2014年度　卒業論文

ハーモニーサーチアルゴリズムとニューラルネットワークによるオセロ戦術の学習

指導教員 ゴンサルベス タッド 准教授

上智大学 理工学部 情報理工学科

A1178647 岡田 良健

目次

1. はじめに 4
   1. 研究背景 4
   2. 研究目的 4
2. 機械学習 5
   1. 機械学習とは 5
   2. 機械学習の分類 5
      1. 教師あり学習 6
      2. 教師なし学習 6
3. ニューラルネットワーク 7
   1. ニューラルネットワークとは 7
   2. ネットワークモデル 8
      1. 階層型ネットワーク 8
      2. 相互結合型ネットワークネットワーク 8
   3. 階層型ネットワーク 9
      1. 単純パーセプトロン 10
      2. 多層パーセプトロン 12
   4. 誤差逆伝搬法(Back Propagation : BP法) 14
      1. BP法とは 14
      2. アルゴリズム 14
4. ハーモニーサーチアルゴリズム 15
   1. メタヒューリスティックアルゴリズムとは 15
      1. 概要 15
      2. 特徴 15
   2. ハーモニーサーチアルゴリズムとは 17
      1. 概要 17
      2. 特徴 17
      3. 確率的導関数 17
      4. 適用例 19
   3. アルゴリズム 20
5. ゲーム 25
   1. ゲーム理論とは 25
   2. ゲーム木 25
      1. ゲーム木とは25
      2. ミニマックス法26
      3. αβ法28
6. オセロ 29
   1. アブストラクトゲームとは 29
   2. オセロとは29
   3. ルール 29
   4. 戦術 31
   5. AI 32
7. 実験 33
   1. 実験概要 33
      1. 目的33
      2. 主な設定33
   2. 実験方法 33
      1. ハーモニーサーチアルゴリズム33
      2. ニューラルネットワーク35
      3. AI37
      4. 対戦38
      5. 観測者38
   3. 実験結果 39
      1. 対観測者 39
      2. 対人43
   4. 考察47
8. まとめ 51

参考文献 52

謝辞 53

**1 はじめに**

**1.1 研究背景**

　世の中には様々な問題が存在する. もし問題の与えられた制約と問題解決に向けた目的が明確化されていれば, それは最適化問題である. 選択肢の中でできるだけ効率的に物を配置するにはどうすればいいか, 一番強い戦術は何なのか, どのように過ごせば一番楽しいのか, と世の中には様々な最適化問題がある. 最適化問題の例としては, 様々な複数の都市を全て1回ずつ回る最小の経路を求める巡回セールスマン問題や, 価値と容積がわかっている複数の品物を一定の容量以下になるように詰める場合に品物の価値の総和が最大になる取り方を求めるナップサック問題などが有名だ.

　ゲームも最適化問題として考えることができる. また, ゲームにおける各プレイヤーの行動やプレイヤー間の競争や協力といった動きは, 現実の状況や動きを単純なモデル化したものとして捉えることができる. ゲームにおいてはルールが制約であり, 各プレイヤーは自らの利益の最大化が目的である. そして各プレイヤーは目的が達成できるように自らの戦術に従って自らの行動のために意思決定をする. ゲームのモデルの例としては, 自分の利得の優先と協調の行動の選択肢が噛み合わない状況である囚人のジレンマや, 他社の供給量が自社の供給量に影響を及ぼすクールノー寡占市場などが有名だ.

　ゲームの研究は今日までに多くなされており, 様々なゲームAIの開発がされている. その多くは強いAIを作ることを目的とした開発である. そしてその多くのAIは強くするための手段として, AIが確かな意思決定ができるように, ゲームの数多くの専門知識をAIに入力する方法がとられてきた. つまり, 人間が経験的に集めてきたデータをゲームの専門知識として表現したものを入力された専門家としてのAIが開発されることが主流である.

**1.2 研究目的**

　本研究の目的は, 「大きな複雑度を持つ問題に対し専門知識をベースとせずに, 機械学習によって状況評価を学習すること」を研究することである. これは, ある問題に対して限られた選択肢の中から, 問題解決のための行動の意思決定の方法を, 機械学習を利用してコンピュータが独自に見つけ出すことができるかに着目している. 本研究では十分複雑な問題としてオセロを題材とし, 機械学習の方法としてオセロAIの状況評価の役割を持つニューラルネットワークとその重みを改善するハーモニーサーチアルゴリズムを組み合わせた機械学習による方法を用いた.

**2 機械学習**

**2.1 機械学習とは**

　機械学習とは, 人間の知能的能力をコンピュータで実現することを目指す分野または技術である.

　人間の知能には三つの能力がある. 知識の獲得をする学習能力, 物事の特徴やパターンを認識して理解する能力, 蓄積した経験や知識をもとに新たな問題の解決方法を推論する能力である. 学習は他人が獲得した知識や自らの経験の概念化により作られた知識を知識ベースに蓄積することによりなされる. 知識ベースの拡大は理解の範囲の拡大へとつながる. 十分な理解能力は問題解決のための推論能力の向上へとつながる. そして新たな問題に直面したときに, その問題を解決へと導くのと共にその経験を概念化することができ, 新たな学習を行う. 人間はこれを繰り返すことにより知識の増加をし, 知能の三つの能力を成長させる.

　機械学習において, 人間の経験にあたるものはコンピュータに入力されるデータである. 数多くのデータを元にコンピュータは知識の獲得, または特徴やパターンの認識と理解, または新たな問題の解決方法を推論する能力を身につけていく. ただし, 入力されるデータや直面する新たな問題は, あらかじめ人間がコンピュータに学習を望むために設定した範囲内のものである.

　機械学習は入力されるデータから知識やルールを自動獲得するため, データの数が多ければ多いほど学習の精度は高くなる. と同時に, 正確なデータを与えることが重要である. 学習する問題が複雑である場合やデータにノイズが含まれる場合, 学習のためにはより多くのデータを必要とする.

　機械学習は様々な分野と関係している. 具体的には, 人工知能, パターン認識, データマイニング, 確率論, 統計, 並列計算, データベース, アルゴリズム, データ構造, クラスタリング, 数値最適化などである. また, 応用分野も広い. 具体的には, 検索エンジン, 文字認識, 画像解析, 評判分析, 市場予測, 医療診断, レコメンデーションクラスタリングなどである.

**2.2 機械学習の分類**

　機械学習において, コンピュータの学習のための改善を訓練という. また, 訓練のためにコンピュータに与えられる入力データの集合を訓練データまたは学習データという. また, 一般に学習後に学習がきちんとなされたか確認するために訓練データとは別のテストデータを用いてテストを行う.

　訓練データの種類によって, 機械学習は教師あり学習と教師なし学習の2種類に分類される.

**2.2.1 教師あり学習**

　教師あり学習とは, 入力とその入力に対する目標出力である教師信号を組とした教師事例を訓練データとするものである. 教師事例は正解事例または訓練事例ともよばれる. 目標出力は一般的には正しい解答または正しい分類であり, ラベルとよばれる. 出力の値がカテゴリ値の場合は分類といい, 連続値の場合は回帰といわれる. 訓練を繰り返すことで, 教師信号とコンピュータによる出力の差を小さくしていく.

　学習後, コンピュータは分類や回帰を行うことができる.　具体例として, スパムメールの分類を挙げる. 入力にメールとその入力するメールがスパムメールか否かという情報の組を訓練データとする. 学習後, コンピュータはスパムメールという概念を理解しており, あるメールを入力として受け取った場合にそのメールがスパムメールかどうかを出力してスパムメールの分類をすることができる.

**2.2.2 教師なし学習**

　教師なし学習とは, 入力のみを訓練データとするものである. 教師あり学習における目標出力は存在しない. コンピュータはデータ表現を学習する場合が多い. 訓練を繰り返すことでコンピュータはデータの要素を理解していく.

　学習後, コンピュータはクラスタリングや外れ値検出を行うことができる. 具体例として, 店舗クラスタリングを挙げる. 店舗情報を訓練データとする. 学習後, コンピュータは店舗の店舗情報の要素を理解しており, 似た店舗をグループ化して分類することができる.

**3 ニューラルネットワーク**

**3.1 ニューラルネットワークとは**

　ニューラルネットワークとは, 生物神経システムにおける神経ネットワークの構造と機能を模した数学モデルであり, 学習アルゴリズムである. 生物神経システムの生物ニューラルネットワークと対比して人工ニューラルネットワークともいわれる.

　生物神経システムにおいて, 個々の神経細胞をニューロンという. ニューロンとその興奮の伝達の仕組みについて簡潔に説明する. ニューロンの入力部分は樹状突起といい, 枝上の繊維の集まりである. ニューロンの出力部分は軸索といい, 長い側枝を持つ繊維である. また, 軸索と樹状突起の接続点をシナプスという. ニューロンは膜電位が閾値を超えると発火する. あるニューロンによって発射された信号は, 軸索側枝の末端からシナプスを経て他のニューロンの樹状突起に達する. これによりニューロン間の興奮が伝達される. 生物ニューラルネットワークでは, ニューロンは密に配置され, 無数の神経枝が互いに交差している. 人の脳にあるニュートンの総数は1000億と推定されている.

　ニューラルネットワークにおいては各ノードがニューロンに相当し, ノード間の枝がシナプスに相当する. 値としては, ノードの活性値とノード間の結合の重みの値が存在する. ニューラルネットワークへ入力がなされたとき, 入力と重みに従った値が閾値の役割の変換をなされたあとに各ノードに伝わり各ノードが活性値を持ち, 最後に出力が算出される. データ値としては重みの値のみを持ち, ニューラルネットワークによる学習は重みを改善し最適化することである. ただし, 学習後の重みは一意に決まるとは限らない.

　ニューラルネットワークにおいてはニューロンが発火するかの判定, つまり, 膜電位が閾値を超えたかどうかの判定の役割のために, 入力に対して0, 1を返す任意の出力関数を利用する. これは活性化関数または閾値関数とよばれる. 一般には, ニューラルネットワークの入出力の関係を非線形にするために非線形の関数が推奨され, 重みの改善のアルゴリズムとの相性のために微分可能で連続な関数が望まれる. 非線形で微分可能で連続な関数であるシグモイド関数やtanh関数がよく利用される. また, 閾値の調整のために入力層に追加のノードとしてまたは中間層の各ノードにバイアスという定数を加えることが推奨される. これにより, 学習の過程において重みの改善と共に閾値の改善もなされる.

　ニューラルネットワークはn次元のデータのベクトルの入力に対して何らかの変換を行いm次元のベクトルを出力する. よって, ニューラルネットワークは関数とみなすことができる. そしてニューラルネットワークによる学習とは, 求めたい関数への近似度の改善である. ニューラルネットワークは学習の結果, 求めたい未知の関数を近似する関数となる.

**3.2 ネットワークモデル**

　ニューラルネットワークのネットワークモデルは, 階層構造をもつ階層型ネットワークと階層構造を持たない相互結合型ネットワークの2種類に分類される.

**3.2.1 階層型ネットワーク**

　階層型ネットワークとは, 階層構造を持ち, 入力層から出力層へと一方向のみに信号が伝搬するネットワークのことである. フィードフォワードネットワークともいう.

　以下の図3.2.1.1が階層型ネットワークのイメージである.

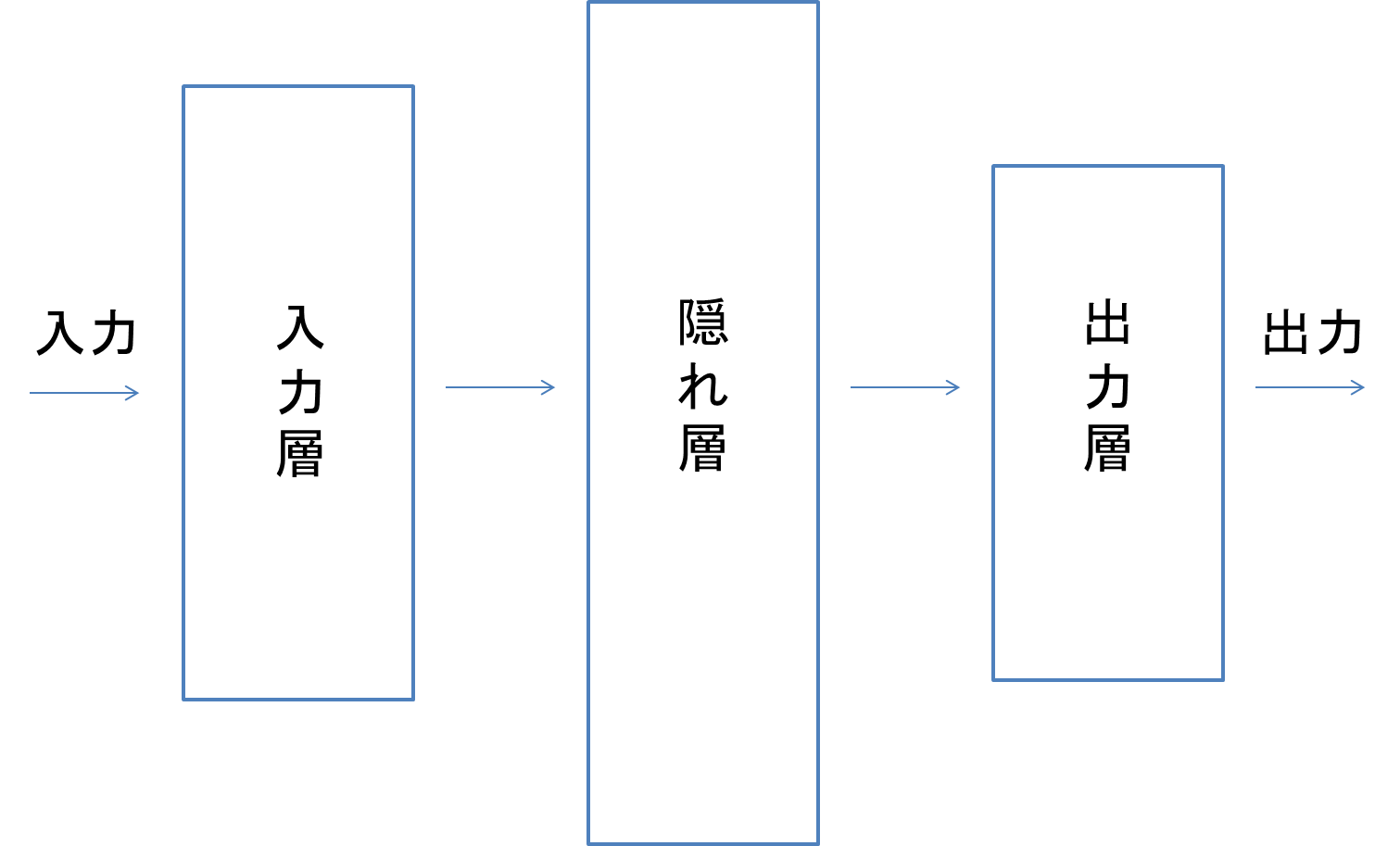


図3.2.1.1 階層型ネットワーク

**3.2.2 相互結合型ネットワーク**

　相互結合型ネットワークとは. 全ニューロン間に双方向の結合があるネットワークのことである. 非階層型ネットワークやリカレントネットワークともいう.

　以下の図3.2.2.1が相互結合型ネットワークのイメージである.

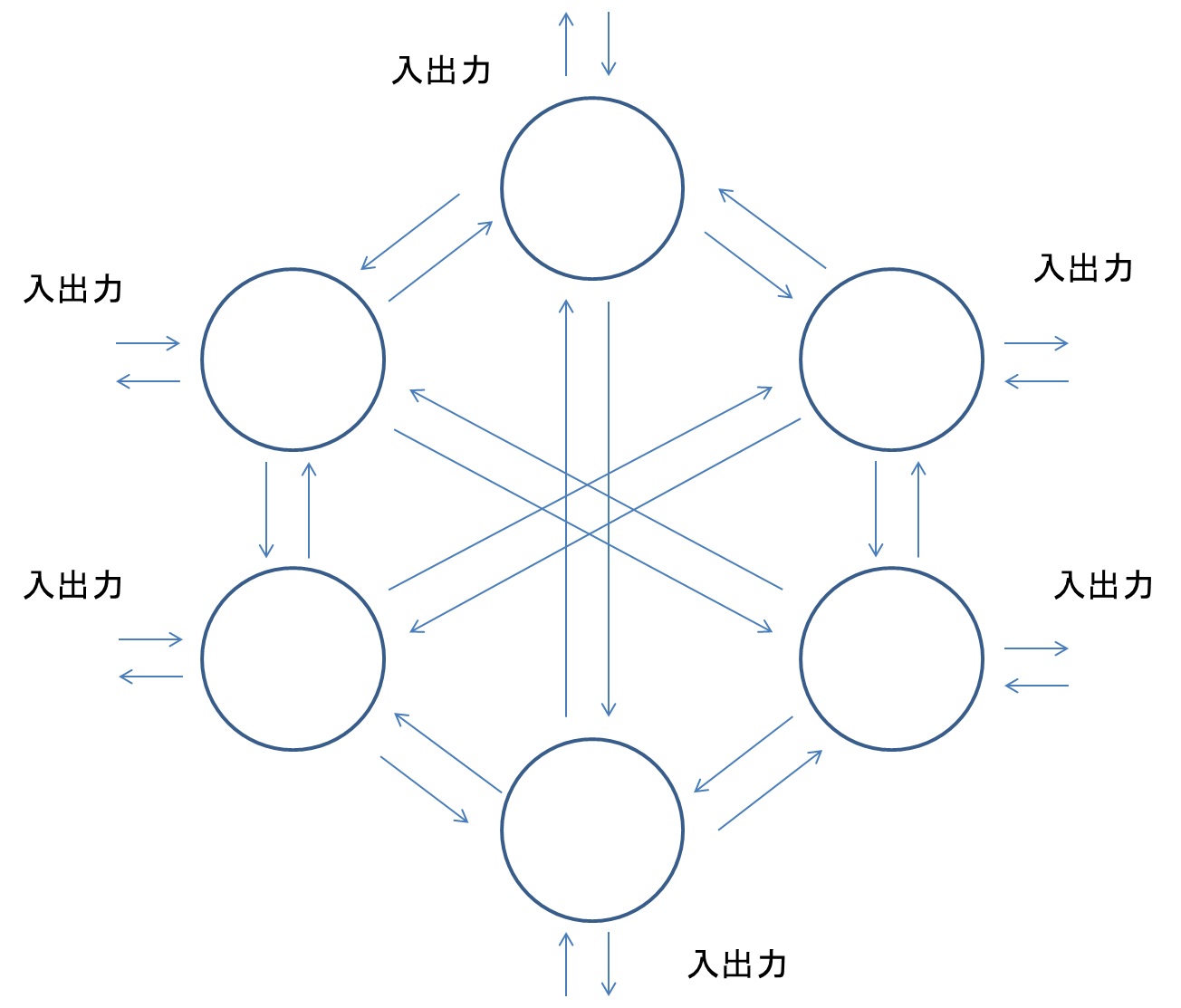


図3.2.2.1 相互結合型ネットワーク

**3.3 階層型ネットワーク**

　階層型ネットワークについて掘り下げる. 階層型ネットワークとは, 階層構造を持ち, 入力層から出力層へと一方向のみに信号が伝搬するネットワークのことであった. 入力層のノード数は入力データのベクトルの要素数であり, 出力層のノード数は問題に対して求めたい解の表現に依存する. 例えば, 画像の文字を判定するパターン認識の問題において, 画像の入力データに対し, 1つのノードの0/1の出力でYES/NOを表すことも, 2つのノードのそれぞれの連続値でYES度とNO度を表すことも可能である.

　入力層と出力層の間にある層を中間層または隠れ層という. 一般に中間層のノード数は定まっておらず, また, 中間層の数も定まっていない. それらは問題の大きさに対して適した数に人間が設定する必要がある. ノード数や中間層の数が大きいほど複雑な問題に対して有効となるが, 計算量は大きくなるとともに過学習に陥りやすい. ただし過学習とは, 訓練データに適応しすぎたために訓練データに類似した新たな問題の入力データに対する解決能力が低下してしまう状態のことである. 典型的には, 入力層と1つの中間層と出力層の3層からなるネットワークモデルが利用される. 歴史的にはニューラルネットワークの中間層は増加の傾向があり, 複雑な問題の学習にも利用されている.

**3.3.1 単純パーセプトロン**

　中間層が存在せず. 出力層のノードが1つしかないものを単純パーセプトロンという. 入力層が中間層にそのまま信号を伝えているだけの3層構造だとみなす場合もある. 以下の図3.3.1.1は単純パーセプトロンの構造を表したものである.

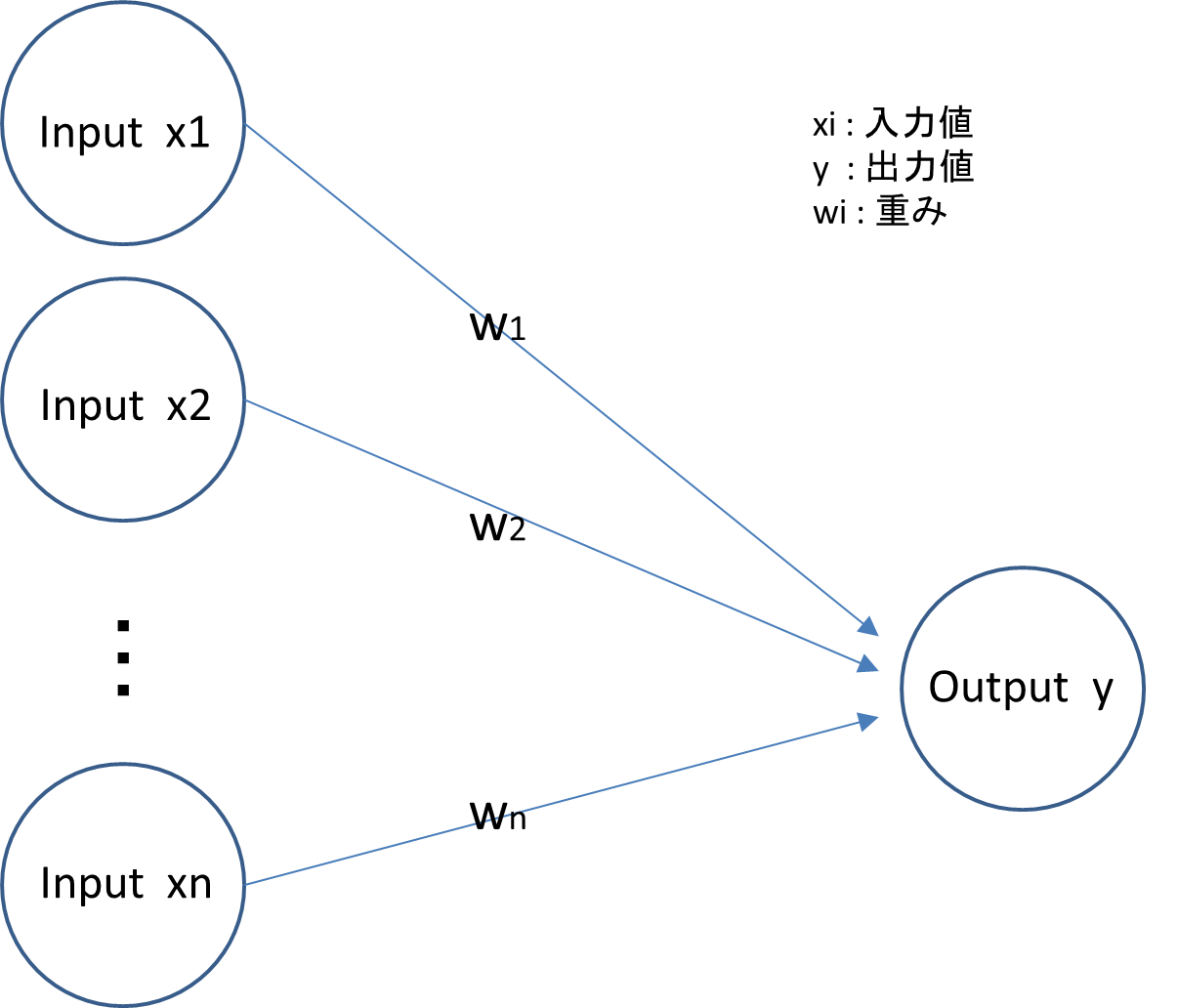


図3.3.1.1 単純パーセプトロン

　単純パーセプトロンは, 線形分離可能な問題のみを解決することができ, 線形分離不可能な問題を解決することができない. つまり, 線形分離不可能な問題を解決する重みの値の組合せが存在しない. ただし線形分離可能であるとは, n次元空間上の二つの集合をn-1次元の超平面で分離できることをいう.

　線形分離不可能な問題の典型例として, XOR演算を挙げる. XOR演算の入力は2つの値であり2次元の入力である. 出力は1つの値であり1次元の出力である. 以下の表3.3.1.1は, XOR演算の入力と出力の対応の表である

表3.3.1.1 XOR演算の入力と出力



　このときの単純パーセプトロンの構成は以下の図3.3.1.2になっている.

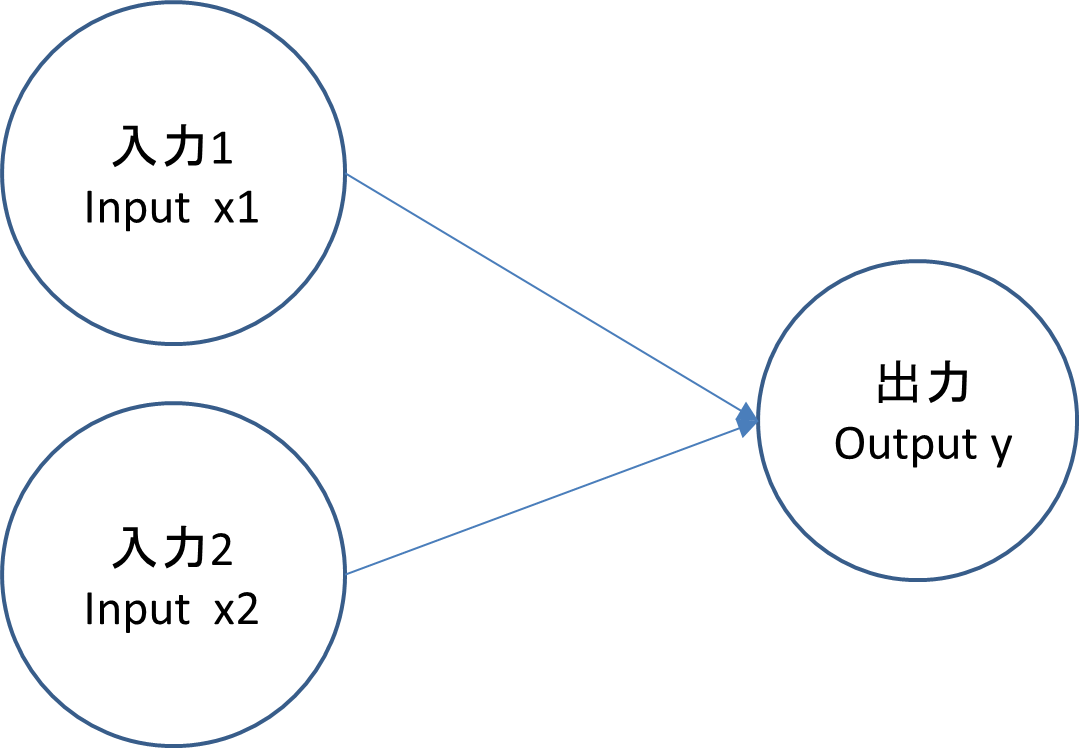


図3.3.1.2 XOR演算と多層パーセプトロン

　また, 2次元空間上でXOR演算の入出力は以下の図3.3.1.3のグラフになる.

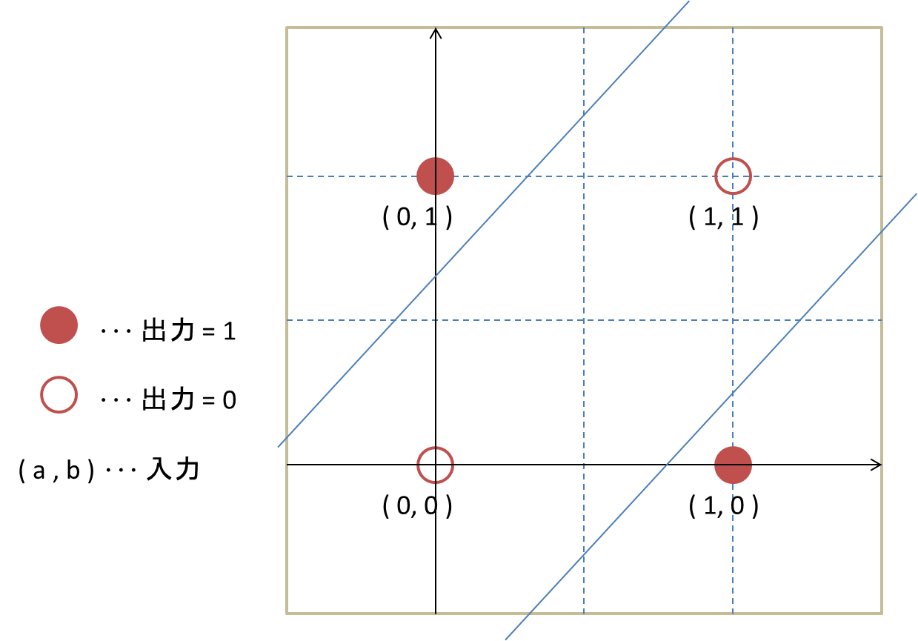


図3.3.1.3 XOR演算の入出力が線形分離不可能な状態

線形分離可能であるためには, 出力が0の集合と出力が1の集合がある直線によって分離できなければならない. しかし, 出力が0の集合と出力が1の集合を分けるためには2つの直線が必要であり, それは不可能である. この問題は多層パーセプトロンでは解決される.

**3.3.2 多層パーセプトロン**

　中間層が存在するものを多層パーセプトロンという. 単純パーセプトロンとは違い, 出力層のノードは1つとは限らない. 以下の図3.3.2.1は多層パーセプトロンの構造を表したものである.

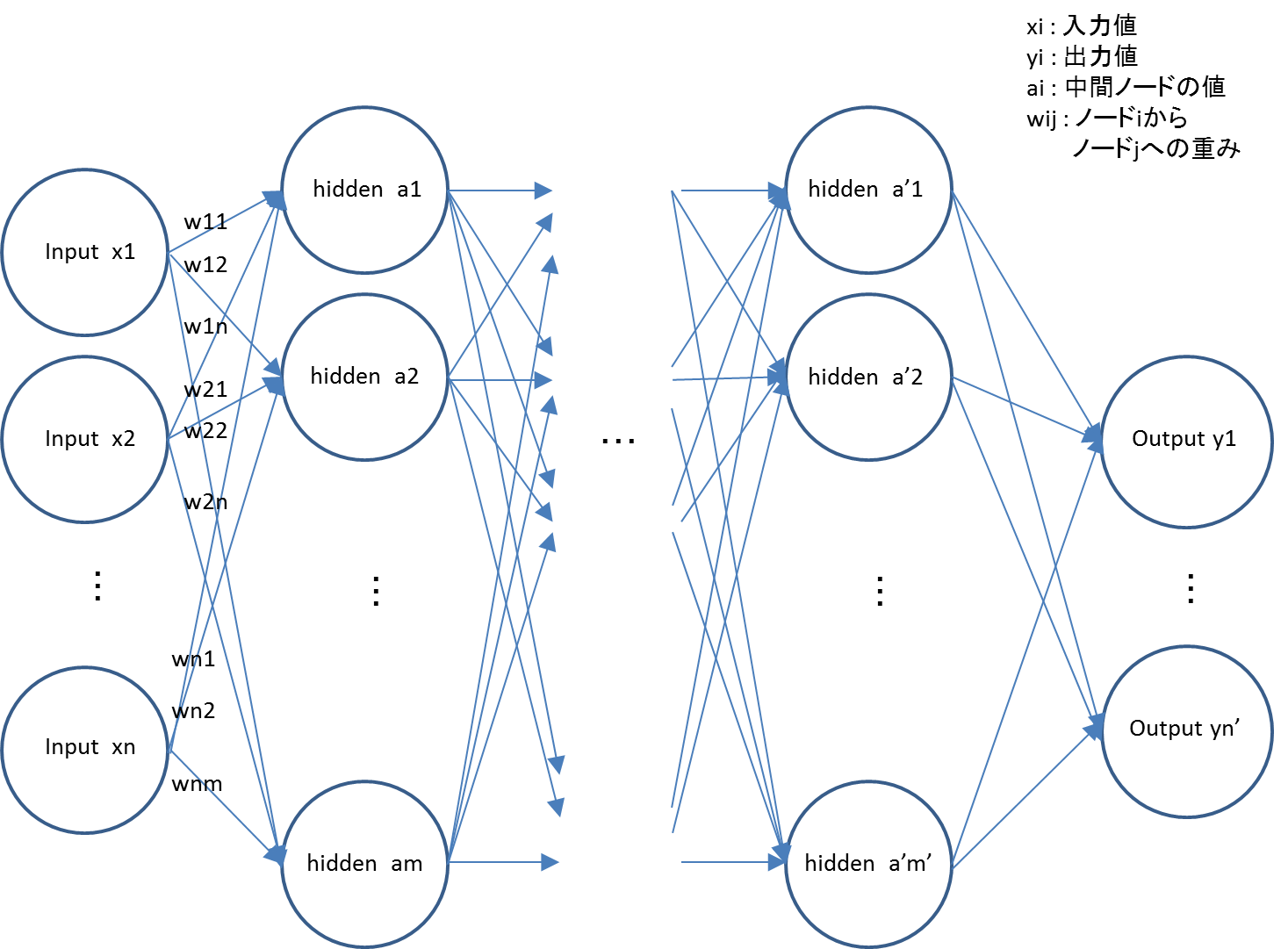
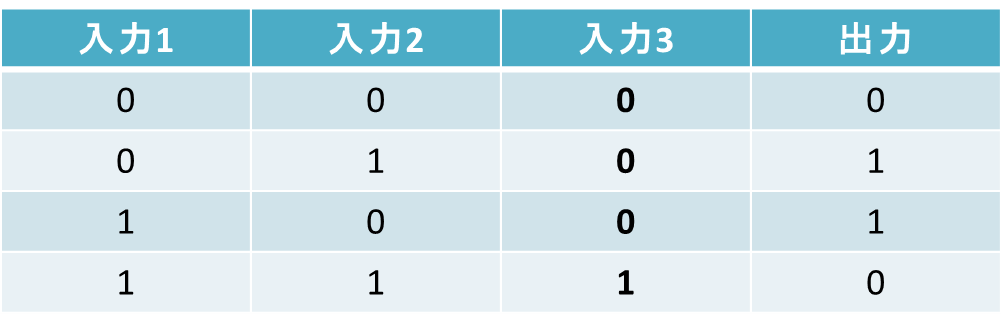


図3.3.2.1 多層パーセプトロン

　多層パーセプトロンは, 単純パーセプトロンでは線形分離不可能な状態であった問題も解決することができる.

　改めて例としてXOR演算を挙げる. 中間層のノード数を3とする. このとき, 中間層からの入力は3次元の入力である. 出力は1つの値であり1次元の出力である. 以下の表3.3.2.1のように中間層ノードと出力の対応を作ることができる.

表3.3.2.1 中間層ノードの入力とXOR演算の出力



　このときの多層パーセプトロンの構成は以下の図3.3.2.2になっている.

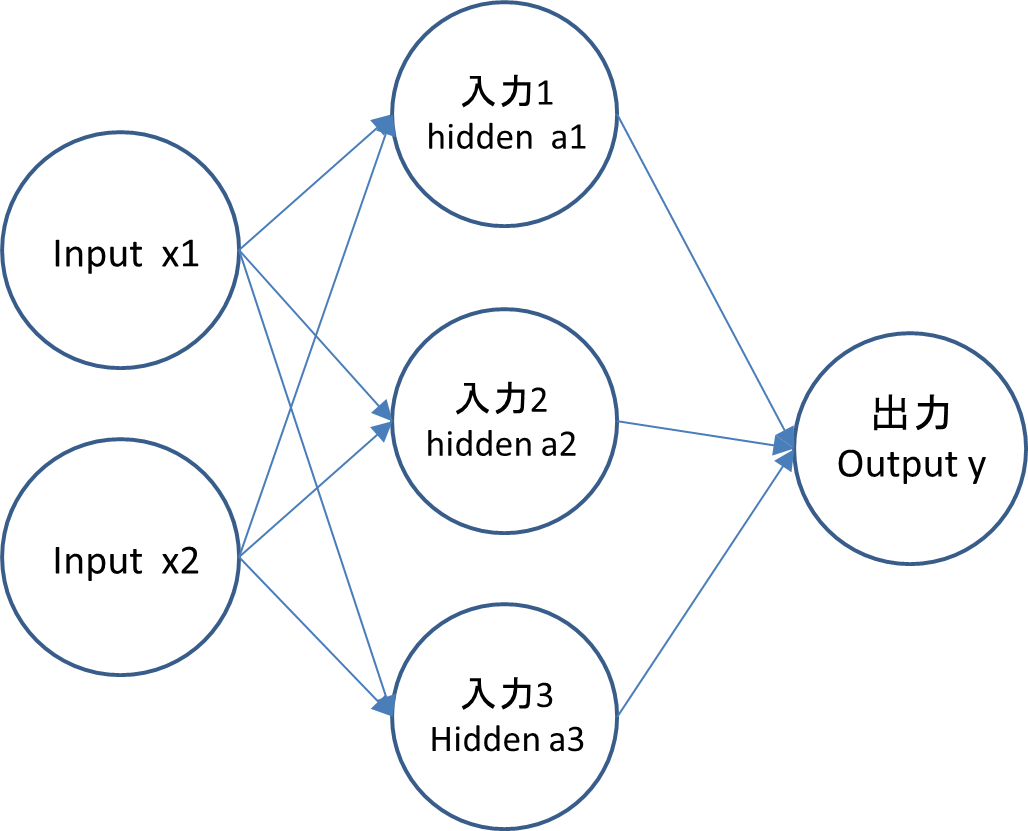


図3.3.2.2 XOR演算と多層パーセプトロン

　また, 3次元空間上で中間層ノードの入力とXOR演算の出力は以下の図3.3.2.3のグラフになる.

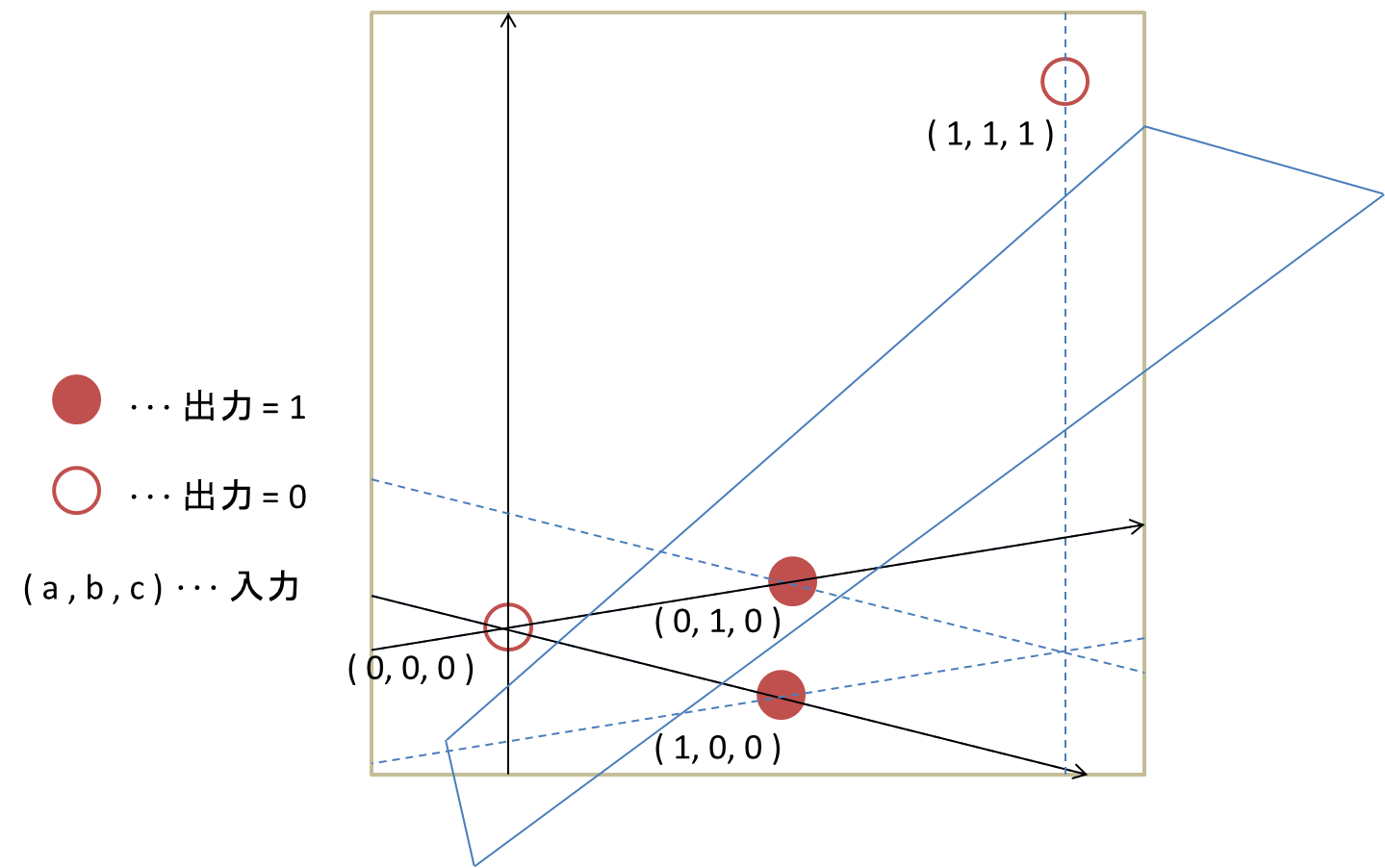


図3.3.2.3 XOR演算の入出力が線形分離可能な状態

**3.4 誤差逆伝搬法(Back Propagation : BP法)**

**3.4.1 BP法とは**

　BP法とは, ニューラルネットワークの教師あり学習の方法であり, 重みを改善するアルゴリズムである. 出力層の出力と目標出力の誤差に中間層の各ノードが与えている影響の大きさを考慮し, 誤差が小さくなるように重みの改善を行う.

　逆伝搬ネットワークは一般にいくつかの中間層を持ち, 各層が上下の層と完全結合している多層パーセプトロンである. よって, ネットワークに入力が与えられると入力層から出力層に向かって活性値の更新の順伝搬をして出力がなされる. 一方で, ネットワークが重みの改善を行うときには出力層から入力層に向かって逆伝搬がなされる特徴を持つ.

　BP法の誤差は平均二乗誤差で定義され, 誤差が十分小さくなった場合に収束とみなして学習の完了とする. 誤差を最小にするアルゴリズムとしては一般に最急降下法を用いる. 最急降下法とは, 最大勾配の方向に値を更新していき, 関数の最小値を探索する勾配法である. この勾配法は大域的最適解ではなく局所最適解に陥る可能性があるため, 適切な初期値を与えて探索を行う必要がある. また, 活性化関数に微分可能で連続な関数が必要である.

**3.4.2 アルゴリズム**

　BP法のアルゴリズムは以下の通りである.

1. 各2層間の重みの初期化

2. 次の訓練データを入力層に入力

3. 各中間層の値を算出

4. 出力層を算出

5. 教師信号と出力層の値から出力層の誤差を算出

6. 出力層の誤差と中間層の値から中間層の誤差を算出

7. 中間層・出力層間の重みを更新

8. 各中間層間の重みを更新

9. 入力層・中間層間の重みを更新

10. 誤差が十分小さい場合終了

11. 2へ戻る

**4 ハーモニーサーチアルゴリズム**

**4.1 メタヒューリスティックアルゴリズムとは**

**4.1.1 概要**

　メタヒューリスティックアルゴリズムとは, 任意の問題に対して近似的に解を求めるアルゴリズムである. 学問分野としては, コンピュータサイエンスやオペレーションズリサーチの分野に属する.

　最適化問題はあらゆるところに存在する. 最適化問題では一般に目的関数の最大化または最小化を求める. 最適化問題には工業デザイン, 経済, プランニング, ルーティングなど様々なテーマが存在し. 最適化の対象はお金, 人, 時間, 資源など様々である. そして, 多くの現実世界の問題は非線形であり多様であり複雑な制約をもつ. しかも, 異なる目的は相反することもしばしばである. 目的が一つであっても最適解が存在する保証はない. 一般にそのような問題は最適解または準最適解を求めることは難しい. そのような場合にメタヒューリスティックアルゴリズムは有効である.

　最適化問題を解決するために様々な最適化アルゴリズムが開発されている. アルゴリズムは最適化のために用いる方法によって様々な基準から分類される. 目的関数の導関数を利用する勾配ベースのアルゴリズム, 探索時に跡を考慮する軌跡ベースのアルゴリズム, 複数のエージェントを用いる個体群ベースのアルゴリズム, ランダム性をアルゴリズムに含まない決定的なアルゴリズム, ランダム性をアルゴリズムに含む確率的なアルゴリズム, 局所最適解を探索するアルゴリズム, 大域最適解を探索するアルゴリズム などである. 問題を解決するためには, 様々な最適化アルゴリズムの中から問題に適するようにアルゴリズムの選択または組み合わせ, 場合によってはアルゴリズムの調整または新たなアルゴリズムの創造が必要である. しかし, それらは問題によっては難しいことであり, 問題に適したアルゴリズムが見つかったとしても実装や計算量などの問題が生じる場合がある. そのような場合にメタヒューリスティックアルゴリズムは有効である.

**4.1.2 特徴**

　メタヒューリスティックアルゴリズムはランダム性を含む拡散と収束によりできている. 拡散の役割は, 探索範囲を大域的なものにすることで解空間を開拓して大域的最適解が見つからないようにしないことある. これは, 局所最適解に陥ることを間接的に防止する. 収束の役割は, 探索範囲を局所的なものにすることで範囲内を集中的に調査して最適解を見つけることである. メタヒューリスティックアルゴリズムにおいてこの2つのバランスは探索範囲や収束速度に直接関わる重要な要素である. メタヒューリスティックアルゴリズムによっては, このバランスも最適化問題を解決するさいに最適化の対象として含めることが可能である. これらにより, メタヒューリスティックアルゴリズムは最適解または準最適解を許容可能な実行時間内に求めることができる. ただし, メタヒューリスティックアルゴリズムは最適解を見つける保証がないことに注意すべきである.

メタヒューリスティックアルゴリズムの例として以下を挙げる.

・Genetic Algorithm

・Ant Colony Optimization

・Particle Swarm Optimization

・Harmony Search

・Simulated Annealing

・Tabu Search

メタヒューリスティックアルゴリズムには様々な種類が存在する. その中で個体群ベースのアルゴリズムの大方の流れは等しく, 以下の図4.1.2.1の通りである.

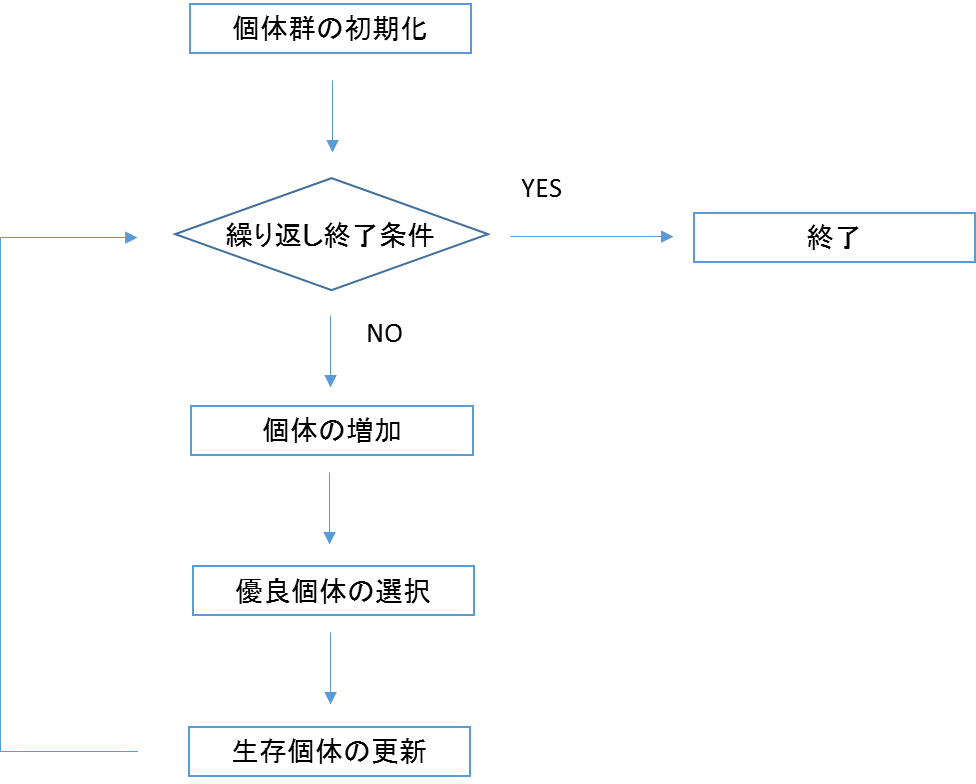


図 4.1.2.1 個体群ベースのメタヒューリスティックアルゴリズムのフロー

　個体群ベースのメタヒューリスティックアルゴリズムでは目的関数のことを適応度関数やフィットネス関数ともいい, 個体が問題に対して適応している度合いを適応度(Fitness)という. アルゴリズム終了時に適応度が最も高い個体が準最適解となる.

**4.2 ハーモニーサーチアルゴリズムとは**

**4.2.1 概要**

　ハーモニーサーチアルゴリズムとは, 個体群ベースのメタヒューリスティックアルゴリズムの一種であり, 音楽家の調和の探し方をアイデアのベースにして作られたメタヒューリスティックアルゴリズムである. ハーモニーが個体である.

　2001年に Zong Woo Geem によって考案された. 音楽家のインプロヴァイゼーション（improvisation, 即興演奏）の過程からインスピレーションされたものである.

　ハーモニーサーチアルゴリズムのイメージは次の通りである. 音楽家 (= decision variable, 決定変数) たちが楽器のピッチの範囲 (= decision variable’s value range, 決定変数の定義域) 内で音符 (= value, 値) を奏でて (= generate, 生んで) ハーモニー (= solution vector at certain iteration, 解ベクトル) をなし, 聴衆 (= objective function, 目的関数) が聞く. 以上でハーモニーの改善を繰り返すことで最高のハーモニー(= global optimum, 大域最適解)を見つける.

**4.2.2 特徴**

　ハーモニーサーチアルゴリズムは以下の性質を持つ.

・改善の対象の変数の初期値の設定を必要としない

・局所最適解を避けることができる

・連続値も離散値も扱うことができる

・離散値を扱う場合に, 確率的導関数が適用できる

・組み合わせ最適化やパラメータ最適化に適している

・構造がシンプル

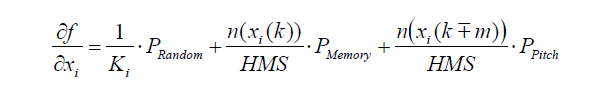
・交叉ではなく調和操作によって変数同士の関係を扱うため, GAの積み木仮説を克服している

　構造がシンプルであるため, コーディングが楽であり, 問題に合わせてカスタマイズすることが可能であり, 他のアルゴリズムと組み合わせることも可能である. 純粋なハーモニーサーチアルゴリズムを発展させる研究もなされている.

**4.2.3確率的導関数**

　ハーモニーサーチアルゴリズムの目的関数の確率的導関数は, Novel Stochastic Derivativeという. 決定変数が次に取り得るいくつかの離散的な値, つまり候補値の中から, ある値を選択する確率を情報として示すものである. 全ての値を選択する確率の和は1である. 確率は世代の更新の繰り返しごとに変化する. 理想的には繰り返しごとに, 問題の最適解ベクトルに含まれる値を選ぶ確率が高くなるように変化していく.

　確率的導関数は以下の図4.2.3.1の式で与えられる.



(4.2.3.1)

図4.2.3.1 Novel Stochastic Derivativeの式 (Novel Derivative of Harmony Search Algorithm for Discrete Design Variables より引用)

式4.2.3.1のパラメータの意味は以下の通りである.

　Ki : 決定変数ベクトルXのi番目の決定変数xiの候補値の個数である.

　HMS : HM(Harmony Memory)内のハーモニーの数である. (= hms)

　PRandom : 次世代ハーモニーAnewの要素ai newの次の値が, 決定変数xiの候補値の中からランダムで選ばれた値を取る確率である. (= 1 - hmcr)

　PMemory : 次世代ハーモニーAnewの要素ai newの次の値が, HM内から選ばれた値をそのまま取る確率である. (= hmcr ･ (1 - par))

　PPitch : 次世代ハーモニーAnewの要素ai newの次の値が, HM内から選ばれた値の近傍値を取る確率である. (= hmcr ･ par)

　xi(k) : 決定変数ベクトルXのi番目の決定変数xiの候補値のk番目のものの値である.

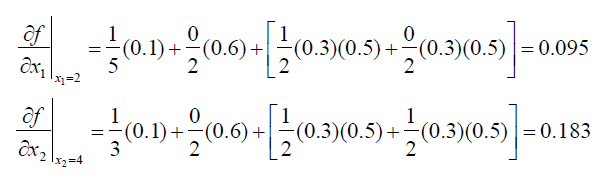
　n(xi(k)) : HM内のハーモニーでi番目の要素ai jがxi(k)であるものの数である.

　n(xi(k±m)) : HM内のハーモニーでi番目の要素ai jがxi(k–m)またはxi(k+m)であるものの数である.

　以下にNovel Stochastic Derivativeの例を挙げる.

　問題の設定は次の通りである. 決定変数x1が3つの候補値 { 1, 2, 3, 4, 5 } を持ち, 決定変数x2が3つの候補値 { 3, 4, 5 } を持ち, 問題の最適解ベクトルX は[ x1, x2 ] = [ 2, 4 ] である. ハーモニーサーチアルゴリズムのパラメータは, HMS = 2, PRandom = 0.1, PMemory = 0.6, PPitch = 0.3, m = 1と設定した. ある時点でHM内のハーモニーAi = [ a1, a2 ] はA1 = [ 1, 3 ], A2 = [ 5, 5 ] であった.

　このとき, 問題の最適解ベクトルにおけるNovel Stochastic Derivativeの値は以下の図4.2.3.2の値になる.

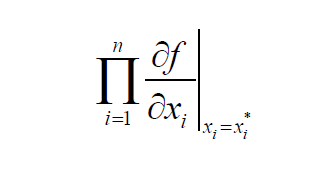


(4.2.3.2.1)

(4.2.3.2.2)

図4.2.3.2 Novel Stochastic Derivativeの計算例 (Novel Derivative of Harmony Search Algorithm for Discrete Design Variables より引用)

　また, 問題の最適解ベクトルが選ばれる確率は以下の図4.2.3.3の式で表される.



(4.2.3.3)

図4.2.3.3 問題の最適解ベクトルが選ばれる確率 (Novel Derivative of Harmony Search Algorithm for Discrete Design Variables より引用)

**4.2.4 適用例**

　ハーモニーサーチアルゴリズムは様々な問題やアプリケーションに適用されている. 適用例を以下に示す.

Real-world applications

・Music composition

・Sudoku puzzle

・Timetabling

・Tour planning

・Logistics

Computer science problems

・Web page clustering

・Text summarization

・Internet routing

・Visual tracking

・Robotics

Electrical engineering problems

・Energy system dispatch

・Photo-electronic detection

・Power system design

・Multi-level inverter optimization

・Cell phone network

Civil engineering problems

・Structural design

・Water network design

・Dam scheduling

・Flood model calibration

・Groundwater management

・Soil stability analysis

・Ecological conservation

・Vehicle routing

Mechanical engineering problems

・Heat exchanger design

・Satellite heat pipe design

・Offshore structure mooring

Bio & medical applications

・RNA structure prediction

・Hearing aids

・Medical physics

**4.3アルゴリズム**

　ハーモニーサーチアルゴリズムの流れは, 他の個体群ベースのメタヒューリスティックアルゴリズムと同様に, 初期化・個体の増加・優良個体の選択・個体群の更新からなる.

**4.3.1 定式化**

　ハーモニーサーチアルゴリズムは最適化問題を解くためのアルゴリズムであるので, 問題を解決するためにその問題の定式化が必要である. 定式化には, 目的関数と制約条件を明確にする意味がある. ハーモニーサーチアルゴリズムは目的関数に対して, 準最適解ベクトルA = { a1, a2, …, an } を求める.

　定式化は以下の図4.3.1.1でされる.

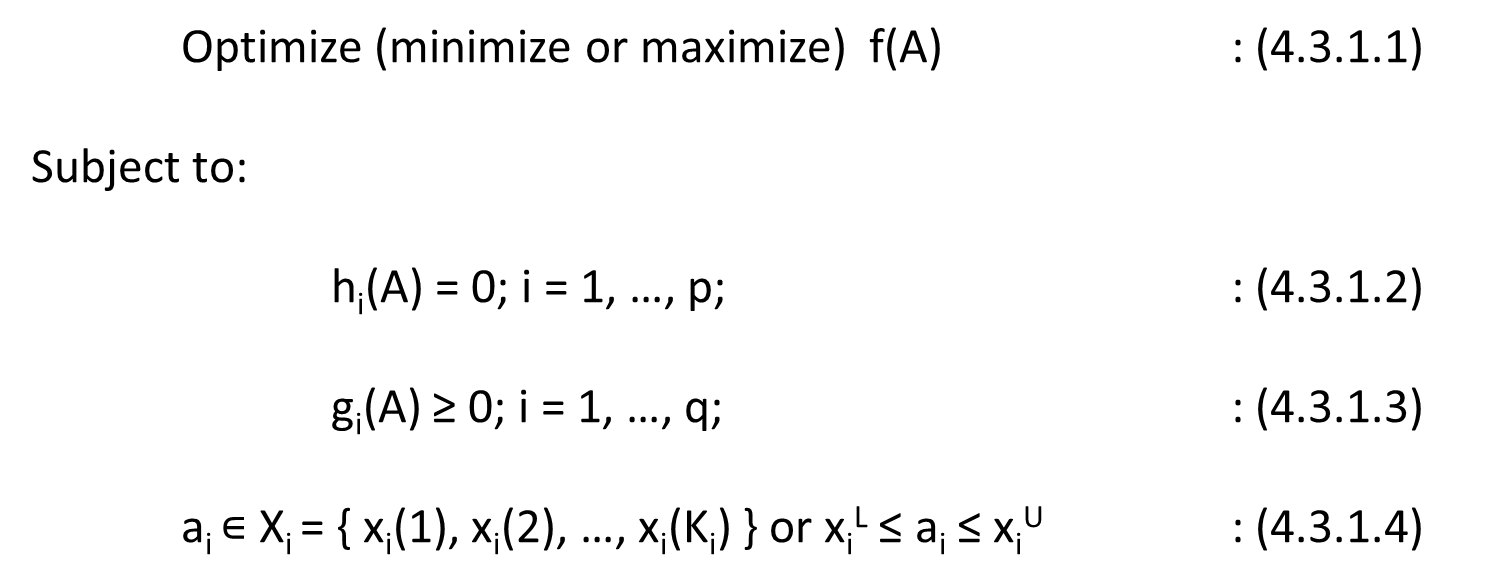


図4.3.1.1 定式化

式4.3.1.1 は目的関数を表している.

式4.3.1.2 は等式である制約式を表している. 制約式の有無は問題による.

式4.3.1.3 は不等式である制約式を表している. 制約式の有無は問題による.

式4.3.1.4 は決定変数の取り得る値の範囲を示している. 決定変数が離散値をとる場合は候補値の組ai ∊ Xi = { xi(1), xi(2), …, xi(Ki) } が範囲の制約であり, 連続値をとる場合は値xiL ≤ ai ≤ xiU が範囲の制約である.

　ハーモニーサーチアルゴリズムは基本的には上記のように目的関数と制約を考慮する. ただ追加の操作として, 制約を満たさない解ベクトルに対して目的関数に影響を与えるペナルティを与えて保持する, という操作をすることができる. また, ハーモニーサーチアルゴリズムはパレートセットと組み合わせることで多目的最適化問題にも適用することができる.

**4.3.2 パラメータ設定**

　問題の定式化がなされたら, 次にはパラメータの設定が必要である. ハーモニーサーチアルゴリズムには以下の公式パラメータがある.

・hms (harmony memory size) :

　解ベクトル(ハーモニー)の数. ハーモニーメモリー(HM)の大きさ.

　一般的な値は1から100. 典型値は30.

・hmcr (harmony memory considering rate) :

　次の値を生成するさいにHM内から値を選択する確率

　0 ≤ hmcr ≤ 1. 一般的な値は0.7から0.99. 典型値は0.9.

・par (pitch adjusting rate) :

　次の値を生成するさいにHM内から選択した値の近傍値から選択する確率

　0 ≤ par ≤ 1. 一般的な値は0.1から0.5. 典型値は0.3.

・fw (fret width, formerly bandwidth) :

　探索する近傍の幅の値.

　一般的な値は (0.01 × allowed range) to (0.001 × allowed range).

・MI (maximum improvisation) :

　解の改善を繰り返す回数. 全世代数

　一般的にはパラメータは固定されるが, 繰り返しと共に変動させてもよい. また, 場合によっては以下のパラメータも自ら設定する必要がある.

・解ベクトルの持つ決定変数の個数 (Dimension)

・決定変数の取り得る値の範囲または最小値と最大値 (Allow Range, min, max)

　また, ハーモニーメモリー(HM)は以下の図4.3.2.1の行列として考えられる. 各ハーモニーは決定変数の値と適応度の値を持つ. 以前のハーモニーサーチアルゴリズムではハーモニーAiの順を適応度でソートして f(A1) ≤ f(A2) ≤ … ≤ f(Ahms) とする必要があったが, 現在は必要ない.

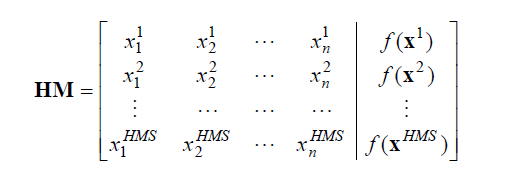


図 4.3.2.1 ハーモニーメモリー(HM) (State-of-the-Art in the Structure of Harmony

Search Algorithm より引用)

**4.3.3 Random Tuning**

　HMの初期化をするステップである.

　ランダムにベクトルAi = (a1, …, adim)をhms個だけハーモニーとして生成し, HMに保存する. hmsより多くのハーモニーを生成して, その中から優秀なhms個のハーモニーを選択することも可能である. 生成の仕方はRandom Selection(4.3.X Harmony Improvization 参照)と同じである.

　ランダムに音を出すイメージである.

**4.3.4 Harmony Improvization**

　ハーモニーの値の即興をするステップである.

　設定したパラメータの確率に従って以下の操作をし, 新しいハーモニーAnewの各値ai newは生成される.

　即興演奏でハーモニーを見つけているイメージである.

[ Random Selection ]

　ランダムに値aiを生成する. 確率 1 – hmcr で行われる.

　ai newは制約条件である決定変数の範囲 ai new∊ Xi = { xi(1), xi(2), …, xi(Ki) } or xiL ≤ ai new ≤ xiU 内である.

[ Memory Consideration ]

　HM内から値を選択して値ai newを生成する. 確率 hmcr で行われる.

　HM内の値ai jは HM = { Ai1, …, Aihms } 内からランダムに選択される. インデックスjは典型的には一様分布U(0,1)に従い, j ← int(U(0,1) ･ hms) + 1 である. ただ, これは違う分布にすることも可能である. 例えば [U(0,1)]2を利用すれば小さいjほど選択されやすくなる. このとき, HMが目的関数f(X)の値でソートされていた場合にはPSOアルゴリズムのような動きになる.

[ Pitch Adjustment ]

　Memory Consideration によってHM内から選択された値に対して, それを少し変えた値ai newを生成する. 確率par で行われる.

　決定変数が離散値の場合は, もしai new= xj(k) ならば ai new← xj(k+m) ただしm = {-1, 1} (一般に) となる. 決定変数が連続値の場合は, xi ← xi + ΔただしΔ= fw(i) ･ U(-1,1) (一般に) となる.

**4.3.5 Harmony Improvization + α**

　ハーモニーサーチアルゴリズムでは以下を利用することもできる.

[ Ensemble Consideration ]

　決定変数間の関係性を考慮する方法である.

　2つの決定変数間ai, ajで強い関係性が存在する場合, ハーモニーの値aiを次のように決定することができる. ai ← fn(aj) ただし max{ [Corr(ai,aj)]2 }

[ Novel Stochastic Derivative ]

　決定変数が離散値である場合には, ハーモニーサーチアルゴリズムには確率的導関数を利用することが可能である.

**4.3.6 Memory Update**

　ハーモニーの選択とHMの更新を行うステップである.

　生成されたハーモニーAnew がHM内で一番良くないハーモニーAworst より優れている場合, つまりf (Anew) > f(Aworst) である場合に, Aworst をHMから取り除いて Anew をHMに取り込む.

　HMの更新のさいにはHMの多様性のために, Anew とHM内の他のハーモニーの類似性を考慮することもある. 特に, HM内に複数存在するハーモニーはHMの早い段階での収束を防ぐために考慮される.

**4.3.7 Memory Update + α**

[ Accidentaling ]

　Memory Update のオプションの操作である.

　生成されたハーモニーAnew がHM内の任意のハーモニーよりも優れている場合, つまりf (Anew) > f(Abest) である場合に, さらにAnew を微小量だけ変化させることでハーモニーの更なる改善を試みる操作である. 値の変化のさせ方は Pitch Adjustment で示した方法と同様である.

　音符にシャープ(#)やフラット(♭)をつけてみるイメージである.

**4.3.8 Performing Termination**

　ハーモニーの改善を終了するステップである.

　ハーモニーサーチアルゴリズムは終了条件が満たされるまで, 新たなハーモニーが生成され続ける. 終了条件は, 繰り返し回数がMIに達した, 十分よいハーモニーが得られた, などである.

**4.3.9 Cadenza**

　ハーモニーサーチアルゴリズムの最後のステップである.

　HM内にある最高のハーモニーを返す.

**4.3.10 擬似コード**

以下の図4.3.10.1に, ハーモニーサーチアルゴリズムの擬似コードを示す.

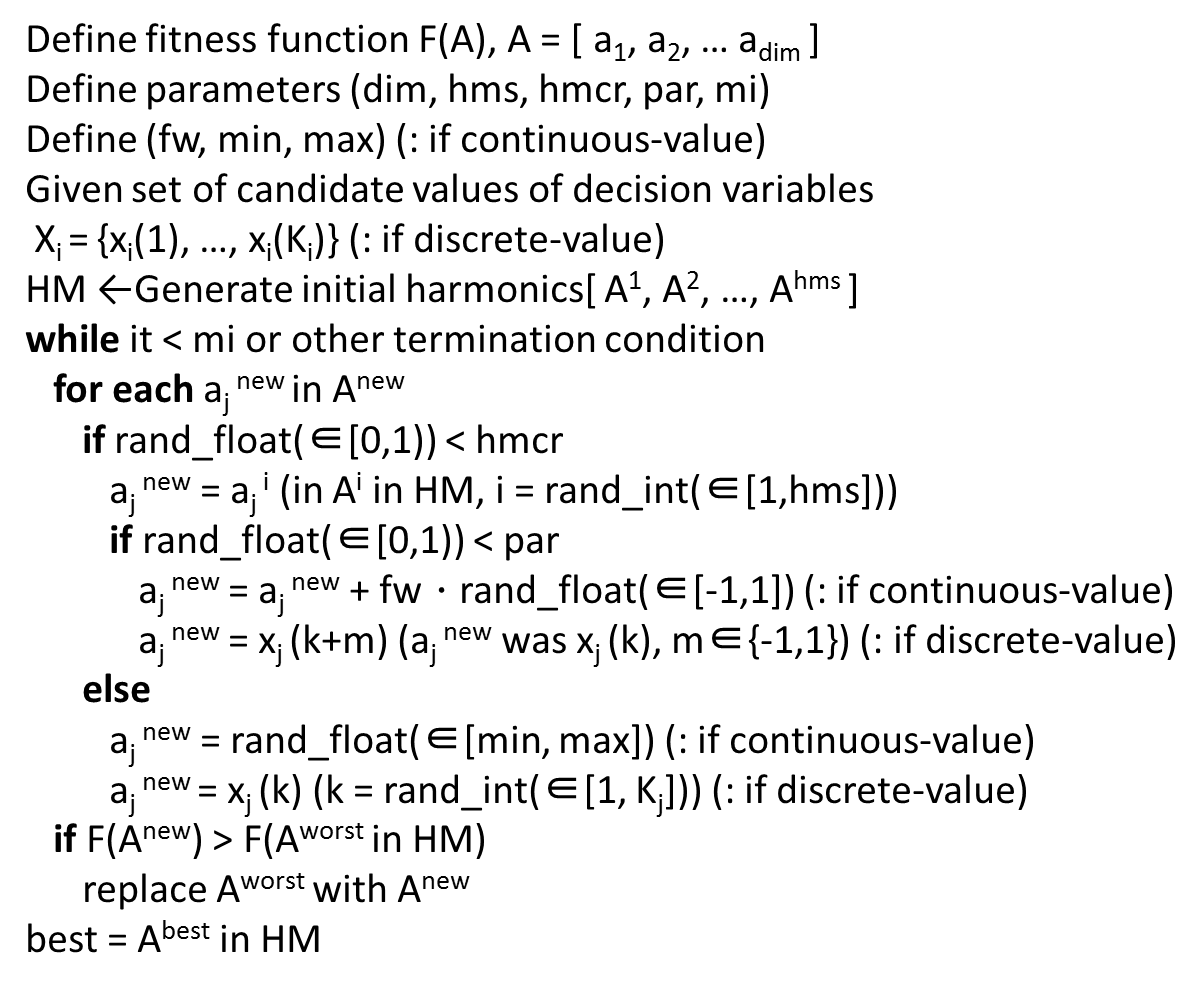


図4.3.10.1 ハーモニーサーチアルゴリズムの擬似コード

**5 ゲーム**

**5.1 ゲーム理論とは**

　ゲーム理論[game theory]は, 現実社会にみられる競争や協力の様々な形態を抽象化して数理的なモデルで捉え, 各自の行動原理を数学的に研究する理論である. ゲーム理論におけるゲーム[game]とは, 複数のプレイヤー[player]が, 自分の利益が最大となることを目的として, 相手の行動を予測しながら自分の行動を決めていくものである. このような抽象的なモデルの上では, 将棋やチェス, トランプ, ギャンブルなど遊戯的なゲームと本質的に変わらないことから, ゲーム理論という名が付けられた. ゲーム理論は, 意思決定の方法論の一種であり, 相互連関的意思決定理論[interactive decision theory]と呼ぶこともできる. (※引用 アブストラクトゲーム博物館 より)

**5.2 ゲーム木**

**5.2.1 ゲーム木とは**

　ゲーム木とは, ゲームの展開を木構造で表したものである.

　ゲーム木を利用してのちの展開を予想して先読みすることで, のちの自分が有利または不利にならないために最良の行動を選択することができる.

　ゲーム木では各プレイヤーが行動をとることで生まれる状況を繰り返し予想し, それを木として表現する. 根は現在の状況であり, 先読みした先の状況が葉に相当する. 葉には評価関数などを用いて状況の価値が与えられる. ゲーム木の深さは, 何手先まで先読みするかに対応する. ゲーム木の深さを深くすることで多くの先読みができるが, 計算時間が指数関数的に増加する. よって, ゲームの最初から考えうるすべての手を容易に先読みすることができない複雑な展開をするゲームにおいては, 数手先までしか先読みをすることができない. そのとき, k手先まで先読みをした結果の最良の手がk + 1手先まで先読みした結果の最良の手にたどり着くまでに必要な手である保証はないこと, また, 自分のゲーム木の予想外の展開を相手が取る可能性があることに注意する必要がある.

　以下の図5.2.1.1にゲーム木の例を示す.

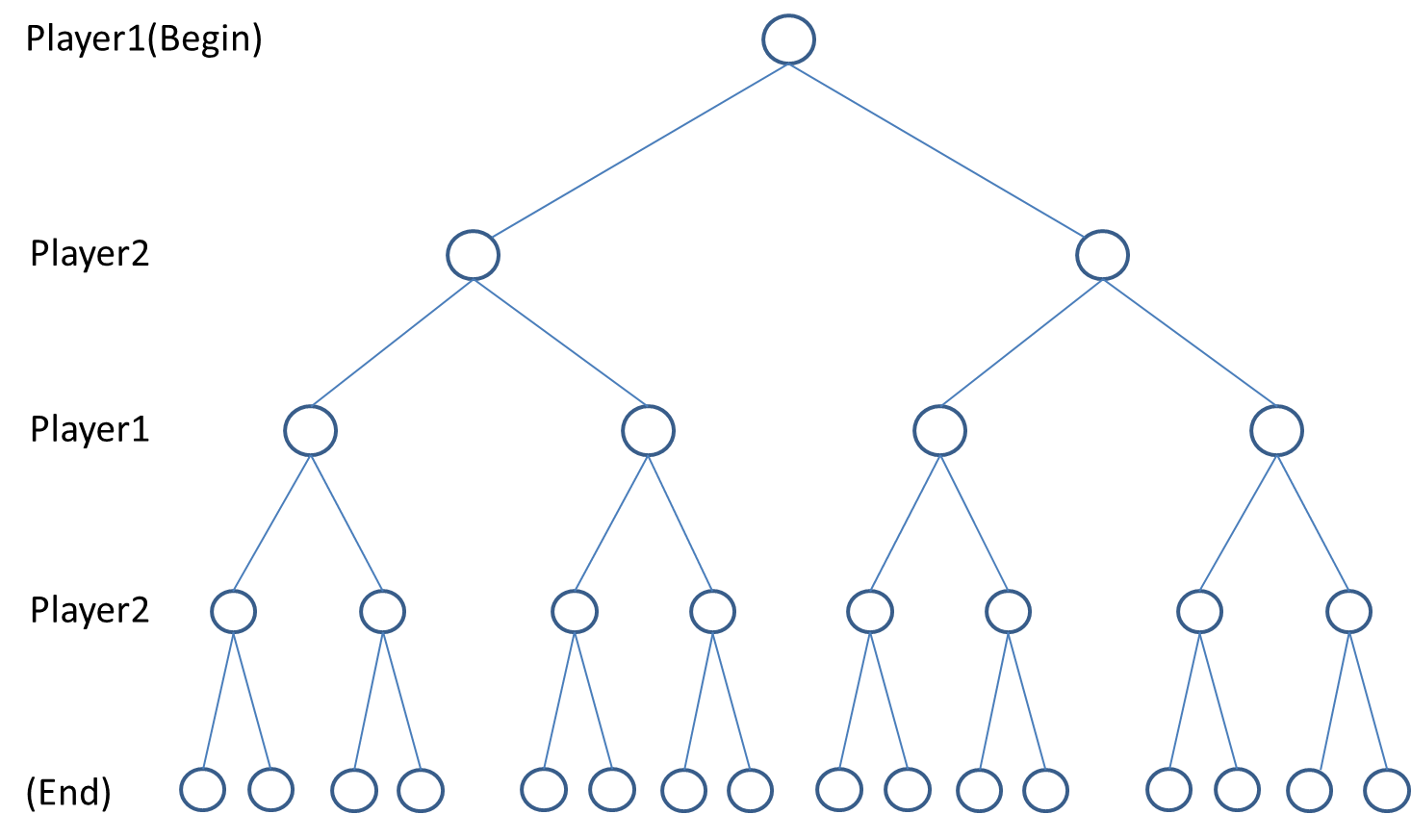


図5.2.1.1 ゲーム木

**5.2.2 Minimax法**

　Minimax法はゲーム木の探索法の一種である.

　ゲーム木の探索は最良の手を探すための方法である. 自分の利益を最大にする最良の手を決定するために先読みをし, 考えられるいくつかの最終的な状況のそれぞれの価値がわかっているとする. このとき, 自分にとって一番価値が高い状況にたどり着ける保証はない. なぜなら, 他のプレイヤーはその状況にならないように阻止することが考えられるからだ.

　Minimax法では各プレイヤーは自分の利益を最大にするために, 自分の損害が最小になるように行動すると仮定した場合の最良の手を求める. これは各プレイヤーにとって他のプレイヤーの利益が自分の不利益になる場合には, 他のプレイヤーの利益を最小にすることと同じである.

　以下の図5.2.2.1にMinimax法の例を示す.

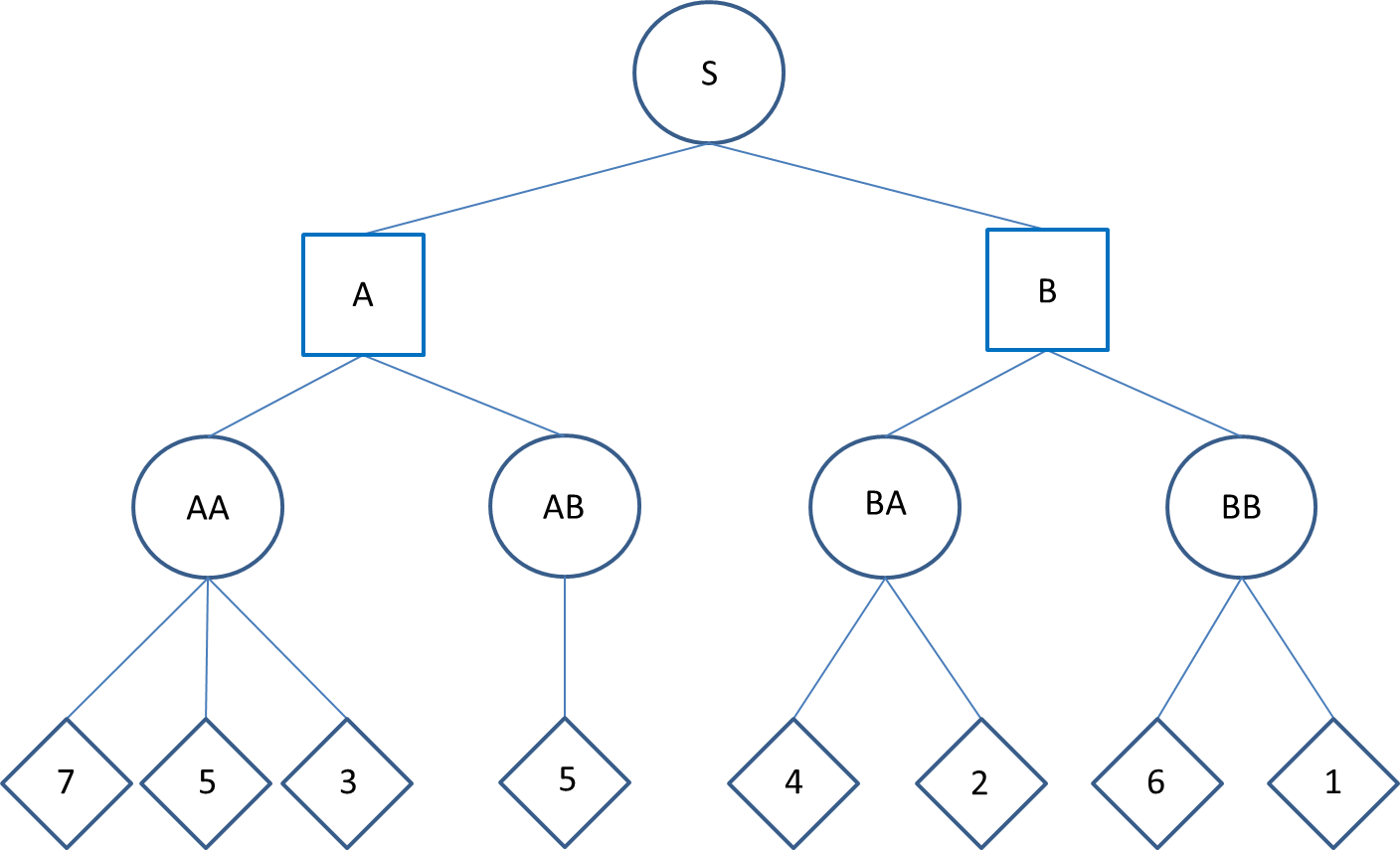


図5.2.2.1 Minimax法

　図5.2.2.1は, ある状況において自分が手番であり2手先まで予想した場合のゲーム木である. プレイヤーは自分と相手の2人であり, Sは現在の状況である. 各プレイヤーの行動を1手とすると, 1手先の状況はA, Bであり, 2手先の状況はAA, AB, BA, BBであり, 最終的な3手先の状況の自分にとっての価値(= 相手にとっての被害の大きさ)が数値としてわかっている状態である.

　このとき, 現在と2手先の状況において, 自分は次の手を選択する権利があり, 1手先の状況では相手に次の手を選択する権利がある. また, 最終的な状況の1つ前である状況である2手先の状況が, 次の手を選択することができるプレイヤーが存在する状況の中で最終的な状況に一番近い状況である.

　Minimax法では最終的な状況から現在の状況に向かって考えていく.

　まず, 2手先の状況について考える. 状況AAでのプレイヤーは自分である. 選択肢は状況の価値{7, 5, 3}の3つの状況であり, 自分にとって最良の選択は状況の価値7への手であるのでそれを選択する. よって状況AAの価値は7と考えられる. 同様にして, 状況AB, BA, BBの価値はそれぞれ5, 4, 6となる.

　次に, 最終的な状況に近いのは1手先の状況である. 状況Aでのプレイヤーは相手である. 選択肢は状況の価値{7, 5}の2つの状況であり, 相手にとって最良の選択は状況の価値5への手であるのでそれを選択する. よって状況Aの価値は5と考えられる. 同様にして, 状況Bの価値は4となる.

　最後に, 現在の状況について考える. 状況Sでのプレイヤーは当然自分である. 選択肢は状況の価値{5, 4}の2つの状況であり, 自分にとって最良の選択は状況の価値5への手であるのでそれを選択する.

　Minimax法ではこのようにして現在の状況における最良の手を求める.

**5.2.3 αβ法**

　αβ法はMinimax法に枝刈りを加えたものである.

　ゲーム木の探索の途中において, 探索済みの手から得られた各プレイヤーの被害の最小値を保存しておくことで, 既知の被害の量より被害が大きくなるとわかった場合にその先の探索を打ち切る方法をとる.

**6 オセロ**

**6.1 アブストラクトゲームとは**

　アブストラクトゲームとは, 特定のテーマを持たず, 使う駒や盤は抽象化されているゲームのことである. その中でも一般には, 対戦相手同士で情報が公開されおり, 偶然要素が無いものを指す. 対象範囲については様々な解釈が存在する.

　アブストラクトゲームの例として, 囲碁や五目並べ, オセロやチェッカーなどが挙げられる. アブストラクトゲームの多くはボードゲームである. 駒に個性が存在する将棋やチェスといって戦争ゲームや隠匿情報のないカードゲームも, 一般にはアブストラクトゲームとして扱う.

**6.2 オセロとは**

　オセロとは, 縦横8×8のマス目のある盤と白と黒の色が付いた石を使い2人で対戦するボードゲームである. リバーシともいう.

　オセロは, 二人零和完全情報確定非対称有限非協力交互進行収束ゲームに属する. オセロのプレイヤーの目的は勝利であり, 各局面においては優勢になるように行動をする. 単純なルールながら, ゲームの複雑度は人間がゲーム木の全展開を把握可能な程度を越えており, ゲーム探索空間は1060ほどである. (三目並べは103未満, チェッカーは1030, チェスは10120, 将棋は10220 囲碁は10360)

**6.3 ルール**

　オセロのルールは一般に以下の通りである.

1. 黒石と白石を盤の中央に２個ずつ, 黒石は右上がりで白石は右下がりにして交差するようにおき, 初期配置とする.

2. はじめに各プレイヤーは自分の色を決める. 一方が黒で, もう一方が白である.

3. 黒が先手で開始する.

4. 各プレイヤーは, 既に置いてある自分の石と新たに置く石で, 縦または横または斜めのいずれかの方向に相手の石をはさめるマスにのみ新たな石を置くことができる.

5. はさんだ石は自分の色の石にする.

6. 黒と白が交互に打ち進める.

7. 一方が打てる手が無い場合はパスとなり, 打てるところができるまでもう一方が続けて打つ.

8. 盤面が全部埋まるか, 両者とも相手の石をはさめなくなった場合に対局は終了する.

9. 終了時に石の数の多い方が勝者である.

　以下の図6.3.1～図6.3.3にオセロの局面の例を示す.

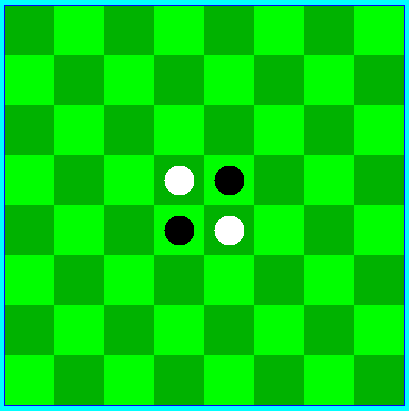


図6.3.1 初期状態

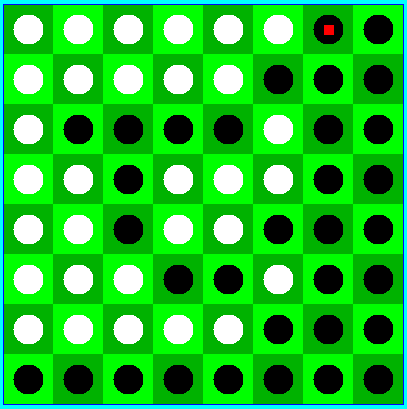


図6.3.2 終局

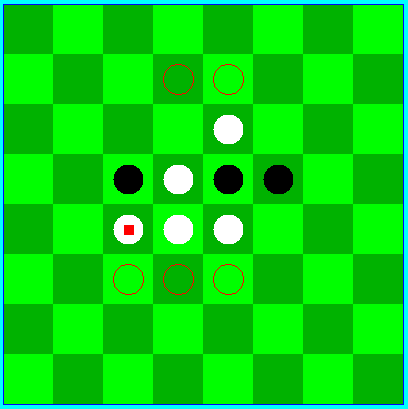


図6.3.3 局面の一例

　図6.3.1 はオセロの初期状態であり, 図6.3.2はオセロの終局の一例であり, 図6.3.3はオセロの対戦中の局面の一例である.　図6.3.3において, 中が空の円は次に黒のプレイヤーが石を置けるマスを示している

**6.4 戦術**

　オセロは様々な戦術が研究されている. オセロの定石や理論, 局面評価といったものが背景には存在する.

　以下はオセロにおいて一般的な常識の例である.

・隅を取ると有利になることが多い

・少なく取ると有利になることが多い

・相手に自分の石を囲ませると有利になることが多い

・自分の打てる選択肢を多くすると有利になることが多い

・序盤は定石を外れると凡手, 悪手になることが多い

　オセロの基本戦術と呼べるものも確立している. 以下がその例である.

・中割り

・引っ張り

・一石返し

　オセロの序盤には定石が存在する. 定石外はほぼ凡手または悪手である. 以下はオセロの定石の例である.

・兎定石

・虎定石

・ねこ定石

　オセロの盤の一番外側の行または列を辺といい, 勝敗を左右する部分である. 辺は形によって良し悪しが存在する. 以下はオセロの特徴のある辺の形の例である.

・山

・ウイング

　オセロにはいくつかの理論が見つけられている. 理論を理解することによって手の質を向上させられる. 以下はオセロの理論の例である.

・着手手の法則

・開放度理論

・偶数理論

　オセロの局面評価とは盤の状態が自分にとってどれくらい有利なのかを示すものである. 有利不利の判断は試合において重要な要素である. 以下は局面評価の例である.

・石の位置による評価

・パターンに基づく評価

**6.5 AI**

　今日では様々なアブストラクトゲームのAIの研究がなされている. 現在のアブストラクトゲームをプレーするコンピュータの取る戦略は, その局面に至る経緯は考慮せず, 評価関数によってそれぞれの局面を静的に判断して着手を決定する. そこで, コンピュータが次の最良の手を探索するさいに, その効果を高める様々な研究がなされている. ゲーム木の利用, ゲーム木の探索の深度延長の判断, 評価関数の改良, 探索アルゴリズムの改良, 定石データベースという知識の充実, など様々である.

　オセロAIの研究もアブストラクトゲームAIの例外ではない. 研究の産物として, 最強のオセロソフトとよばれる「ロジステロ」が誕生している. ロジステロは1997年8月には世界チャンピオンとの対戦で見事に人間に打ち勝った. 最大1秒間に48万手を読むロジステロは序盤の20手を過去の膨大な棋譜の中から定石として記憶して悪手を打たず, 最後の26手を6分で完璧に読んで勝敗を判断できていた. 2002年5月の元日本チャンピオンとの対戦でも勝利を収めた. このときのロジステロの能力は既に終盤で毎秒180万局面を読むレベルである. オセロにおいて, AIは既に人間の能力を超えているといえるだろう.

**7 実験**

**7.1 実験概要**

**7.1.1 目的**

　この実験の目的は, 「大きな複雑度を持つ問題に対し専門知識をベースとせずに, 機械学習によって状況評価を学習すること」を, オセロAIの状況評価の役割を持つニューラルネットワークとその重みを改善するハーモニーサーチアルゴリズムを組み合わせた機械学習を行い, 観察することである.

**7.1.2 主な設定**

　この実験の主な設定は以下の通りで,

・ハーモニーメモリー(HM) 内の各ハーモニーは, ニューラルネットを重みに相当する値を持つ.

・ニューラルネットワークは, 局面状況のよさを算出する局面評価関数の役割を担う.

・各ハーモニーの値を元に生成されたニューラルネットワークに従うAI間のオセロの対戦成績でハーモニーを評価する.

・ハーモニーサーチアルゴリズムでハーモニーを改善していく.

である.

**7.2 実験方法**

**7.2.1 ハーモニーサーチアルゴリズム**

[ 役割 ]

　実験におけるハーモニーサーチアルゴリズムの役割は以下の通りで,

・ニューラルネットワークの重みの改善

である

[ パラメータ設定 ]

　実験におけるハーモニーサーチアルゴリズムのパラメータは以下の通りで,

・MI (maximum improvisation) : 1000

・hms (harmony memory size) : 128

・hmcr (harmony memory considering rate) : 0.95

・par (pitch adjusting rate) : 0.3

・fw (fret width, formerly bandwidth) : 0.005

・|Allow Range| ≤ 1

である. これらのパラメータの値は典型値を元にし, 実験を繰り返して経験的に調整した値である.

[ 目的関数 ]

　実験におけるハーモニーサーチアルゴリズムの目的関数は

・試合に対する得点( 勝ち:+10, 負け:+0, 引き分け:+1 )

で定義している. これは

・勝つことが良いこと

・負けよりは引き分けが良い

ということを示している. 各ハーモニーは適応度の最大化を目的とする.

　ただし, 実際には各ハーモニーの値を利用したそれぞれのAI間でオセロの対戦を, 組合せを変えて繰り返しすることでハーモニーの適応度を計算している. 適応度の計算方法は以下の通りであり,

1. 各世代の第一試合の前にFを0で初期化する

2. 試合の勝敗から( 勝ち:+10, 負け:+0, 引き分け:+1 )の点数を得る

3. 試合数だけ2を繰り返す

4. 点数を試合数で割る

である. よって, 目的関数は

・HM内での相対的強さ

を求めるためのものとなっている.

　この目的関数によって求められる値は一般的な目的関数の場合とは違い, 相対的な値を示していることに注意する必要がある. 加えて, この相対的な値は学習におけるある世代での相対的な値であることにも注意する必要がある. これらについて, 以下に述べる.

　一般に, 複雑なゲームにおける“ゲームの強さ”を測る絶対的な指標はなく, この“ゲームの強さ”は正確に絶対値で表すことや単純な式によって算出することはできない. そのためにゲームAIにおいて改善度の確認は, 実際に対戦を行うなどして強さの目安となるものを測定する必要がある. ただし, 特定の対戦相手または戦術に対してではなく, あらゆる戦術に対して強い戦術へと改善しなければいけない. これは学習において, 本質的にゲームの戦術を学習しなければ学習をしたと言うことはできない, ということである.

　個体群ベースのメタヒューリスティックアルゴリズムにおける目的関数の本来の役割は, 問題に対する適応度を算出することである. 本実験においてそれはニューラルネットワークの重みの改善度であり, オセロへの本質的な学習度であり, オセロおける”ゲームの強さ”である. 前述した理由から, 本実験では個体間の対戦によって得られる相対的な”ゲームの強さ”を測定し, 適応度としている.

　また, 個体群ベースのメタヒューリスティックアルゴリズムにおいて各世代の個体群の中身は異なっている. よって, 本実験の適応度は各世代において各ハーモニーに対して測定しており, 学習におけるある世代で算出した相対的な強さとなっている.

[ アルゴリズム ]

　本実験では, 基本的に一般的なハーモニーサーチアルゴリズムと同じアルゴリズムを用いているが, 毎世代hmsと同数の新しいハーモニーを生成している点が異なっている. これは, 効率の良いオセロの対戦方法を取るためと共に, ハーモニーの進化の効率化を狙いとしている.

**7.2.2 ニューラルネットワーク**

[ 役割 ]

　実験におけるハーモニーサーチアルゴリズムの役割は以下の通りで,

・盤の情報からその局面の価値を出力

である. これにより, ニューラルネットワークをもつプレイヤーは, 次の手の候補の内の最善手を見つけることができる.

[ パラメータ設定 ]

　実験におけるニューラルネットワークのパラメータは以下の通りで,

・バイアスの数 bias size : 1 (bias = -1)

・盤の各マスの状態以外の盤情報の数information size : 4

・入力層のノード数 input layer size : 68

・中間層1のノード数 hidden layer size : 208

・中間層2のノード数 hidden layer size2 : 60

・中間層3のノード数 hidden layer size3 : 20

・出力層のノード数 output layer size : 1

である. ただし, input layer size に bias size は含んでいない.

[ 活性化関数 ]

　実験において使用した活性化関数は以下の通りで,

・シグモイド関数 f(x) = , (α = 1)

である.

[ 入力 ]

　ニューラルネットワークへの入力情報は, 盤の各マスの状態( 自分の石:1, 相手の石: -1, 石なし: 0) の情報を64つと, 盤の各マスの状態以外の盤情報が4つである.

　入力層の各ノードは各情報に対応しており, 入力層のノード数はそれらの和68である.

盤の各マスの状態以外の盤情報には以下を含んでいる.

・盤上の自分の石の数

・盤上の相手の石の数

・盤上で自分が石を置けるマスの数

・盤上の相手が石を置けるマスの数

これらはオセロの一般的な常識レベルの情報である. よって, 本実験ではこれは専門知識をベースとしていないとみなしている.

[ 中間層 ]

　入力層と中間層1の間の重みの一部のみ0で固定しており, 特徴抽出のフィルタの役割をしている. それにより, 中間層1の各ノードがそれぞれ, 盤上のn×nの各ブロック(1×1のブロック64つ, …, k×kのブロックが(9 - k)2つ, …, 8×8のブロック1つ)に対応している. この特徴抽出の仕方は機械的であり, オセロをする人間が一般的に行う盤を局所的または大域的に見る動作に相当する. よって, 本実験ではこれは専門知識をベースとしていないとみなしている.

　中間層1のノード数は, 各ブロックに対応する各ノードの数204(= )と盤の各マスの状態以外の情報に対応している各ノードの数4の和208である.

　中間層の数, 中間層2のノード数, 中間層3のノード数は, 実験を繰り返して経験的に調整した値である.

[ 出力 ]

　局面の価値を出力するノードのみからなり, ノード数は1である.

以下の図7.2.2.1にn×nのブロックの取り方の例として, 4×4のブロックの一例と6×6のブロックの一例を示す.

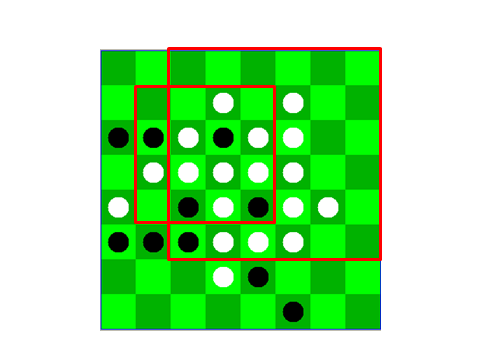


図7.2.2.1 n×nブロック

以下の図7.2.2.2に本実験のニューラルネットワークの構造を示す.

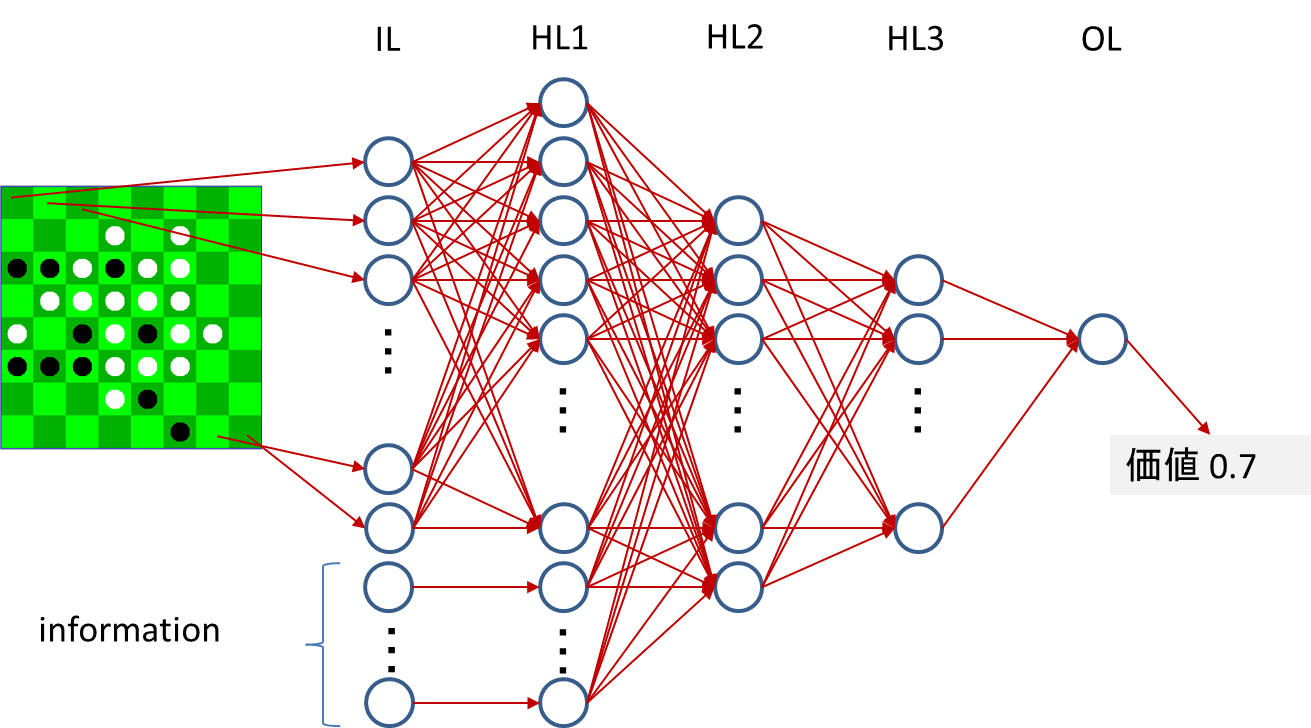


図7.2.2.2本実験のニューラルネットワーク

**7.2.3 AI**

[ 役割 ]

　実験におけるAIの役割は以下の通りで,

・ニューラルネットワークの学習度の確認をするために, あるハーモニーの決定変数に従い生成された重みを持つニューラルネットワークに従った戦術でオセロの対戦を行う

である.

[ ゲーム木探索 ]

　ゲーム木の探索として深さ2(次の次の自分の手まで)のαβ探索法を, 学習時と学習後の対戦時の両方で用いている. 葉における評価は与えられたニューラルネットワークに従う.

　ただし, ある局面を評価するさいにオセロの対称性を利用し, 元々の局面を90°回転させたもの, 180°回転させたもの, 270°回転させたものも考慮することで, 計4つの局面を評価値の計算に利用している. これはオセロの局面評価の安定性の向上とともに, 学習の促進を狙いとしている. 例えば, 盤の左上の角の隣と盤の右上の角の隣と盤の左下の角の隣には石を置かないが盤の右下の角の隣には石を置いてしまう癖がある場合, これは右下の角の隣に石を置かないように矯正される. この方法はニューラルネットワークによる局面評価の回数を4倍にするため, 実行時間の急上昇を招くことに注意する必要がある. 同様にして, 局面の反転を行うことで計8つの局面を評価値の計算に利用することができるが, 実行時間の観点から本実験では行っていない.

**7.2.4 対戦(Game)**

[ 役割 ]

　実験における対戦の役割は以下の通りで,

・各ハーモニーの適応度を計算する

である. これにより優れた個体の判別をすることができる.

[ パラメータ設定 ]

　実験における学習のためのオセロの対戦のパラメータは以下の通りで,

・対戦相手の数 MaxOppNum : 11

・対戦相手との試合数 MaxGametimes : 1

である. MaxOppNumはsqrt(hms)を切り捨てた値としている.

　これらの値は, 実験を繰り返して経験的に調整した値である. 過多な対戦相手の数と対戦相手との試合数は実行時間の無駄な増加を招くだけでなく, 個体の多様性の保持を阻害し, 過学習を引き起こす可能性を増加させる.

[ 対戦方法 ]

　先攻後攻はランダムである.

　対戦はハーモニーベースのニューラルネットワークベースのAI間で行う. つまり, 個体間の対戦である. 生成された各新しいハーモニーをベースとしたそれぞれのAIが, 指定した対戦相手の数だけHM内の各旧ハーモニーをベースとしたそれぞれのAIと, 指定した試合数だけ対戦を行う. これにより, 新旧問わず全個体がそれぞれ(対戦相手の数･対戦相手)回の試合を行わせることができ, 各個体を同条件の下で評価することにより, 優れた個体の選択をすることができる.

**7.2.5 観測者(Observer)**

　観測者とは, 学習とは別に用意したAIである. あらかじめ固定された評価関数を持つ.

[ 役割 ]

　実験における観測者の役割は以下の通りで,

・ハーモニーと対戦させ, 改善度をおおざっぱに測る

である.

　目的関数によって求められた各ハーモニーのHM内での相対的な改善度だけでは, HM内の全ハーモニーの学習が停滞している可能性が拭えない. よって, 学習が継続されているかの確認をする指標として観測者との対戦をし, 改善度を測定する. ただし, 観測者との対戦成績も”ゲームの強さ”を測る絶対的な指標とはならないため, あくまで一つの指標である.

　本実験では以下の2種類の観測者を用意した. 両者ともαβ探索は組み込まれていない.

・AI (No.1) : 自分の石が一番多くなる手を最良の手とするもの

・AI (No.2) : 盤の各マスの価値を知識として持ち, 石の位置による局面評価の値が一番大きくなる手を最良の手とするもの

**7.3 実験結果**

　学習後にきちんとオセロの戦術の学習がなされているかを確認するため, いくつかの対戦を行った. ハーモニーサーチアルゴリズムとニューラルネットワークと個体間の対戦の繰り返しにより学習を行う過程で, 観測者との対戦成績のデータを記録した. また, 学習の終了後, 一般人として36名の大学生に学習済AIとの対戦の協力をしてもらい, その対戦成績を記録した.

**7.3.1 対観測者**

　観測者との対戦の結果を示す. ここではαβ探索なしの学習済AIの対戦成績も記録した.

　観測者との対戦は, 学習の過程の50世代ごとに, 適応度の上位3個体それぞれが2種類の観測者と50戦ずつ行った. 上位3個体の勝率の平均を結果として記録した.

　表7.3.1.1と図7.3.1.1は, αβ探索なしの学習済AIと2種類の観測者との対戦における,学習済AIの勝率を, 学習済AIが先攻の場合と後攻の場合で分けたものを示している.

　表7.3.1.2と図7.3.2.1は, αβ探索ありの学習済AIと2種類の観測者との対戦における学習済AIの勝率を, 学習済AIが先攻の場合と後攻の場合で分けたものを示している.

表7.3.1.1 対観測者（学習済AIは対戦時にαβ探索なし）の勝率(%)



表7.3.1.2 対観測者の勝率(%)



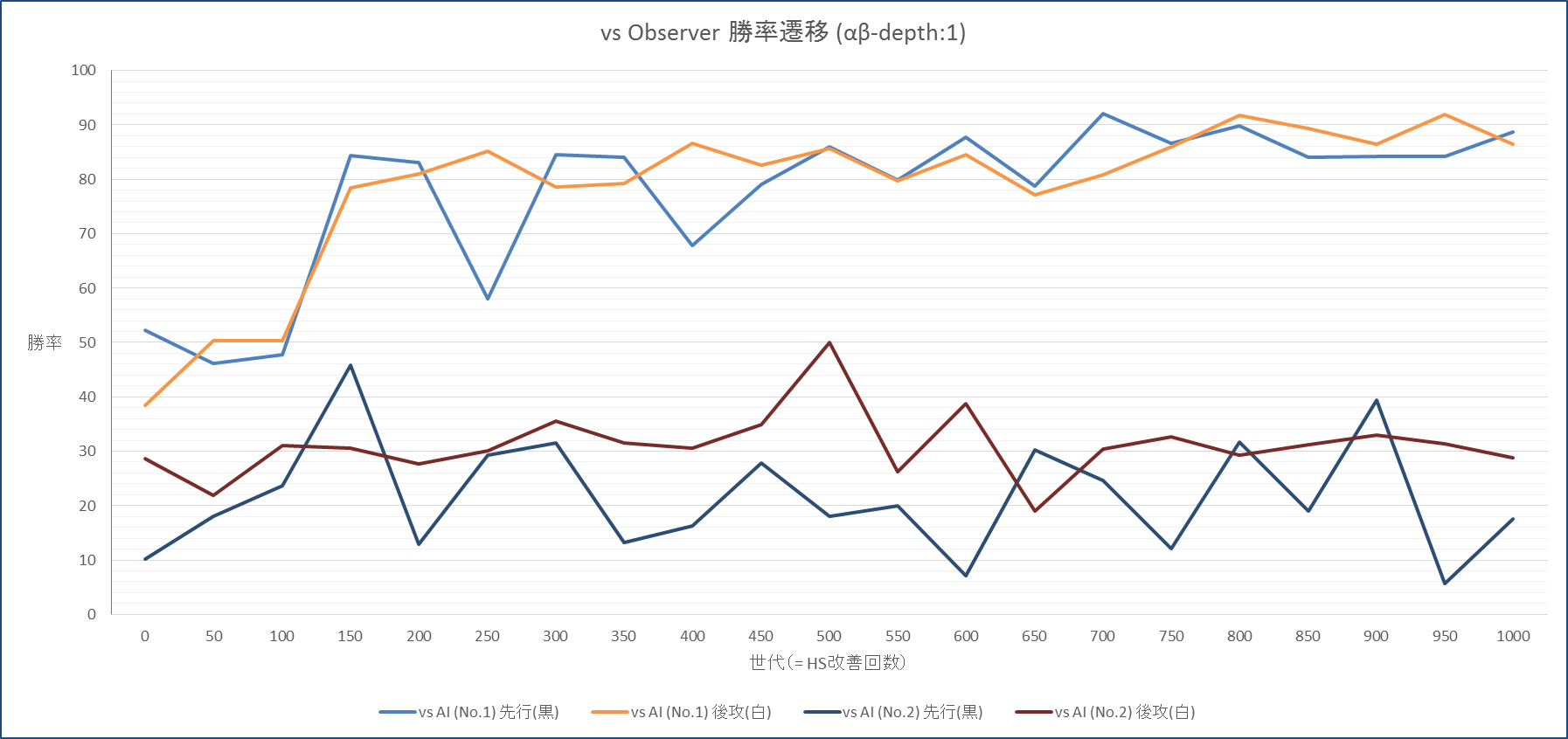


図7.3.1.1 対観測者（探索の深さ1）の勝率の遷移

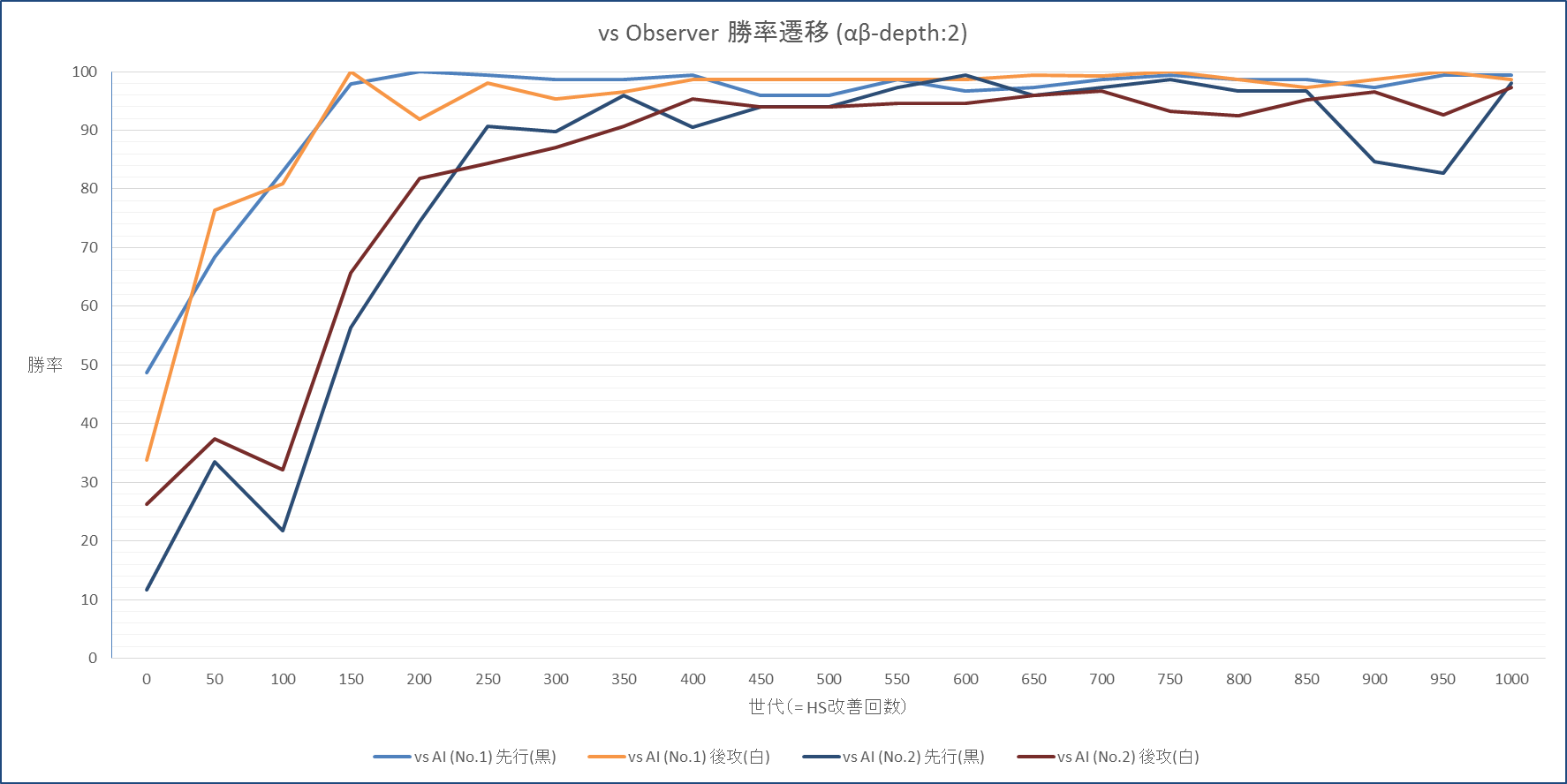


図7.3.1.2 対観測者（探索の深さ2））の勝率の遷移

**7.3.2 対人**

　人との対戦の結果を示す.

　人との対戦は, 適応度の最も高い個体のみが36人に対して各約3戦ずつ, 計126戦行った. 対戦相手の人は無作為に選ばれた大学生である.

　表7.3.2.1は学習済AIの対人試合の全ての集計データを示している.

　表7.3.2.2は学習済AIの対人試合のうち, オセロの経験がないまたは少ないと答えた人を除いた集計データを示している.

　表7.3.2.3は学習済AIの対人試合の試合結果を示している.

表7.3.2.1 対人試合結果の集計



表7.3.2.2 対人（経験なし, 経験少ない を除く）試合結果の集計



表7.3.2.3 対人試合の試合結果









**7.4 考察**

　対観測者の対戦結果について考察する.

　αβ探索の有無や先攻後攻や対戦相手に関わらず, 学習済AIの勝率は終始上下している. これは, 同じ相手に対してであっても, 新たな戦術の学習の結果が以前の戦術に完全に勝る保証はないことを示しているだろう. 学習が進むことで, 以前打ち負かしていた戦術に対して弱くなる場合があるということだ.

　αβ探索の有無や先攻後攻に関わらず, ある相手に対して勝率が上昇しても他の相手に対して勝率が下降している過程が存在する. これは, ある相手の戦術に強い戦術を学習したとしても, その戦術は他の相手の戦術に通用するとは限らないことを示しているだろう. 例えば, 200世代目から250世代目のαβ探索なしの先攻の試合において, AI(No.2)に対しての勝率が12.84%から29.33%に上昇しているのに対し, AI(No.1)に対しての勝率は83.0%から58.0%に下降している.

　同じ相手に対しても, αβ探索の有無で勝率の変化が異なる過程が存在する. これは, 先読みの程度によって学習した戦術の価値が異なることを示しているだろう. 例えば, 300世代目から350世代目のAI(No.2)との試合において, αβ探索なしの対戦成績は先攻が31.59%から13.25%で後攻が35.57%から31.54%に下降しているが, αβ探索ありの対戦成績は先攻が89.81%から96で後攻が86.99%から90.67%に上昇している.

　全体的には, αβ探索ありの場合の対戦成績に比べてαβ探索なしの場合の対戦成績の改善度は悪い. これは, 単純にαβ探索がAIの強さを上げるために優秀であることが考えられるが, それ以外にも, 学習時の個体間の対戦においてもαβ探索ありで行っていたことから, 学習する戦術がαβ探索ありの, 更にいえば2手先読みをした場合に効率的な戦術に進化していったことが考えられる. 結果, 対戦時の先読みの程度によって学習した戦術の価値が異なるということだ.

　また, 全体的に先攻と後攻の戦術の学習の改善度に差は特になく, 学習のしやすさに差はないように思われる.

　また, 全体的にどの場合においても450世代あたりから勝率の変化があまりなく停滞している. これは観測者が改善度を観測できなくなったというより, 本実験の実験方法で学習することのできる戦術の限界がきているように思われる. 実際に600世代目の適応度の最も高い個体との対戦や1000世代目の適応度の最も高い個体との対戦を行っても強さに大きな差は感じられない. “ゲームの強さ”のピークが最終世代である保証はないといえる. ただし, 1000世代目においてもHM内には弱い個体も存在しており, HM内の多様性は保持されていて, 過学習はなされていない. つまり, “ゲームの強さ”の観点では同等であるが戦術としては新しいものを学習し続けていることが考えられる.

　もう一つの実験結果である, 人との対戦の結果について考察する.

　多くの人との対戦から, 学習済AIの”ゲームの強さ”を測るために十分多くの指標が得られた. 結果として, 全試合の勝率は80.16%にも上り, 経験が少ない人との対戦を除いても72.15%に上った. これは, 十分オセロの戦術を学習したといえる結果だ.

　最後に学習済AIのオセロの戦術的課題について考察する.

　図7.4.1～図7.4.3は学習済AIとの対戦中で課題が見えた局面の一例である.

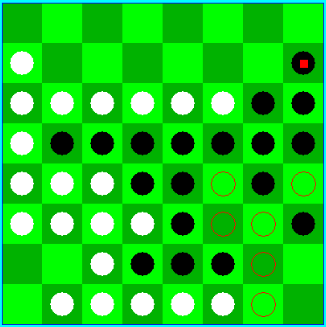


図7.4.1 局面A

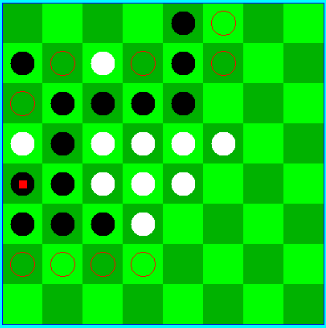


図7.4.2 局面B

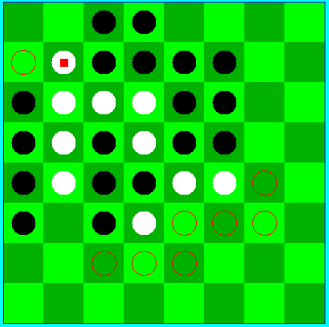


図7.4.3 局面C

　図7.4.1ではプレイヤーは白でAIは黒であり, 図7.4.2ではプレイヤーは白でAIは黒であり, 図7.4.3ではプレイヤーは黒, AIは白である. 図7.4.1～図7.4.3で, ある石の上にある赤い四角はAIの直前手を示している.

　図7.4.1ではプレイヤーは次の手で上から5列目, 左から8列目に手を置くことで, 右上の角を取れてしまう. 図7.4.2ではプレイヤーは上から3列目, 左から1列目に手を置くことで左上の角が取れてしまう. 図7.4.3では, すぐに角を取られることはないが後々に左上の角を取られることが予想される. いわゆる角に隣接する3マスに序盤から置いてしまうという悪手である.

　このような悪手を置いてしまう原因として考えられることは, αβ探索法のゲーム木の探索の深さを2(次の次の自分の手まで)で, 学習時と学習後の対戦時の両方に用いていることである. 上記のどの場合も, ゲーム木の探索の深さ2より1手先まで先読みすることで, 相手は角を取れる状況となることがわかる. つまり, 先読みが足りていないということである. これは, “ゲームの強さ”を求めるためには, 状況評価の学習だけではなく, ゲーム木の探索が重要であるとうことだ.

　しかし実験時の対戦において, オセロの戦術知識におけるウイングの形など, 後の展開が悪くなる典型的なわかりやすいパターンは避ける行動をとっていることが見られた. これは, ゲーム木の探索の深さが足りていない場合でも, 後の展開が悪くなる局面だということを対戦で負けることにより学習している場合があることがわかる. よって, 上記のような悪手を学習できなかった原因として, ゲーム木の探索の深さが足りていないという原因以外に, 学習過程における個体間の対戦において上記の状況は, 後の展開が悪くなる状況に近い状況の中でも出現頻度の低かった状況であることが考えられる. さらに, 学習過程のある世代において出現頻度が高い状況を一旦学習したとしても, 後の世代でその出現頻度が低くなった場合には, 学習したその状況を忘れてしまう可能性があることも考えられる. つまり, 複雑な問題に対して個体間の対戦によって問題の本質の学習を試みた場合に, その問題の中で稀少な場合にも対応した学習は容易ではなく, 最終的に学習されない可能性があるということである.

**8 まとめ**

　本研究の目的は, 「大きな複雑度を持つ問題に対し専門知識をベースとせずに, 機械学習によって状況評価を学習すること」を研究することであり, 題材としてオセロを用いた.

　結果として, ハーモニーサーチアルゴリズムとニューラルネットワークの組み合わせにより, 専門知識をベースとせず, 特別な対戦相手ではなく個体間の対戦経験だけを経て, 改善と後退を繰り返して複雑な状況評価の学習をし, オセロの戦術を独自に理解したといえる勝率を残した. これは, ある問題に対して限られた選択肢の中から, 問題解決のための行動の意思決定の方法を, 機械学習を利用してコンピュータが独自に見つけ出すことができたことに他ならない.

　本研究の課題としては, 各パラメータの調整やアルゴリズムの組み合わせや改良による学習の効率化, 探索アルゴリズムの高速化や状況評価の軽量化による学習時間の短縮, 改善度の測定精度向上による学習精度の向上, 学習の入力データの特徴抽出の自動化や最適化, などがあげられる.

　今回の学習方法をベースとし, オセロのみならず他分野への応用を行っていきたい.

**参考文献**

[1] Zong Woo Geem, “State-of-the-Art in the Structure of Harmony Search Algorithm”, Recent Advances In Harmony Search Algorithm, Studies in Computational Intelligence, vol. 270, 2010, pp. 1-10.

[2] Zong Woo Geem , “Novel Derivative of Harmony Search Algorithm for Discrete Design Variables”, Applied Mathematics and Computation , 2008, pp. 1-9.

[3] Siang Y. Chong, Mei K. Tan, and Jonathon D. White, “Observing the Evolution of Neural Networks Learning to Play the Game of Othello”, IEEE Transactions on Evolutionary Computation, vol.9, no.3, June 2005, pp. 240-251.

[4] 和多田作一郎, AIの基礎を知る辞典, 実務教育出版, 1986/10

[5] J.デイ, ニューラルネットワークアーキテクチャ入門, 森北出版, 1992/4

[6] 岡田 章, ゲーム理論・入門--人間社会の理解のために, 有斐閣, 2008/8/23

[7] 機械学習の理論と実践 (株式会社Preferred Infrastructure岡野原 大輔

(http://sacsis.hpcc.jp/2013/files/sacsis2013\_ml\_okanohara.pdf アクセス 2014/12/23)

[8] Metaheuristic Optimization - Scholarpedia

( http://www.scholarpedia.org/article/Metaheuristic\_Optimization, アクセス 2014/1/4 )

[9] アブストラクトゲーム博物館 ( http://www.nakajim.net/, アクセス 2015/1/8 )

[10] オセロプログラムと人間はどっちが強いのか？ロジステロとの戦い( http://uguisu.skr.jp/othello/7-2.html, アクセス 2015/1/8 )

**謝辞**

　本研究を行うにあたって研究をサポートしてくださったゴンサルベス・タッド准教授, 研究室の学生や実験に協力してくださった学生の方々, そして参考文献の著者および携わった方々に心から感謝しお礼を申し上げます. ありがとうございます.