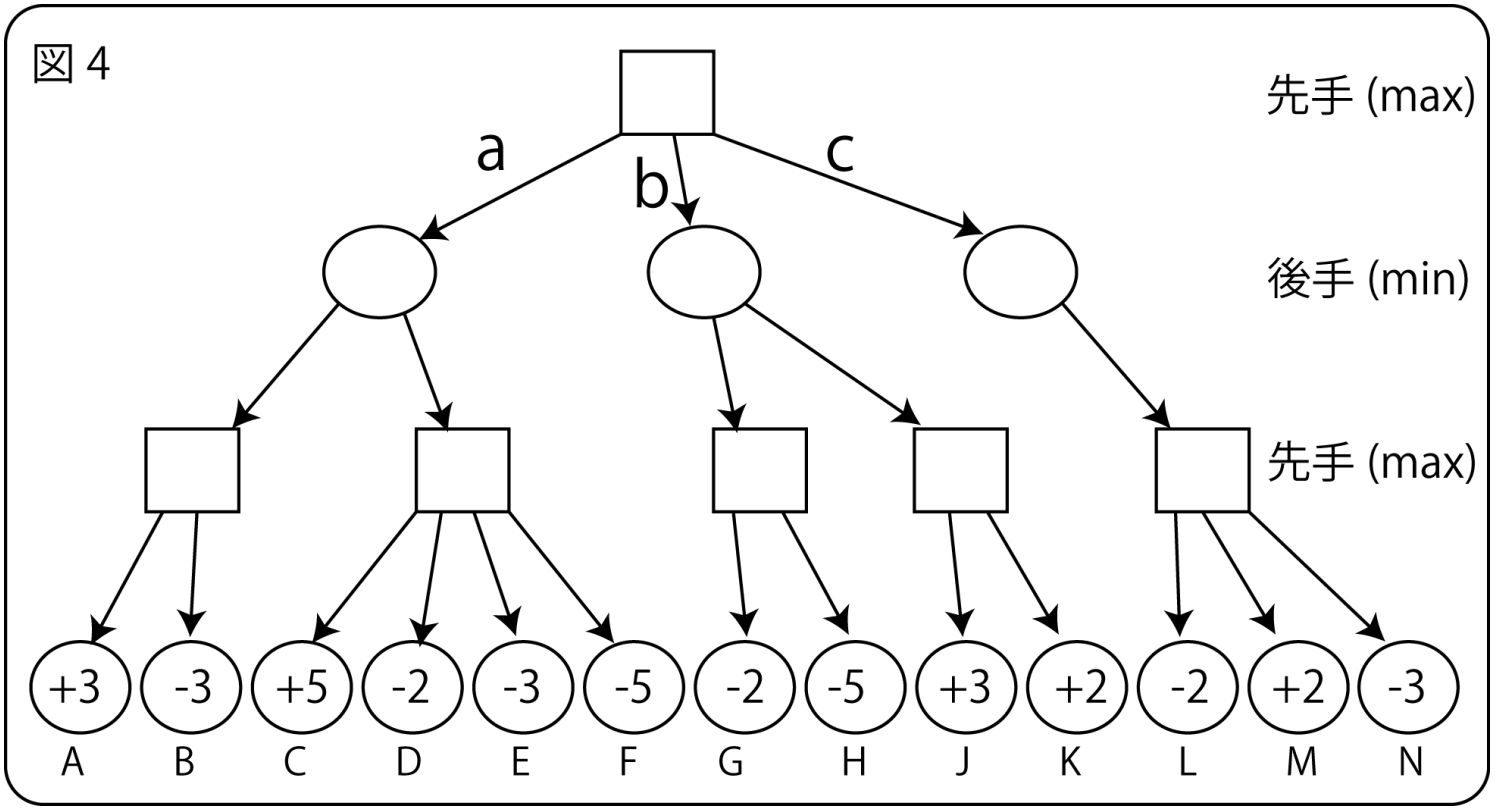


過去問(練習用)

4.

(1)先手側は初手でa b cのどの行動を選択すべきで,
利得はいくつになると予測されるか. **[7]**

(2) αβ法を使った場合, 調べなくてよいleaf nodeはどれか.
A~Nの記号で全て答えよ. **[7]**



今日のまとめ

- ２人ゲームの実践
- ２人ゲームの基本的な考え方　minmax
- 巧妙な探索省略法　αβ
- Move Ordering による効率化

第５回レポート（5/14〆）

以下の課題のうち２～３つを選んで，
A4 1~2枚にまとめて提出せよ．　　（４点）

【１】何か二人ゲームを取り上げ，状態評価関数として使えそうな項目を挙げよ（メモ：ぷよぷよ，五目ならべ，AoE，大貧民，全部見え麻雀，しりとり）

【２】手本となるプレイ記録（棋譜）が大量にある場合に，状態評価関数をそこからうまく調整する方法を考えよ

【３】状態評価関数では使えないような項目を行動評価関数では使える場合がある．例示せよ．

【４】深さ4の四分木にはいくつの評価すべきノードがあり，αβ探索の最良ケース（ノードがたまたま全て良い順に並んでる）ではいくつのノードを評価すればよいか．