■S3 群（脳・知能・人間）－ 3 編（人工知能と学習）

5 章 ゲームと人工知能

（章主任：小谷善行）［2008 年 12 月 受領］

■概説■

複雑な機械を作れるようになると，人々は知能を持つ機械を作れないか夢想した．コンピュータという機械が生まれるとその夢想は手の届きそうな夢想になる．そしてゲームを自動的にプレイする問題がその手近な目標になった．このように，ゲームプレイはコンピュータ成立当時から一つの興味の対象かつ目標物となっている．ゲームは人工知能の実蝿（ショウジョウバエ）である，という言い方をする．遺伝や進化などの生物学での題材としての実蝿の役割を，ゲームが人工知能（古くはコンピュータ自体）に対して果たしてきたと言える．

本章ではゲームをプレイするシステムの研究について，学術的成果を中心に紹介する．アカデミックに成果があがり，また理論化されているのが，将棋，囲碁，オセロといったゲームの範疇であり，確定的完全情報 2 人零和ゲームと呼ばれる．こうしたものを中心に節を立てて説明する．

5-1 節では，基礎概念の解説とともに，こうした研究のスタートとなったチェス（Chess，西洋将棋）について歴史的にとらえて説明する．5-2 節では，コンピュータ将棋の研究と，ゲーム木探索について掘り下げる．ゲーム木探索は思考ゲーム技法の最も中心となるものである．詰将棋解法アルゴリズムは 1990 年代から飛躍的に進歩し，証明数探索と呼ばれる AND/OR 木探索技法でほぼ完成されたものになった．これを 5-3 節で解説する．5-4 節では，囲碁の思考アルゴリズムを説明する．近年になりモンテカルロ学習を用いたコンピュータ囲碁が劇的に進歩してきており，その技法を述べる．5-5 節では，強化学習の一つの方法である TD（Temporal Difference）法について解説する．これは 1990 年代にバックギャモンに対する成功が報告され，その後，いろいろなゲームに応用されるようになった．

【本章の構成】

本章では以下について解説する．

■S3 群－3 編－5 章

5-1 チェスと思考ゲーム研究の歴史

（執筆者：小谷善行）［2008 年 12 月 受領］

チェスは将棋と起源を同じくするゲームである．チェスを機械に行わせることは昔から人々の好奇心の対象であった．19 世紀に既に，コンピュータの始祖というべきチャールズ・バベッジは，解析エンジンと呼ばれる最初のコンピュータを歯車による機械的機構で作ろうとした．その彼もコンピュータチェスの定式化を考えていたようである．彼は先読みとミニマックスの戦略さえも思いついていた可能性もある．

コンピュータの歴史のなかで，フォンノイマン型コンピュータが確立した 1950 年前後になると，情報理論で有名なクロード・シャノンと，コンピュータサイエンスの始祖とも言えるアラン・チューリングは，それぞれ独立に，チェスを指させるための手順（すなわちアルゴリズム）を明示的に示した．そのなかには，ミニマックス戦略や評価関数も含まれる．シャノンもかなり精密な定式化をしていた．駒の損得だけでなく駒の位置関係を見ること，選択的探索やのちに静けさ探索と呼ばれる考え方に到達していた．

ゲームにおけるミニマックス戦略とは，再帰的に書くと次のようなものである．今，指そうとしている局面から一定手数先（探索深さという）まですべて先読みし，その先で評価関数の評価を行う．現在局面の価値とは，

・探索深さに到達しているときには，評価関数を用いて計算した局面の評価値
・そうでないとき
　それが自分の手番なら，可能な着手をそれぞれ指した後の局面価値の最大値
　それが相手の手番なら，可能な着手をそれぞれ指した後の局面価値の最小値

ということであり，現在局面で指すべき着手は上記最大値を与える着手である．これがコンピュータのゲームプレイの核になる．これが 20 世紀半ばにもう見つかっていた．

チェスアルゴリズムが最初に動いたのは 1951 年で，チューリングが作ったアルゴリズムと人間の対戦が行われている．アルゴリズムはコンピュータ上ではなく，紙の上のシミュレーションであった．一方はチューリングのもの 1956 年にはロスアラモス科学研究所のグループが初めてコンピュータ上で動作するチェス（ただし 6×6 のミニチェス）が生まれた．1958 年には，バーンスタインが通常のチェスのプログラムを作った．ソビエト連邦でもほぼ同時期にチェスのプログラムが作られた. 1960 年前後には, LISP で有名なマッカーシーら MIT の人々が，αβ法の考え方を発見している．αβ法，別名αβ枝刈（Pruning）は，ミニマックス戦略と同じ結果を出力するが，非常にそれを高速化するアルゴリズムであり，ほかのゲームの場合も，ミニマックスを求める際に必ず使われる．

αβ法を簡単に説明する．これは計算する評価値の下限値αと上限値βを引数として持っている．動作はミニマックス戦略と同じように行われるが，可能な着手をそれぞれ指した後の局面価値の最大値を計算するにあたり，ループ中で途中までの最大値がβを超えたらそこでループを終了する．なお，再帰呼び出しする際にマイナスαを次の上限値βにする（計算は通常，計算は手番側から見た値を計算する，ネガマックス方式で書く）．着手を決める際に最初にこれを呼び出すときには，αを －∞，βを＋∞にして呼び出す．しかし，再帰呼び出しの中ではαとβを使って必要な値の範囲だけしか計算しない．

αβ法は，ミニマックス法のプログラムと比べて，2，3 行増しになるに過ぎない．しかし，その効果は著しい．1975 年にはクヌースとムーアがαβ法の効果を理論的に示している．だいたい深さが 1/2 になる効果を持つ．例えば，枝分かれが 10 ずつある深さ 8 のゲーム木を考えると，到達する末端は 1 億局面である．これが深さ 4～5 になったとすると末端は 1 万～10 万局面ということになる．

1960 年代には，チェスの終盤データベースというものも考えられている．レトログレードアナリシスといい，キングがない（取られた）局面から逆順に局面を構成し，構成した局面がどちら側が勝ちかを求めてそれをデータベース化する．残った駒が数個になったときは表引きで答えがでる．

また，トランスポジションテーブルというものが考えられた．これはほかのコンピュータゲームにも広く用いられる．これは，探索中に，一度訪問した局面を再度訪問したとき，最初に訪問したときに求めた評価値などを再利用するものである．このことにより，そこからの先読みをしなくて済むようにしたものである．これは一種のキャッシュメカニズムである．ハッシュテーブルを用いるが，ほかのハッシュ法と異なり，キーが衝突しても上書きしてしまう（データがなくなっても先読みするだけなので）など特有の方法である．書かれている評価値が真値か上限か下限か，また深さの差により上書きするか否かを決める．

並べ替え，ウィンドウ探索，キラー手，等々の技術が 1970 年頃までに生まれ，コンピュータチェスが進歩した．ハードウェアはムーアの法則として述べられるような進歩があったが，コンピュータチェスのアルゴリズムでも同様な一定の進歩が続いたと思われる．

スポーツなどで強さを測る尺度として Elo レーティングがある．レーティング 200 差というのが 3 勝 1 敗の強さの差に相当する．これを用いて調べてみると，年率 50 位の一定の速度でコンピュータチェスの強さが上昇してきたのが分かる．

CRAY-BLITZ，HITECH，DEEP THOUGHT らが覇を競っていた 1990 年ころまでのレーティングデータを未来に外挿すると，1994 年にコンピュータが人間のチャンピオンの強さになる，と予測された．一方，コンピュータチェスが人間のチャンピオンをいつ破るかのアンケートが 1989 年に行われた．その平均値は 2005 年だった．結局，1997 年に DEEP THOUGHT の後継である Deep Blue が人間のグランドチャンピオンの Gary Kasparov に勝った．レーティングによる機械的予測の方が，人間の心情的な予測より現実に近かった．

こうしてコンピュータチェスが人間に勝つという目標が終わった．ICCA（国際コンピュータチェス協会）が ICGA（国際コンピュータゲーム協会）になったことで分かるように，コンピュータチェスに対する興味も小さくなった．ただ，ここまでにたくさんの成果が生まれた．その意義は大変大きいと思う．また，まだ解かれていないチェスの課題もいくらでもある．チェスゲームを解くという課題は永遠に不可能といってもよいだろう．今日のコンピュータ思考ゲーム将棋や囲碁など，まだ人間に追いついてないゲームに興味が向かっているが，それにはコンピュータチェス研究の成果が多く活用されている．

■参考文献

1) D. Levy, M. Newborn : “How Computers Play Chess,” W. H. Freeman and Company, 1990.（小谷善行（監訳）: “コンピュータチェス,” サイエンス社, 1994.）

■S3 群－3 編－5 章

5-2 将棋とゲーム木探索

■S3 群－3 編－5 章

5-3 詰将棋と AND/OR 木探索

（執筆者：岸本章宏）［2008 年 12 月 受領］

5-3-1 詰将棋探索の研究意義と歴史

起源が江戸時代であると言われる詰将棋には，将棋図巧や将棋無双などの芸術的で難解な作品も存在し，現在も数多くの愛好家がいる．詰将棋は，攻め方が王手の連続によって，相手玉（玉方）を詰める手順を見つける問題である．例えば，**図 3･1** は，▲3 四馬△同歩▲2 三歩成の 3 手詰（▲は攻め方で，△は玉方）である．玉方には，△同歩以外に△1 三玉や駒を打つ手などがあるが，どの指し手に対しても，攻め方には玉方を詰める手がある．

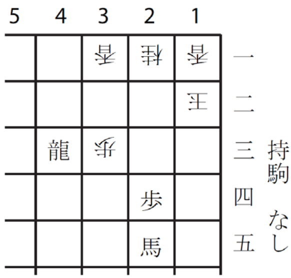

図 3・1　詰将棋の例（文献 4）より抜粋）

詰将棋では，攻め方の局面 P の詰みを示すには，P でのすべての王手のなかで，少なくとも 1 つが詰みであればよい．一方，受け方の局面 Q の詰みを示すには，Q でどの受け手（王手から逃れる指し手）を指しても，詰みであることを示す必要がある．このため，局面を探索木の節点，指し手を探索木の枝に対応させれば，詰将棋を解くプロセスは，AND/OR 木探索にモデル化できる．本稿では，詰将棋に利用する AND/OR 木探索を詰将棋探索と呼ぶ．詰将棋探索は，詰み手数が長くなれば組合せ爆発を起こす計算機科学としても難しい問題である．また，詰みを見つける際に調べる必要のない局面があり，このような局面を多く見つければ，探索性能が大幅に向上する面白い問題である．

詰将棋探索の研究は 1970 年頃から始まった．90 年代に盛んに行われ，脊尾は，現在の最長手数問題である 1525 手詰の「ミクロコスモス」を解くなど，顕著な成果を挙げた [1)]．2000 年代に，長井は df-pn [2)] を開発し，当時に存在した 300 手以上の長手数の問題をすべて解いた．

5–3–2 詰将棋探索手法の詳細

代表的な詰将棋探索である df-pn は，最良優先探索である証明数探索（Proof-number Search）[3)] の考え方に基づくが，効率化のために深さ優先探索に変換している．df-pn における節点の有望度を表す指標は，証明数（Proof Number）と反証数（Disproof Number）[3)] である．証明数と反証数は双対な概念であり，節点 n の証明数（反証数）は，n が詰む（詰まない）ことを示すのに展開する必要のある先端節点数の最小値である．証明数（反証数）が小さければ，攻め方（玉方）にとって有利であると考える．

df-pn は，攻め方は証明数が最小の指し手を選び，玉方は反証数が最小の指し手を選択する．大まかには，ルート局面からこの選択で到達した先端節点を展開し，証明数・反証数を再計算するという作業をルート局面が解けるまで繰り返す．df-pn は，証明数と反証数の 2 つの閾値を利用して，この過程を深さ優先探索で行う．証明数（反証数）の閾値は，証明数（反証数）がその閾値未満で，節点 n を解けるかどうかを調べるのに利用する．現在の証明数または反証数の閾値で n が解けなければ，より大きな閾値で n が解けるかどうかを調べる．このため，df-pn では，同一節点の再展開が生じる．この再展開の手間を減らすために，局面をハッシュキーにしたハッシュ表を利用して，以前の探索結果を保存している．

df-pn を詰将棋に利用する際には，様々な問題が生じる．代表的な問題と解決策を次に示す．

(a) Graph History Interaction（GHI）問題

詰将棋では，以前に出現した局面に戻る手順がある．ルールでは，攻め方の連続王手で同一局面が 4 度出現すれば，攻め方の負けである．つまり，詰将棋の探索空間は，サイクルを含むグラフであり，詰むかどうかが，手順に依存することがある．しかし，df-pn のハッシュ表では，局面 P に至る手順を無視して，P の探索結果を保存するので，詰将棋探索が詰む問題を詰まないと誤判定する GHI 問題[5] が生じる．GHI 問題解決策には，岸本の手法[6] がある．

(b) 無限ループ問題

df-pn には，サイクルのために新たな先端節点を永久に展開できないことがある．この問題に対する有効な解決策は，文献 7) がある．

(c) 合駒対策

玉方に様々な駒を打つ受け（合駒）がある場合に，攻め方はその合駒を単に取った後，駒を使わなくても玉が詰むときには，無駄合と呼ばれる．無駄合は，探索空間を大幅に大きくするうえに，詰将棋の解答手順には含めない．合駒の効率的な探索方法には文献 8) がある．

(d) メモリ対策

計算機のメモリは有限であるので，df-pn がハッシュ表の要素を使い切った後に，更に探索結果を保存したければ，ハッシュ表から以前の探索情報を消す必要がある．長井は，各節点の部分木のサイズに注目し，このサイズが小さな節点をハッシュ表から消去する方法を開発した[2]．この方法が，高性能な詰将棋探索を実現する原動力になった．

■参考文献

1) 松原　仁, 竹内郁雄(編)：“ゲームプログラミング,”共立出版, 1997.
2) 長井　歩：“df-pn アルゴリズムと詰将棋を解くプログラムへの応用,”アマ 4 段を超えるコンピュータ将棋の進歩 4, pp.96-114, 共立出版, 2003.
3) L.V. Allis, M. van der Meulen, and H.J. van den Herik：“Proof-number search,” Artificial Intelligence, vol.66, no.1, pp.91-124, 1994.
4) 松田茂役：“将棋必修　詰将棋　200　題,”梧桐書院.
5) A.J. Palay：“Searching with Probabilities,” PhD thesis, Carnegie Mellon University, 1983.
6) 岸本章宏：“不詰を正しく証明するアルゴリズム,” 第 9 回ゲーム・プログラミングワークショップ, pp.1-8, 2004.
7) A. Kishimoto and M. M ¨ uller：“Df-pn in Go: Application to the one-eye problem. in Advances in Computer Games,” Many Games, Many Challenges, pp.125-141, 2003.
8) 脊尾昌宏：“詰将棋を解くアルゴリズムにおける優越関係の効率的な利用について,”ゲーム・プログラミングワークショップ ’99, pp.129-136, 1999.

■S3 群－3 編－5 章

5-4 囲碁とモンテカルロ木探索

（執筆者：但馬康宏）［2018 年 12 月 受領］

二人完全情報ゲームにおいてゲーム木探索は有効な着手選択手法であるが，必勝必敗の探索のみならず途中局面に対する評価値の探索でも局面数の爆発的増加のために大きな計算量を必要とする．そのため α β 探索を含め多くの改善手法が提案されているが，2007 年頃からランダムシミュレーションによる探索手法が注目を集めるようになった．この手法は評価関数の作成が難しい囲碁において大きな成果をあげており，その他の二人完全情報ゲームにおいても比較的手軽にある程度の強さを実装することができる．

ランダムシミュレーションによる着手選択の基本は，着手可能な手についてその勝率を求めることにより行われる．勝率は，その手以降をランダムに選択するシミュレーションを複数回行うことで求められる．しかし，単純な勝率では必勝手を求めることはできないので，ゲーム木と組み合わせて探索が行われる．そこでまず，多腕バンディット問題をゲーム木探索に適用することを考える．

多腕バンディット問題

プレイヤーは k 個のスロットマシンから 1 つを選択しプレイする．各スロットマシンは報酬の期待値があらかじめ設定されており，その期待値を持つ確率変数として報酬をプレイヤーに返す．プレイヤーは n 回のプレイで報酬をできる限り大きくすることがゲームの目的である（**図 4･1**）．

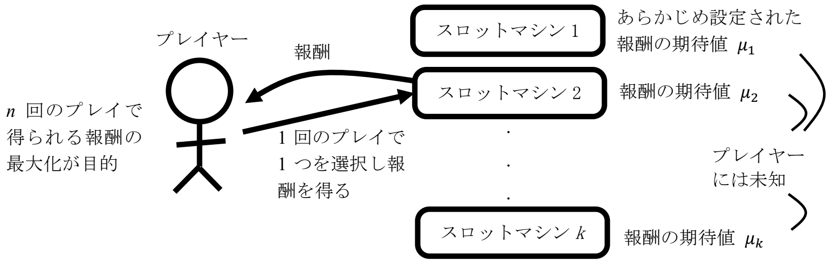

図 4・1　多腕バンディット問題

この問題に対しては様々な問題設定の派生とそれに対するアルゴリズムが考案されているが，古典的なアルゴリズムとして UCB1 アルゴリズム [1] がある．これは過去 m 回のプレイの後，次のスロットマシンの選択を $\bar{X}_j + \sqrt{\frac{2\ln m}{m_j}}$ が最大であるスロットマシン j とするものである．ここで，m_j は過去 m 回のプレイでスロットマシン j を選択した回数であり，$\bar{X}_j$ はその m_j 回のプレイで得られた報酬の平均値である．このアルゴリズムでは，報酬期待値の最も大きな値 $\mu^* = \max \mu_j$ と n 回のプレイで得られた報酬の総和 S_n について

$$n\mu^* - S_n \leq \left[\sum_{j:\mu_j<\mu^*} \frac{\ln n}{\mu^* - \mu_j}\right] + \left(1 + \frac{\pi^2}{3}\right)\sum_{j=1}^{k}(\mu^* - \mu_j)$$

となることが知られている．すなわち，真の期待値 μ_j が小さなスロットマシンの選択回数が，その小ささに応じて制限されるアルゴリズムである．

モンテカルロ法によるゲーム木探索においては，途中局面での手の選択を多腕バンディット問題とみなすことにより効率的なシミュレーションを行っている [2]．しかし，実際のゲーム木探索では展開した途中局面の深さに依存して勝率も変わるため，以下のようなサイクルを繰り返して探索精度を上げていく．この探索が一般的にモンテカルロ木探索 [3] と呼ばれている（**図 4･2**）．

1. Selection：展開済みのゲーム木を多腕バンディット問題の適用などで葉までたどる．
2. Expansion：たどり着いた葉からゲーム木を新たに展開する．
3. Simulation：新たに展開した節点から，ランダムな着手でゲームの勝敗を求める．
4. Backpropagation：得られた勝敗をゲーム木の途中節点の勝率などに反映させる．

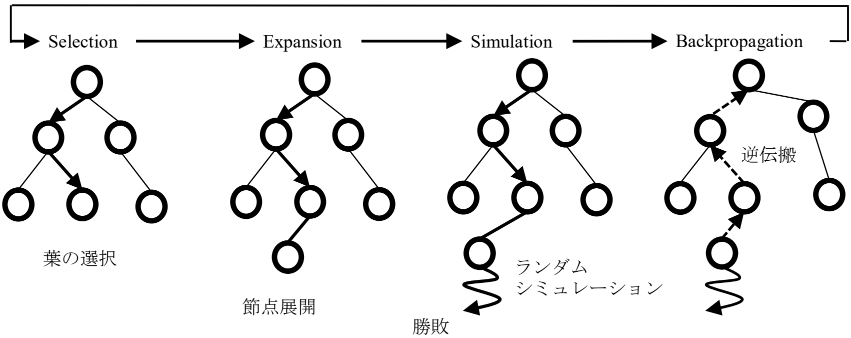

図 4・2　モンテカルロ木探索

他の二人完全情報ゲームと同じくコンピュータ囲碁でも評価関数を用いたゲーム木探索が研究されてきたが，囲碁は可能手が多い上に盤面の評価が単一的な視点では難しく，大局観と局所的な状況判断を同時に行う評価関数を作成することが難しいとされてきた．しかし，モンテカルロ木探索の登場により急速に棋力が上がり，プロ棋士とも互角に対戦できるようになった．更にモンテカルロ木探索とディープラーニングとの組合せによりトッププレベルのプロ棋士を凌駕する強さとなった [4]．

コンピュータ将棋においては，モンテカルロシミュレーションは主流とはなっていない．これは囲碁に比べ着手の順番が大きな意味を持つことなどから，モンテカルロ木探索の効果が大きくないとみなされている．

■**参考文献**

1) P. Auer, N. Cesa-Bianchi, P. Fischer：“Finite-time analysis of the multiarmed bandit problem,” Machine Learning, vol.47, nos.2,3, pp.235-256, 2002.

2) L. Kocsis, C. Szepesvari：“Bandit based Monte-Carlo planning,” LNCS, vol.4212, pp.282-293, 2006.

3) G. M. J-B. Chaslot, M. H.M. Winands, H. J. van den Herik, J. W.H. Uiterwijk, B. Bouzy：“Prograssive strategies for Monte-Carlo tree search,” New Mathematics and Natural Computation, vol.4, no.3, pp.343-357, 2008.

4) D. Silver et al.：“Mastering the game of Go with deep neural networks and tree search,” Nature, vol.529, pp.484-489, 2016.

5-5 バックギャモンと TD 法