# 第6講 探索手法（3）

- 問題の分解（分割）と探索
  - 問題の分解（分割）
  - AND/ORグラフ
- ゲーム探索
  - ゲーム木の探索
  - ミニマックス法
  - アルファベータ法

# 問題の置換と分解

- 複雑な（現実の）問題の解決
  - ひとつの状態空間で表現すると、大きな状態空間になってしまい、解を求める効率が低下してしまう
- 複雑な問題の解決
  - より簡単な別の問題に置き換える（問題の置換）
  - 問題をいくつかの単純な問題に分解する（問題の分解）
- 取り扱う問題を置換や分解できれば、問題解決は容易になる

# 旅行計画の例

- 旅行計画を立てる
  1. 資金を調達する
  2. 行き先を決める
  3. 日程を決める
- 資金を調達する
  – アルバイトする
  – 貯金をおろす
- アルバイトする
  1. 求人広告を調べる
  2. 応募する

－－－－－－－－－－－

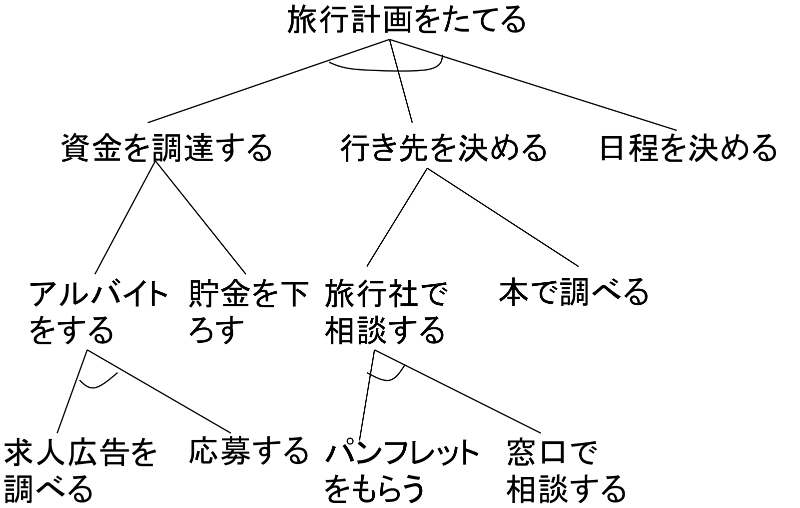
# AND分解とOR分解

- 問題分解の方法
  – AND分解
    - 問題Aを解くためには、副問題Xと副問題Yの両方を解かなければならない
    - 問題Aは副問題Xと副問題YにAND分解されるという
  – OR分解
    - 問題Aを解くためには、副問題Xか副問題Yのいずれかのひとつが解ければよい
    - 問題Aは副問題Xと副問題YにOR分解されるという
- 実際の問題解決
  – AND分解とOR分解を組み合わせながら、もとの問題よりも単純（解きやすい）問題に変換する

# AND/ORグラフ

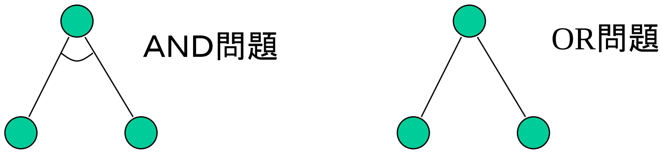
- 実際の問題解決
  - AND分解とOR分解を組み合わせながら、もとの問題よりも単純（解きやすい）問題に変換する
- AND/ORグラフ
  - 上記の分解による問題解決の形式を表現
  - ある問題を複数の副問題に分解された場合
    - 問題を親問題、副問題を子問題という
    - 解くべきもとの問題を源問題という
    - これ以上分解できない問題を素問題という

# 旅行計画のAND/ORグラフ



# AND/ORグラフの構成規則

1. AND/ORグラフは解くべき問題の集合とそれらの関係（AND分解とOR分解)から構成される
2. 源問題はAND/ORグラフの根であり、素問題はグラフの葉（終端）である
3. OR分解された副問題は、親問題に対応する子の問題になり、OR問題と呼ばれる
4. AND分解された副問題は、親問題に対応する子の問題になり、AND問題と呼ばれる




# 解グラフ

- 解グラフとは
  - AND/ORグラフの探索における解
  - AND/ORグラフに一部分
- 解グラフの性質
  - 初期ノードを含む
  - あるノードがAND分解している場合、その子ノードのすべてを含む
  - あるノードがOR分解している場合、どれか一つの子ノードを含む
  - すべてのノードは目標ノードである

# AND/ORグラフ探索

- アルゴリズム
  1. 出発ノードからなる未解決グラフをOPENリストに入れる。
  2. OPENリストから先頭の要素Nを取り出す。
  3. OPENリストが空であれば、失敗で終了。要素Nが解グラフであれば、成功で終了。
  4. 要素Nを展開し、未解決グラフの集合を得る。これらの未解決グラフを評価し、無解グラフ以外をOPENリストに入れる。
  5. ２へ戻る。

# ゲームのプログラム

- チェス，チェッカー，オセロ，五目並べ（連珠），将棋，囲碁
- コントラクト・ブリッジ，ポーカー，麻雀，バックギャモン

## ゲームの分類

- 人数（一人，二人，三人，四人，n人）
- 零和ー非零和
- 有限ー無限
- 完全情報ー不完全情報
- 確定ー不確定

## ゲームの必勝法

- 二人・零和・有限・完全情報・確定ゲームには必勝法が存在する

  先手必勝・後手必勝・引き分け
- 簡単なゲームは具体的な手順がわかる

  〇×（三目並べ）　引き分け

  五目並べ　先手必勝

  6x6　オセロ　後手必勝

## 必勝法の存在

- 将棋には必勝法が存在する！
  先手必勝？　引き分け？
- チェッカー，（8x8）オセロ，チェス，囲碁
- しかし（まだ）具体的な必勝手順は見つかっていない

## ゲームの場合の数（初手から勝負がつくまで全部を読むと．．）

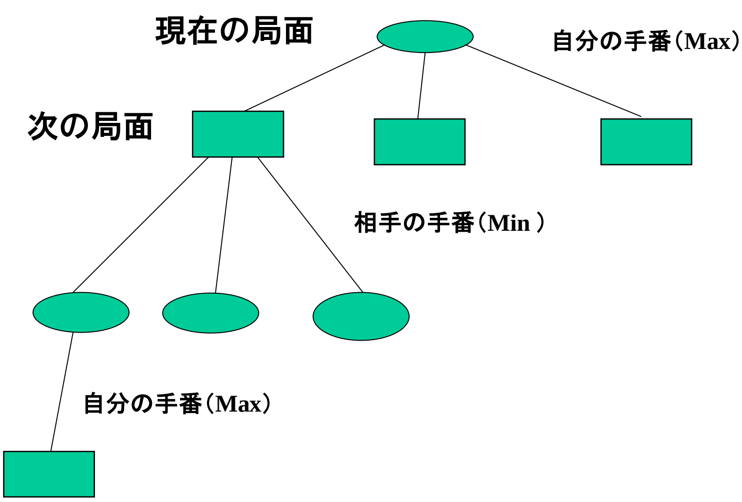
- チェッカー　　10の30乗
- オセロ　　10の60乗
- チェス　　10の120乗
- 将棋　　10の220乗　　アマ4段
- 囲碁　　10の360乗　　アマ5級

# 必勝法がわからないならば．．．

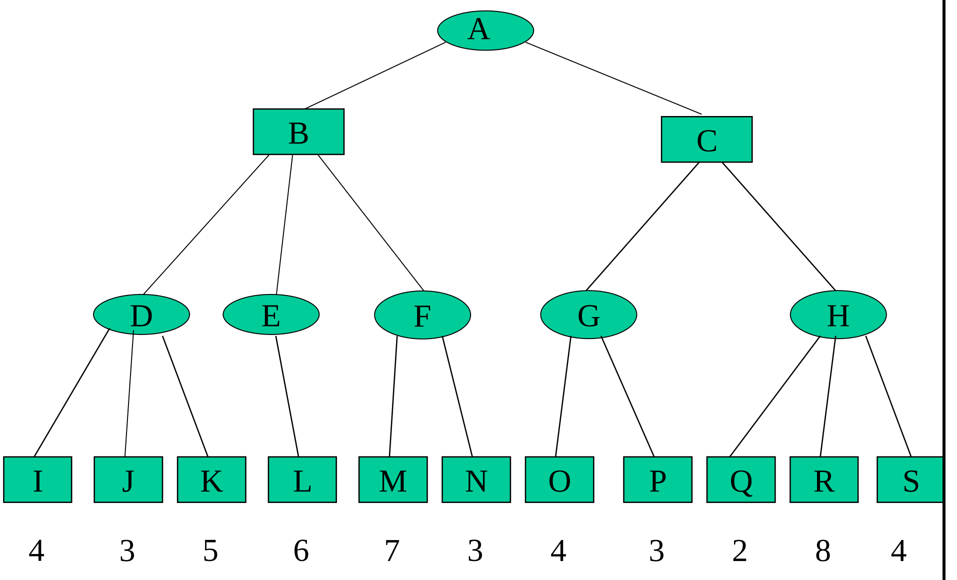
- ミニマックス法
- アルファーベータ法
- 静的評価関数

将棋では，評価関数の要素5，６－20程度で7－10手程度先読み

# ゲームの木



# ゲームの木の評価




# ミニマックス法（1）

1. N手先のノードの評価値を計算する
2. K手先のノードの評価値がわかっている時、（K－1）手先のノードの評価を以下のように行う。
   – （K-1）手先が自分の手先のノードである時
     - V（N）= max（V（N1）,V（N2）, V（N3）, …）
     - ただし、ノードNに連結している子ノードN1,N2,N3,..の評価値の最大値をNの評価値とする。
   – （K-1）手先が相手の手番のノードである時
     - V（N）= min（V（N1）,V（N2）, V（N3）, …）
     - ただし、ノードNに連結している子ノードN1,N2,N3,..の評価値の最小値をNの評価値とする。

## ミニマックス法（2）

2. 2の続き

   上記を繰り返して、N手先ノード、（N－１）手先ノード、（N－２）手先ノード、…..、２手先ノード、１手先ノード、ルートノードのように各ノードの評価値を計算する
3. ルートノードと同じ評価値をもつ１手先のノードへ移るオペレータを選択する。

## アルファベータ法（１）

- ミニマックス法
  1. ゲーム木のすべての末端ノードの盤面の評価値を計算
  2. 交互に最大値と最小値を求めて、中間ノードを計算
  3. ルートノードへ至った段階で最善手を求める
- ミニマックス法の問題点
  – 先読みの深さが深いと、末端のノード数が膨大になり、各ノードの評価値を求める時間が非常にかかる

# アルファベータ法（2）

- ミニマックス法での計算時間の短縮
  - 以下を並行しておこなう
    - 盤面の評価
    - 最大値と最小値の計算
  - 一部のノードの評価を省略できる
- アルファ（α）カット
  - 相手の手番ノードの評価値の計算に反映されないノード生成を省略すること
- ベータ（β）カット
  - 自分の手番ノードの評価値の計算には反映されないノードの生成を省略すること