2.2 詰将棋

宇宙開発事業団
長井　歩
nagai.ayumu@nasda.go.jp

■詰将棋とは？

詰将棋とは，攻め方と玉方がいて，攻め方が王手のみによって玉を詰ますことができるかを楽しむパズルである．**図 -1** は詰将棋の例である（内藤国雄著「九級から一級までの詰将棋」から引用）．正解手順は▲ 3 三角成 ▽同桂▲ 3 一銀不成▽ 1 一玉▲ 2 二金までの 5 手詰である（ここで，「▲」は攻め方の手を，「▽」は玉方の手を指す）．攻め方の手はいずれも王手になっている．玉方の手は攻め方の王手から逃れる手である．

詰将棋は江戸時代に起源があるといわれている．最初は指し将棋の終盤的要素が強かったが，現在では完全に作品として捉えた方が適切である．詰将棋には目から鱗が落ちるような技巧的なものも多い．まるでからくり人形を見ているような，職人芸的なからくりを持っているのである．ときどき詰将棋が箱根の寄木細工のように感じることがある．詰将棋を芸術として捉える人々は，この絶妙なからくりを堪能しているのである（詰将棋のからくりを堪能されたことのない方は，「将棋図巧」の詰将棋を並べてみて，本に載っている正解手順の通りに駒を動かしてみることを勧める）．このように詰将棋がパズル化，芸術化した要因として，ルールの整備が大きな役割を果たしていると考えられる．すなわち，詰将棋の最大のルールとは，正解手順が唯一であることである．

将棋よりも世界的な広がりをみせているチェスにも「チェスプロブレム」という詰将棋に相当するパズルがあるが，1 つ 1 つの問題があまりにもバラバラで，パズルとして成立させるために苦労してルールを作っている印象がある．

詰将棋を計算機によって解く試みは 1970 年頃から行われている．初期のプログラムはアルファベータ法を用いていた．アルファベータ法をベースにさまざまな工夫を採り入れることによって，徐々に解答能力が向上し，1994 年までに野下のプログラム T2[8] は 25 手詰以下の問題はほぼすべて解けるまでに至った．しかしアルファベータ法をベースとする探索法はこの辺りで限界を迎えた．その後は，詰将棋の探索木を AND/OR 木と見なした探索法が着目されるようになった．最良優先法の採用によって，25 手よりも手数の長い詰将棋も徐々に解けるようになっていった．そして，1994 年には 611 手の超長手数詰将棋「寿」が河野のプログラム，伊藤のプログラム Ito，野下のプログラム T3 によって解けるまでに至った[4]．これらはいずれも最良優先法であった．さらに 1995 年，脊尾のプログラム「脊尾詰」[6] が登場し，「寿」はもちろん，873 手の「新扇詰」など，それまで計算機で解かれていなかった数多くの詰将棋を解いた．そして 1997 年，ついに現在の最長手数の詰将棋 1,525 手の「ミクロコスモス」を解くという偉業を成し遂げた．脊尾のプログラム「脊尾詰」は完璧にみえたが，苦手な詰将棋がいくつか存在した．2001 年，長井のプログラムは「脊尾詰」にも解けなかった詰将棋 12 題を含む，ほとんどの詰将棋を解くことに成功し，300 手以上の詰将棋のすべてを解くことに初めて成功した[7]．

■計算機による探索

思考ゲームの探索は木（正確にはグラフ）の探索というかたちでモデル化できる．思考ゲームの「局面」は探索木の「節点」に，思考ゲームの「指し手」は探索木の

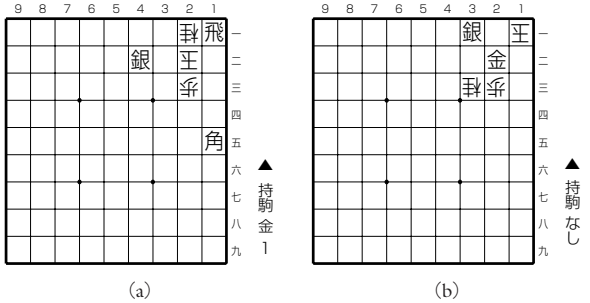

図 -1　詰将棋の例．（a）が問題，（b）は（a）の詰め上がり

Game Informatics

	アルファベータ法	AND/OR 木探索法（df-pn）
探索の打ち切り	深さによる打ち切り	しきい値による打ち切り
葉節点にて	評価関数呼び出し	特に何もしない
内部節点にて	min-max 原理による伝播	証明数・反証数としきい値計算

表 -1　アルファベータ法と AND/OR 木探索法

「枝」に対応する．したがって，「局面」を「節点」に，「指し手」を「枝」に置き換えてしまえば一般の探索についての議論になる．

与えられた局面（ルート局面）から到達可能な局面すべてを探索できれば，常に最善手を指すことが可能になる．しかしそれは多くの場合は，現実的な時間内では不可能である．将棋では，各局面にてルール上指せる手は平均 80 前後あるといわれており，すぐに組合せ爆発を起こしてしまうからである．大詰めの終盤の局面ならともかく，プロの将棋指しでも完全に読み切るのは不可能である（だからこそ今に至るまで将棋愛好家は尽きないのである）．そこで，ルート局面から到達可能な膨大な局面のうち，ごく一部のみを探索して指し手を決定しないとならない．人間の場合，各局面にて平均 80 ある可能な指し手のうち，ほんの数手の有望な手のみを深く読む．それ以外のほとんどの手はそれほど真剣には読まないのである．数手の有望な手の抽出の仕方は，その人の経験・勘・研究に大いに依存する．

計算機で思考ゲームをプログラミングするに際には，大きく次の 2 つの手法が採用されている．1 つはアルファベータ法（の何らかの派生形）で，もう 1 つは AND/OR 木探索アルゴリズムである．アルファベータ法は探索木を比較的浅く網羅的に探索する手法で，前章「将棋」でもミニマックス法とアルファベータ枝刈りを利用した方法として紹介した．アルファベータ法は一定の深さまでを読み漏らしなく探索できる．思考ゲーム木探索の基本であり，序盤から終盤までを通してこの手法を用いるのが一般的である．一方，AND/OR 木探索アルゴリズムは終盤に威力を発揮し，勝ち負けを正確に判定することができる．思考ゲーム木探索の終盤や，詰将棋のようなパズルに適用することができる．AND/OR 木探索法の代表として df-pn とアルファベータ法の特徴を**表 -1** にまとめた．アルファベータ法では葉節点において評価関数を呼び出すのが特徴的なのに対し，AND/OR 木探索法では内部節点における証明数・反証数の計算が重要である．アルファベータ法と AND/OR 木探索アルゴリズムのどちらを実装する場合にも，結局人間の思考法が非常に良い手本になる．人間ならどう思考するかと照らし合わせることによって探索を改善することができる．

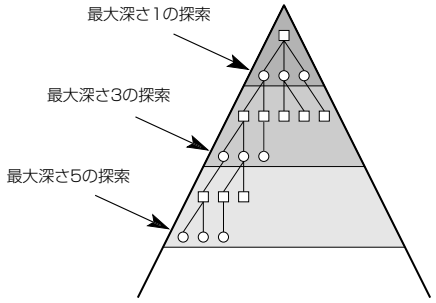

図 -2　アルファベータ法＋反復深化のイメージ

■アルファベータ法によるアプローチ

初期の詰将棋を解くプログラムはアルファベータ法をベースに，詰将棋用にアレンジしていた．最も大きなアレンジは，指し手の生成である．攻め方は王手のみに限定し，玉方は王手から逃げる手に限定している．アルファベータ法は通常，反復深化とともに用いる．すなわち，最大探索深さを 1, 2, 3…のように 1 ずつ増やしながら，アルファベータ法を繰り返し何度も用いるのである．詰将棋用にアレンジするにおいては，最大探索深さを 1, 3, 5…のように 2 ずつ増やしながら，アルファベータ法を繰り返し何度も用いる．それは，1 手詰はないか，3 手詰はないか，5 手詰はないか…という探索に相当する（**図 -2** 参照）．アルファベータ法をベースとする詰将棋プログラムの代表は野下の T2[8] である．T2 は指し手の順序付け，合駒対策，ハッシュ表の導入など，考え得るさまざまな工夫を導入している．その結果 1994 年までには，17 手詰までの問題は人間以上に高速に解くことができ，25 手詰まではほぼすべて解くことができた．しかし，計算量爆発のため，27 手詰以上の詰将棋はほとんど解けなかった．詰将棋を解くことに対しては，この辺りがアルファベータ法の限界であった．

■ AND/OR 木探索法によるアプローチ

アルファベータ法の限界が見え始めた 1990 年代始め頃から，アルファベータ法に代わるアプローチの模索が始まり，AND/OR 木探索法が着目されるようになった．AND/OR 木探索法は，双方が最善を尽くしたとして，最終的な勝ち負けを正確に調べる探索である．もし勝てるなら，勝ちに至る指し手以外の手を調べるのはあまり意味がない．そこで，もし勝てるとすると，どの手が最も有望かを「ある仮定」のもとに計算し，常にそのとき最有力な手を探索し続けるものである．必然的に最良優先的な振舞いをする探索になる．「ある仮定」というのは，どの未展開節点（その時点の探索木では葉節点になっているが，まだ未探索の子節点が存在する節点）も同じ労力で勝ちや負けに至ることが判明する可能性がある，ということである．その上で，自分の勝ちが判明するために必要な労力の最小な指し手のことを，最有力な指し手と考えるのである．労力のことを正確には「証明数」や「反証数」と呼ぶ．証明数は勝ちを示すために最低限必要な労力を表し，反証数は負けを示すために最低限必要な労力を表す．両者は双対な関係にある（アルファベータ法のアルファ値とベータ値が双対な関係にあるのと同様である）．攻め方は勝ちを狙っているので，証明数に興味がある．証明数最小の子節点に至る指し手が最有力ということになる．逆に玉方は反証数最小の子節点に至る指し手が最有力ということになる．したがって AND/OR 木探索は最良優先的な探索法になる．証明数最小・反証数最小という意味で最良な節点を展開していくことになる．

最良優先の AND/OR 木探索法

AND/OR 木探索法は自然に考えると最良優先探索になるので，アルファベータ法からの打開を図った結果たどり着いたアプローチも最良優先の AND/OR 木探索法であった．しかも証明数のみを用い，反証数は用いていなかった．当時は深さ優先の AND/OR 木探索法は知られていなかった．また，反証数という概念も知られていなかった．伊藤のプログラム Ito，河野のプログラムがその典型的な例である [4]．ルート節点から証明数最小の子節点をたどっていき，未展開節点（これが最有力な節点である）にたどり着いたら展開する．証明数の計算をし直して，再びルート節点から証明数最小の子節点をたどる，というのを繰り返す（**図 -3** 参照）．これは AND/OR 木探索法として古くから知られている AO* [3] の一種である．しかしながら，AO* には非常に

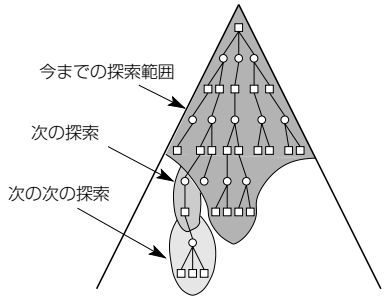

図 -3　AO*，pn-search のイメージ

	内部節点にて利用する情報	
	証明数	証明数と反証数
深さ優先 最良優先	脊尾のアルゴリズム AO*	df-pn pn-search

表 -2　AND/OR 木探索法の比較

多くのパラメータが出現するので，有効な適用の仕方が分かりにくい．伊藤，河野らの成果は AO* の簡単で実用的な適用の仕方を示したものといえる．最良優先の AND/OR 木探索法の採用により，それまでアルファベータ法によっては手数が長過ぎて歯が立たなかった長手数の詰将棋が解けるようになってきた．611 手詰の「寿」を解いたのが最大の成果である．

深さ優先の AND/OR 木探索法

1995 年，脊尾のプログラムがそれまで計算機で解かれていなかった数多くの詰将棋を解くことに成功した [6]．注目すべきは，脊尾のアルゴリズムが深さ優先探索法であった点である．それまで AND/OR 木探索は最良優先になると信じられていたが，深さ優先でそれと等価な探索を実現できることを示したのである．すなわち，伊藤，河野らが用いていた，簡単で実用的な AO* と等価な振舞いをする．脊尾のアルゴリズムも証明数のみを用いている．ここでいう，「等価」の意味であるが，ルート節点から子節点をたどるとき，証明数最小の子節点をたどるが，この時兄弟節点の間で同じ証明数の節点が出現しない，もしくは同じ証明数の節点同士の間に何らかの優先順位付けができるという仮定のもとで，未展開節点の展開の順序が一致するということである．その意味で，最良優先である AO* と深さ優先である脊尾のアルゴリズムは等価な振舞いをする（**表 -2** 参照）．アルファベータ法＋反復深化は，1 手詰はないか，3 手詰はないか，5 手詰はないか…という探索であった．それに対

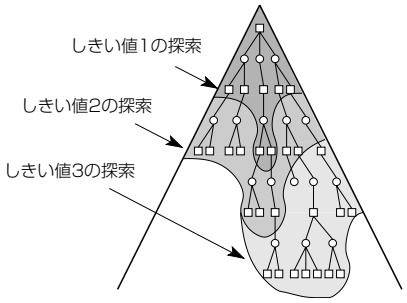

図 -4　脊尾のアルゴリズム，df-pn のイメージ

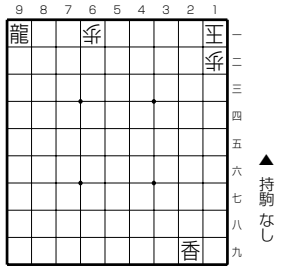

図 -5　無駄合いの例（▲９一飛成に対する▽６一歩）

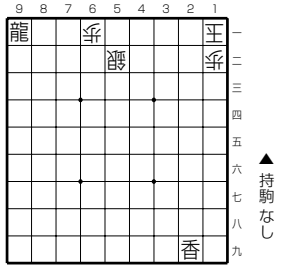

図 -6　有効な合駒の例（▲９一飛成に対する▽６一歩）

し，脊尾のアルゴリズムでは，証明数に対してしきい値を導入し，証明数がしきい値に達したら探索を打ち切るようにしている．証明数のしきい値は 1, 2, 3…と反復ごとに増やしていき，その都度探索空間は広くなっていく．これは，1 本道の詰み（一直線の詰み）はないか，2 本道の詰み（途中に 1 つだけ分岐のある詰み）はないか，3 本道の詰み（途中に 2 つだけ分岐のある詰み）はないか…という探索に相当する（**図 -4** 参照）．このようにして，最良優先的な振舞いをする深さ優先探索法を実現しているのである．

脊尾のアルゴリズムは証明数のみを用いた，深さ優先探索法であった．深さ優先で最良優先的振舞いを実現することの最大の利点は，メモリ消費に関して優位に立てることである．最良優先探索ではメモリを使い切ると，それ以上の探索続行が困難になるという大問題がある．しかし，深さ優先ならハッシュ表を上書きしてしまえばよいのでそのような問題からは解放される．

証明数・反証数の両方を用いた深さ優先探索法

1994 年 Allis により，反証数の概念が確立され，証明数と反証数の両方を用いた最良優先探索法 pn-search が提案された [2]．それまでは反証数の概念は一般には知られてはいなかった．1999 年，長井は pn-search と等価な振舞いをする深さ優先探索法 df-pn（表 -2 参照）を提案し，2001 年，脊尾のプログラムをもってしても解けていなかった長手数の詰将棋 12 題を解き，300 手以上の超長手数詰将棋のすべてを初めて解くことに成功した [7]．df-pn ではしきい値を 2 つ用いている．証明数に対するしきい値と反証数に対するしきい値の 2 つである．前者のしきい値は，1 本道の詰みはないか，2 本道の詰みはないか…という探索を実現するためのしきい値で，後者のしきい値は，1 本道の不詰はないか，2 本道の不詰はないか…という探索を実現するためのしきい値であり，両方の探索を同時に行っている．

また，長井のプログラムではメモリ消費対策にかなり気を配っている．現在の汎用 PC はメモリを大量に搭載できるようになり，長手数詰将棋を力技で解けるようになってきた．しかしながら現在の計算機環境をもってしても，メモリ不足に陥りやすいような難易度の高い詰将棋も存在する．長井のプログラムが 451 手詰「乱」[1] を解けるのはメモリ消費対策が功を奏しているのだと考えている．また，最長の詰将棋である 1,525 手詰「ミクロコスモス」[5] を解けるプログラムは現在複数存在するが，いずれの報告も膨大なメモリを使用した計算機環境（300MB ～ 800MB 程度）によるものである．それに対し，長井のプログラムは 5.6MB という極端に少ないメモリでも解ける．

■詰将棋探索の難しさ～合駒～

詰将棋で厄介なのが合駒の処理である．合駒とは，王手している駒の効きをブロックすることによって受ける手のことである．しかし合駒の可能な局面において，多くの場合は単に取られるだけで合駒する意味のない手である（「無駄合い」と呼ぶ）．このような手は，いたずらに手数を引き延ばし，見苦しいだけなので，正解手順の中に「無駄合い」の手は含めない．**図 -5** は無駄合いの例である．▲９一飛成に対し，▽６一歩と合駒してい

るが，これは典型的な無駄合いである．▲6一同龍のようにただで取られるだけで，まったく受けになっていない．一方，**図 -6** は有効な合駒の例である．同じように▲9一飛成に対し，▽6一歩と合駒しているが，▲6一同龍に対しては▽6一同銀の手があるので，▽6一歩は有効な合駒である．図 -5 では，▽6一歩が無駄合いとなっていたが，玉方は2一から8一のどこにどの駒を合駒しても，それは無駄合いになってしまう．正解手順に無駄合いの手は含めないので，図 -5 の1手前の局面（▲9一飛成まで）は，合駒という合法な手が多数あることはあるが，詰め上がりということになる．このように，正解手順の表示の際には，合駒の有効性の判定が必要になってくる．合法な手として実際に出現する合駒の多くは無駄合いである．正解手順に無駄合いの手を含まないようにする対策としては，「柿木の無駄合い判定アルゴリズム」が有名である．また，「無駄」をより厳密に定義した上での判定法としては，「河野の判定法」がある．

また，合駒の処理について，人間は無意識に効率の良い探索をしている．図 -5 のような無駄合いについて，すべての組合せについて逐一調べるようなことはしない．「▽6一香も▽6一歩と同様」のように，端折った探索を行っている．計算機にも，これに相当するような処理を行えるようにしないと，人間の思考に追いつけないという問題も存在する．これに対しては，探索に使用するハッシュにある工夫を施すことによって，探索中に自然に解消できる方法が脊尾によって提案されている．

このように合駒の処理は，単に正解手順を表示するときの問題だけでなく，人間の思考法を参考に，探索方法についても効率化の余地があることを示唆している．

■詰将棋探索の難しさ～千日手～

将棋を指していると，まれに同じ手順を繰り返してしまうことがある．これを「千日手」と呼ぶ．千日手の厳密なルールは，同一局面が4回出現したときである．通常，千日手は引き分けになる（プロの対局では，千日手になると，先手と後手の手番を逆にしてもう一度指し直しになるので，結局は勝負がつく）．千日手は千日手でも，王手を連続して千日手に至った場合，王手している側が負けになる（「連続王手の千日手」）．詰将棋では，攻め方は王手の連続によって玉を詰ます．したがって，詰将棋における千日手は攻め方の負けとなってしまうのである．詰将棋では，攻め方は千日手を避けながら玉を詰まないとならない．

これを実際の探索において実装しようとすると，非常に悩ましい状況になってしまう．長手数の詰将棋を解こうとする場合，千日手対策は絶対不可欠である．長手数の詰将棋はほとんど同じ手順を何度も繰り返すことによって手数を伸ばしているからである．ほとんど同じ手順でありながら，同一局面の出現はかろうじて回避している．したがって長手数の詰将棋を解こうとすると，探索木の中に潜在的な千日手は必然的に多数出現する．

これを回避する基本的なアイディアは脊尾が考案した．長井のプログラムにおいては，脊尾のアイディアを発展させて用いることによって完全に回避している．

■今後の展望

現在の詰将棋を解くプログラムは実用上，問題のない域に達している．あえて挙げるとすれば，裸玉問題に対する解答能力が弱いことである．裸玉問題とは盤面上に玉のみが乗っているような詰将棋である．裸玉問題にも強い詰将棋プログラムは作れるのであろうか．また，何らかの意図を持った詰将棋を自動生成するような研究も発展して欲しいものである．さらに，詰将棋よりも一段難しい必至問題を解く研究も楽しみである．

参考文献

1）詰将棋パラダイス，1999 年 10 月号．
2）Allis, L. V., Vander Meulen, M. and Vanden Herik, H. J.: Proof-Number Search, Artificial Intelligence, Vol.66, pp.91-124（1994）．
3）Nilson, N. J.: Principles of Artificial Intelligence, Tioga Publishing Company, Palo Alto, CA（1980）．
4）伊藤啄巳，河野泰人，野下浩平：非常に手数の長い詰将棋問題を解くアルゴリズムについて，情報処理学会論文誌，Vol.36, No.12, pp.2793-2799（Dec. 1995）．
5）角　建逸：詰将棋探検隊，毎日コミュニケーションズ（1995）．
6）脊尾昌宏：共謀数を用いた詰将棋の解法，第2巻，1コンピューター将棋の進歩，pp.1-21, 共立出版（1998）．
7）長井　歩，今井　浩：df-pn アルゴリズムの詰将棋を解くプログラムへの応用，情報処理学会論文誌，Vol.43, No.6, pp.1769-1777（June 2002）．
8）野下浩平：詰将棋を解くプログラム T2，第1巻，3コンピューター将棋の進歩，pp.50-70, 共立出版（1996）．

（平成 15 年 6 月 9 日受付）

Game Informatics