将棋ソフト作成入門

はじめに

チェス、囲碁、将棋などのソフトウエアは年々進歩し、将棋においては名人でもコンピュータに勝てない状況になっています。それとても将棋ソフトは高度な人工知能の手法を駆使した特殊なプログラムという見方がなされるようになり、一般の情報技術者には縁遠い存在になっています。

しかし、課題を限定すれば初心者にも作成可能な将棋関連のプログラムはあります。これらは将棋を趣味としている人にとって価値があります。そしてその製作は汎用的なアルゴリズム手法を活用し、広く情報技術者にとって有意義なものです。

将棋ソフトは人工知能を活用していると言われます。人工知能は先進的なコンピュータであると考えている人が多いようです。将来性があるとして関心が高まっています。しかし人工知能は従来型のコンピュータとは大きく異なる特徴を持っています。気をつけないと、従来のコンピュータが備えていた長所が損なわれる危険性もあります。次の言葉を考えてみましょう：

> 「予算というものには正解は無い。しかし、明らかな間違いはある」

ここの「予算」というところは色々な言葉に置き換えられます。このような性質の事象は世の中に多く存在します。コンピュータは上の「正解が見えない中でなるべく良い答を出す」「明らかな間違いを防ぐ」双方の仕事を担ってきましたが、これまでは後者の役割で多く使われて来ました。

「完璧でなくとも最良の答を出す」コンピュータの利用法はなかなか難しいものですが、最近は自動学習（機械学習）が進歩し、これまで出番が無かった分野にもコンピュータが進出すると期待されています。しかし気をつけないと、コンピュータが「明らかな間違い」を犯すようになります。これをどう防ぐか、今後大きな課題になって行くでしょう。

将棋ソフトはソフト同士を戦わせながらパラメータの調整していく自動学習の手法の効果が大きかったと言われています。しかし将棋ソフトの強さはそれだけではありません。プロの棋士でも稀に二歩などの反則で負けますが、将棋ソフトはそのようなミスはしません。「明らかな間違いを避ける」のに秀でているのです。ミスはしないように構成されているのです。こうした構成が可能なのは将棋等盤上遊戯が比較的単純な課題であることによる所が大きいと思います。皆様も単純な課題に取り組むことを通して、構成上の留意点を確認することができます。

本稿では将棋ソフトでも初歩的な盤面の入力、盤面の画像出力、そして詰将棋問題の自動解決を行う物を取り上げます。特定の言語を意識せず、コードの例は画像出力の節だけにとどめます。

コンピュータにおける将棋盤面

チェスや将棋、囲碁などの盤上遊戯（ボードゲーム）をコンピュータで扱うには盤とそこに配置された駒をどう記録するか決めなければなりません。

すぐに思いつくのは２次元配列を用意し、駒の種類を数値等で表現して盤面の現状を表現する方法です。駒が置かれない空白のますは０とすれば良いでしょう。

上が一般的な方法ですが、他の方法もあります。将棋の駒は敵味方で合計４０枚でこの数は対局中増減しません。駒を表現する記憶領域を４０用意し、それぞれの位置、敵味方の別、成駒かどうかの情報を記録しておく方法もあります。盤中心と駒中心どちらが扱いやすいかをよく検討します。盤中心が駒中心に勝っていそうですが、複数のアプローチを考えるのはプログラミングの上では良い心がけです。異なる方途は検証の際に役立つものです。

盤面の入力と棋譜の再生

新聞・戦術書の棋譜

以下は新聞等に掲載される棋譜の例です：

☗７六歩　☖３四歩
☗６八飛　☖８四歩
☗４八玉　☖６二銀
☗３八玉　☖４二玉

将棋盤の座標系は横をアラビア数字、縦を漢数字で書く習わしです。横を先に書きます。
ｘ軸は左が大、ｙ軸は下が大、原点は右上と学校の数学とは反対になっています。
縦書きの日本語の順番と覚えておくと良いでしょう。

９一	８一	７一	６一	５一	４一	３一	２一	１一
９二								１二
								１三
	８四					３四		１四
								１五
		７六						１六
								１七
９八			６八					１八
９九				５九				１九

伝統的な棋譜の座標系

☗が先手、☖が後手です。先後の色は囲碁と同じ、チェスと反対です。☗☖の代わりに▲△と三角がよく使われています。

新聞や戦術書の棋譜表記は将棋を知っている人が見るためのものです。☗７六歩という表記が上にありますが、駒の動きを知っている人なら盤上に多数ある歩兵のどれが動いたのかすぐわかります。

将棋の棋譜を電子媒体に保存するフォーマットはいくつかあります。csa, kif, ki2 が人気があります。コンピュータに分かりやすい棋譜のフォーマットは csa です。人が見やすいのは ki2 で kif はその間となります。

csa 棋譜形式

以下は csa 形式の棋譜の例です。

```
V2.2
N+今川義元
N-織田信長
$EVENT:合戦
$SITE:桶狭間
$START_TIME: 1560/6/12 13:00:00
$OPENING:矢倉
P1-KY-KE-GI-KI-OU-KI-GI-KE-KY
P2 * -HI *  *  *  *  * -KA *
P3-FU-FU-FU-FU-FU-FU-FU-FU-FU
P4 *  *  *  *  *  *  *  *  *
P5 *  *  *  *  *  *  *  *  *
P6 *  *  *  *  *  *  *  *  *
P7+FU+FU+FU+FU+FU+FU+FU+FU+FU
P8 * +KA *  *  *  *  * +HI *
P9+KY+KE+GI+KI+OU+KI+GI+KE+KY
+
+7776FU
-8384FU
+7968GI
-3334FU

以下省略
```

csa 形式は全てアスキー英数字記号で記されています。コンピュータで読み取るのは簡単です。

最初に csa のバージョンがつきます。最近の棋譜ファイルなら v2.2 でしょう。次に対戦者、対局日時などの情報があります。この部分では漢字も利用します。その次に初期配置があります。初期配置図では+が先手、-が後手、そのどちらでもなければその位置は空白となりますが、（半角空白）*（半角空白）とするのが一般的です。

csa 形式 駒の表記

王	金	銀	桂	香	飛	角	歩
OU	KI	GI	KE	KY	HI	KA	FU
		成銀	成桂	成香	竜	馬	と
		NG	NK	NY	RY	UM	TO

指し手は次のように、+-による先手後手の別、移動の開始座標、終点座標、駒と順に記します：

+7776FU　　先手の歩兵が７七から７六に移動
-2262HI　　後手の飛車が２二から６二に移動

持駒を使う場合は開始座標を 00 とします:

+0055KA　　先手が角行を５五に打った

駒が成る場合、成駒の名前を記します:

-8687TO　　後手の歩兵が８七に進んでと金に成った

駒の動きは始点終点が明記されているので棋譜の進行は盤面現況を記す配列と持駒の状態を更新して行けばよく、初歩的なプログラミングの課題と言えます。

kif 棋譜形式

次に kif 形式を紹介します。この形式では半角英数字、全角漢字全角数字が使われます。

```
開始日時： 1555/10/16 6:00:00
棋戦：合戦
場所：厳島
持ち時間：各５時間
手合割：平手
先手：毛利元就
後手：陶晴賢
戦型：筋違角
手数----指手---------消費時間--
1 ７六歩(77) (00:00/00:00:00)
2 ３四歩(33) (00:00/00:00:00)
3 ２二角成(88) (00:00/00:00:00)
4 同　銀(31) (00:00/00:00:00)
5 ４五角打 (00:00/00:00:00)
6 ５二金(61) (00:00/00:00:00)
7 ３四角(45) (00:00/00:00:00)
8 ６二銀(71) (00:00/00:00:00)
9 ６六歩(67) (00:00/00:00:00)

以下省略
```

平手の対局では初期配置は省略されることが多いようです。詰将棋のように初期配置が必須の場合は後述の bod 形式を使います。

竜は「龍」とも表記します。成銀は「全」成桂は「圭」成香は「杏」と表記することが多いようです。１字にすると他と揃うのがメリットです。

動いた駒をすぐとる場合は「同」とします。
持駒を打つ場合は必ず「打」をつけます。
成る場合は「成」をつけますが、不成は表記しません。

ファイル名拡張子が .kif なら漢字・倍角文字は shift-JIS、.kifu なら UTF8 という情報もありますが、最近は拡張子が.kif でも UTF8 が多いようです。DOS-Windows 環境で作成されたファイルは改行が CR-LF となります。また UTF8 のファイルの最初に BOM がつくことがあり、注意する必要があります。

倍角数字や漢字を読み取る作業が加わりますが、指し手ごとに移動の始点が明記されているので、kif ファイルを読み込んで棋譜を進行させるのはさほど困難ではありません。これは初級のプログラミングの課題と言えましょう。

初期配置図と棋譜の部分は出現する漢字・倍角数字記号の種類と出現位置が限られているのでオリジナルの c 言語のようにキャラクターが８ビット、複数バイト文字非対応の処理系で読み込むプログラムもそれほど困難なく作成可能です。こんな課題、実際の仕事ではまず無いでしょうが一度やって見ると UTF8 等複数バイト文字の仕組みの理解が深まります。

bod 形式

詰将棋の棋譜など初期配置図が必要な場合使われる書式です。半角と全角を組にするのが特徴です。

後手の駒は前に半角 v をつけます。先手はここが半角空白です。先手後手どちらの駒も無いところは半角空白の後全角・とします。

```
後手の持駒：なし
  9  8  7  6  5  4  3  2  1
+---------------------------+
|v香v桂v銀v金v王v金v銀v桂v香|一
| ・v飛 ・ ・ ・ ・ ・v角 ・|二
|v歩v歩v歩v歩v歩v歩v歩v歩v歩|三
| ・ ・ ・ ・ ・ ・ ・ ・ ・|四
| ・ ・ ・ ・ ・ ・ ・ ・ ・|五
| ・ ・ ・ ・ ・ ・ ・ ・ ・|六
| 歩 歩 歩 歩 歩 歩 歩 歩 歩|七
| ・ 角 ・ ・ ・ ・ ・ 飛 ・|八
| 香 桂 銀 金 王 金 銀 桂 香|九
+---------------------------+
先手の持駒：なし
```

bod 形式による初期配置

```
後手の持駒：飛　角二　金三　銀三　桂四　歩十六
  9  8  7  6  5  4  3  2  1
+---------------------------+
| ・ ・ ・ ・v銀 ・ ・ ・v歩|一
| ・ ・ ・v歩 ・ ・v王 ・ ・|二
| ・ ・ ・ ・ 竜 ・ ・ ・ ・|三
| ・ ・ ・ ・ ・ ・ ・ ・ ・|四
| ・ ・ ・ ・ ・ ・ ・ ・ ・|五
| ・ ・ ・ ・ ・ ・ ・ ・ ・|六
| ・ ・ ・ ・ ・ ・ ・ ・ ・|七
| ・ ・ ・ ・ ・ ・ ・ ・ ・|八
| ・ ・ ・ ・ ・ ・ ・ ・ ・|九
+---------------------------+
先手の持駒：金
```

bod 形式による詰将棋問題図
（佐藤紳哉七段作 九手詰め）

ki2 棋譜形式

ki2 形式は新聞に掲載される棋譜に近く、これを読み慣れた将棋愛好家に好まれる形式です。移動の始点は記載無く駒と終点のみが記載されます。この形式を解釈するソフトウエアは駒の動きを理解している必要があります。

```
開始日時：
棋戦：合戦
場所：川中島
手合割：平手
先手：武田信玄
後手：上杉謙信
戦型：向飛車

▲７六歩 △３四歩 ▲２六歩 △３二金 ▲２五歩 △３三角 ▲同角成 △同　金 ▲８八銀 △２二飛
▲７七銀 △４二銀　　以下省略
```

csa 形式や kif 形式を扱うプログラムでも、駒の動きを理解したものも可能です。記録者の入力ミスを発見してくれる高級なソフトウエアとなります。

将棋ソフト作成に挑戦したいという小中高生がいたら、ki2 形式の棋譜、あるいは手で打ち込んだ新聞記載棋譜を読み込み、下記の画像出力を行うプログラムを目標とするよう助言すると良いでしょう。

盤面の画像出力

上記の csa 形式の初期配置図、bod 形式はテキストで盤面を表示する仕組みです。慣れない人にはわかりにくいでしょう。新聞に掲載されているような局面図を画像として出力することが望まれます。

HTML の表の利用

各駒の画像をそれぞれ独立したファイルとして用意します。形式としては gif jpeg png　のうような HTML でよく利用されている形式が良いでしょう。

HTML のページの中に９ｘ９のテーブルを作ります。（単純化のため、３ｘ３で例示）

```
<TABLE frame="box" rules="all" cellspacing="0">
<TR>
<TD><img src="fu.gif"></TD>
<TD><img src="fu.gif"></TD>
<TD><img src="fu.gif"></TD></TR>
<TR>
<TD><img src="spc.gif"></TD>
<TD><img src="kaku.gif"></TD>
<TD><img src="spc.gif"></TD></TR>
<TR>
<TD><img src="kyosha.gif"></TD>
<TD><img src="keima.gif"></TD>
<TD><img src="gin.gif"></TD></TR>
</TABLE>
```

盤面の駒配置を記した配列からこのような文字列を機械的に出力すれば盤面を描画できます。HTML ですから、改行は影響しません。

別の方法としては、固定的な HTML のテンプレートを作成し、画像ファイルをコピーやリンクによって HTML ファイル内の名前を対応させることも可能です。

```
<TABLE frame="box" rules="all" cellspacing="0">
<TR><TD><img src="31.gif"></TD>
<TD><img src="21.gif"></TD>
<TD><img src="11.gif"></TD></TR>
<TR><TD><img src="32.gif"></TD>
<TD><img src="22.gif"></TD>
<TD><img src="12.gif"></TD></TR>
<TR><TD><img src="33.gif"></TD>
<TD><img src="23.gif"></TD>
<TD><img src="13.gif"></TD></TR>
</TABLE>
```

初期配置なら 11.gif が後手の香車、21.gif が後手の桂馬、22.gif が後手の飛車になります。

HTML の記法についてはインターネット上に多数のガイドがあります。詳しくはそちらを参照して下さい。

１つの画像にまとめる

やはり各駒の画像を用意します。いくつか方法がありますが、歴史のある Netpbm の pnmcat を利用して結合する方法を紹介します。Netpbm は UNIX 系 OS でコマンドラインで使う道具集です。

pnmcat は Netpbm の形式である PBM PGM PPM PAM を入力とします。広く使われている PNG GIF JPEG TIFF などの画像形式は使えません。結合は左右方向なら -leftright または -lr 上下なら -topbottom または -tb です。

```
pnmcat -lr fu.pbm fu.pbm fu.pbm    > P7.pbm
pnmcat -lr spc.pbm kak.pbm spc.pbm > P8.pbm
pnmcat -lr kyo.pbm kei.pbm gin.pbm > P9.pbm
pnmcat -tb P7.pbm P8.pbm P9.pbm > partial_board.pbm
```

Netpbm のプログラムは一般に拡張子を考慮しないので次のようにもできます：

```
pnmcat -lr 91 81 71 61 51 41 31 21 11 > P1
pnmcat -lr 92 82 72 62 52 42 32 22 12 > P2
 ...
pnmcat -lr 99 89 79 69 59 49 39 29 19 > P9
pnmcat -tb P1 P2 P3 P4 P5 P6 P7 P8 P9 > board.ppm
```

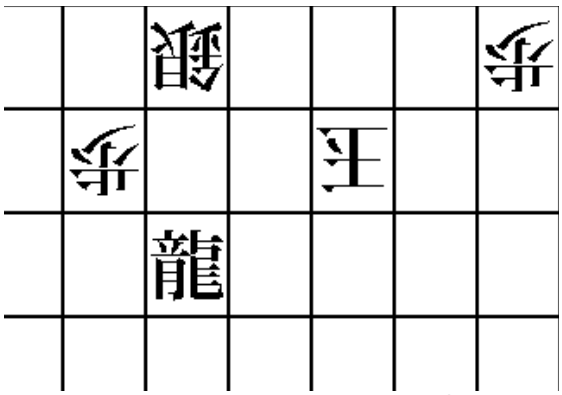
Netpbm には縦横のセルを一括で組み合わせる専用の命令もあります。

```
pamundice cell_%1d_%1a.ppm -across=9 -down=9 > board.ppm
```

引数は上下方向が先、左右方向が後０から始まります。
x 軸は右が大、y 軸は下が大です。

００	０１	０２	０３	０４	０５	０６	０７	０８
１０								１８
２０								
３０						３６		
４０								
５０								
６０							６７	
７０								７８
８０				８４				８８

pamundice の入力ファイルの引数



先手持ち駒：金

詰将棋 盤面を画像化　pamundice 利用

この pamundice は 2007 年に Netpbm に加えられたプログラムです。Debian と Ubuntu の Netpbm はそれより古い版を元にしているので使えません。Fedora 等では問題なく使えるはずです。

Netpbm は決まった画像方式しか扱えません。GIF PNG JPEG 等のフォーマットはそのままでは使えません。まず変換します。

```
giftopnm kin.gif > kin.pbm
giftopnm gin.gif > gin.pbm
giftopnm kei.gif > kei.pbm
giftopnm kyo.gif > kyo.pbm
```

giftopnm のところは PNG なら pngtopnm、JPEG なら djpeg または jpegtopnm となります。

個別に行うのは非効率で間違いの元です。一括して変換したいものです。シェルでは次の for 構文が一般的です：

```
for i in kin gin kei kyo
 do
 giftopnm $i.gif > $i.pbm
 done
```

しかし慣れないとなかなか上手くいかないものです。最初は変換をただちに行うのではなく、変換命令を列挙するプログラム（シェルスクリプト）を作成することを目指すと良いと思います。出力リダイレクトのところの引用符には注意して下さい。

```
for i in kin gin kei kyo
 do
 echo “giftopnm $i.gif > $i.pbm”
 done
```

この作業は使い慣れている言語で行うべきです。以下は awk の例：

```
ls *.gif | awk '{destfile=$1; sub(".gif",".pbm",destfile);
        print "giftopnm "$1" > "destfile}'
```

余白を足して大きさを揃えるには pnmpad 命令を使い縦横の画素数を指定します。元の画像はセンタリングされます。

```
giftopnm kin.gif | pnmpad -white -width=30 -height=40 > kin.pbm
```

外枠を追加するのにもやはり pnmpad を使います。

```
giftopnm kin.gif | pnmpad -white -width=28 -height=38 | \
        pnmpad -black -top=1 -left=1 -right=1 -bottom=1 > kin.pbm
```

後手の駒を作るため画像を 180 度回転するには pamflip 命令を使います。

```
pamflip -r180 sente_gin.pbm > gote_gin.pbm
```

強調するには白黒反転させるのが一つの方法です。

```
pnminvert kaku.pbm > hl_kaku.pbm
```

直接画像を書き出す

上記の２つ方法では既存の画像ファイルを貼り合わせる形で盤面の表示を行います。HTML 閲覧ソフトやコマンドライン（シェルから起動）の画像編集コマンドを利用するため間接的な出力と言えます。

盤面現状の内部データから直接画像ファイルを出力すれば効率的です。各種言語には画像を扱うための関数ライブラリが装備されています。c 言語の標準ライブラリ関数にはグラフィックスはありませんが、一定の知名度のある外部画像ライブラリとしては cairo などがあります。cairo には線や長方形の描画、文字の画像化の機能がありますので将棋盤面の画像化に使えます。出力されるファイルはデフォルトで PNG 形式となります。

cairo 等ライブラリを利用せず、全部自分で作成したい場合はまず画像形式を決める必要があります。画像圧縮を伴う形式は大変な手間がかかりますので非圧縮の単純な形式が適切です。直接書き出すには Netpbm の単純な画像形式が適しています。

```
P1
18 18
000000000000000000
000000001000000000
000000011100000000
000000110110000000
000001100011000000
000011000001100000
000110000000111000
011101111111101110
000000001000000000
000000001000000000
001111111111111100
000000001000000000
000100001000001000
000110001000011000
000011001000110000
000001001000100000
011111111111111110
000000000000000000
```

上記は「金」のフォントデータです。コピーしてファイルに書き込んで、それを画像閲覧ソフトで開くすると閲覧できます。

各ますを例えば横３０画素、縦４０画素とすると９ｘ９ますで盤は２７０ｘ３６０画素になります。この大きさの二次元配列を用意して上のようなフォントを０と１（アスキー英数字）で表示したのを

適切な所に入れていけば駒が配列された将棋盤が出来上がります。画素データー(ラスター）の前に次のヘッダーをつければ PBM 形式の画像です。

```
P1
270 360
```

この 270 と 360 は横と縦の画素数です。P1 を P4 とした場合、画素データはアスキー文字でなく２進数となります。８ビットに８画素が入り効率的です。しかしソースコードでの扱いは難しくなります。バイナリ形式では駒、ますの横幅を８や３２の倍数にすると扱いやすくなります。バイナリ形式でも 270 と 360 は同じです。

この方法の大きな課題は駒の絵のデータの入手と、０１の羅列への変換です。素材集で将棋の駒の絵が入手できるかもしれません。素材集は GIF や PNG 等の形式が主流ですから Netpbm の形式に変換する必要があります。例えば PNG 形式なら以下のようにします。

```
pngtopnm -plain keima.png > keima.pbm
```

オプションの -plain で画素データをバイナリでなく、アスキーで記す形式を指定します。pnmpad などの命令で画像を加工する場合は最後の命令に -plain をつけます。

PBM 形式は gimp,LibreOffice,Firefox はじめ多くのソフトウエアが対応しています。テキストエディタで作成してどこでも画像表示できる汎用性の高い形式です。ただ効率は低く、画像通信には向いていません。この他に単純な画像形式としては XBM XPM BMP がありますが、一番単純なのは PBM です。XBM と XPM は X－Window がよく対応していて、BMP は DOS-Windows で対応しているというメリットがありますがかなり複雑になります。

PBM 形式、並びに同系の PGM PPM PAM についての詳細はウィキペディアでお調べ下さい。

盤面画像化の方法の比較

３つの方法を簡単に紹介しましたが、どれが最良かは簡単には決められません。

プログラムを書く際に心がけるべきはコードの最小化です。ライブラリを使えば自前で用意するコードは少なくなりますが、汎用性の高いライブラリは機能豊富で大掛かりです。コードを最小化するとともに、ライブラリなど外部の機能を活用する場合は軽い物を使うべきです。

HTML を使う方法は一見簡単明瞭ですが、HTML を解釈するブラウザがある環境でしか使えません。ブラウザはパソコンやスマートフォンで標準装備なので問題は無い、という考え方もあります。しかしブラウザは X などのウインドウシステムを前提として、それはさらに多くの要素に依存しています。全体としては重くなります。

一枚の PBM 形式の画像にする方法なら、画像閲覧機能さえあれば盤面を表示できます。画像表示ソフトにはウインドウシステムを必要としないものもあります。フレームバッファーのフォーマットがわかれば、そこに直接書き込んで画像を表示することもできます。これは X-Window 等を介するより格段に効率の良い表示方法です。画像表示の負担が少ないということは将棋ソフトの性能を競う大会でそれだけ指し手思索に割り当てられる CPU の動作時間が多くなり、有利になるということです。

PBM からフレームバッファーへの形式の変換が必要ですが、PBM 形式は単純なので変換プログラムはすぐに書けます。フレームバッファーの利用方法については文末参考文献をご覧下さい。

詰将棋を解くアルゴリズム

詰将棋のルール

詰将棋は駒の期盤面配置、持ち駒の状況が示され、これを詰ます手順を問うものです。先手攻め方の手は必ず王手です。王手のできない局面になったら、攻めは失敗です。後手玉方はなるべく長い手数逃亡を続けようとします。二歩や打ち歩詰めの禁止は普通の将棋と同じです。問題図に無い駒は全て後手の持ち駒で自由に使えるものとされています。合駒してもすぐ取られるだけの場合は手数の勘定に入れません。

普通詰将棋には攻め方の王将はありません。先後両方の王が盤面に現れる例外的な問題は「双玉問題」と呼ばれます。攻める方はうっかり自分の王が取られないよう注意しなければならず、難易度の高い問題となります。

アルゴリズムの核心：候補手列挙とミニマックス法

まず将棋の駒の動きを理解した上で局面ごとに可能な手（候補手）を列挙するプログラムを書きます。先手用に王手となる指し手を判定選別、後手用に王手から逃れる指し手を判定選別するサブルーチンを書きます。この際、反則となる手を判定して候補手から除去するようにします。必須ではありませんが、盤面を画像化するプログラムはあった方が作業がしやすいので用意しておくべきです。

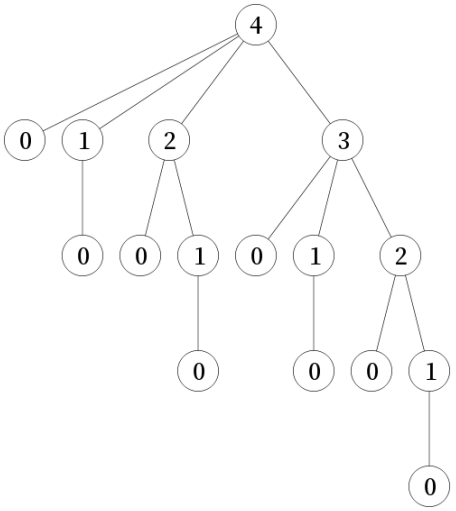
先手と後手で異なる手続きとなりますが、双方とも目標は明確なので少々作業量はありますがそれほど難しい課題ではありません。ここまでは中級の課題です。

これが完成すれば、先手後手それぞれが手を発見したら実際に駒を動かし、次の盤面を生成して相手に手を渡す、というのを交互に行う機構を構築することになります。この統括機構はやや難しいでしょう。

上記のルールの所で解説しましたが、詰将棋は詰めば正解、というのではありません。先手も後手も最善の手を指し続けた場合の均衡点を正解とします。先手は最短の手数で確実に相手玉を詰めようとし、後手はなるべく詰を先延ばしにしようとします。この種のせめぎ合いは「ミニマックス法」と呼びます。

ミニマックス法は文章を読んだだけではなかなか理解できません。実際に図を描いて納得するのが一番です。そのためには簡単なゲームを例にします。

ここでは石取りゲームを例に考えます。石が４個あり、そこから二人で交互に石を取り、最後の一個を取った方が負けとします。石は何個取るのも自由ですが、０個（言い換えれば「パス」）はだめです。このゲームの先手後手の可能な手を全て図に描きます。これは「ゲーム樹」(game tree)と呼ばれる図です。結節点は「ノード」(node)、終端は「葉」(leaf)と言います。多数に分岐しているノードで自分にとって最適と思われるてを選びゲームを進行させます。



丸の中がその時々の石の数です。最上段が先手、その下が後手と各段が先手後手交互の手です。０個になった所はゲームの終わりです。

この例の石取りゲームは結末が自明です。これではゲームにならないので実際には石の数を多くし、一度に取れる石の数に上限を設定します。

しかしここは敢えて結果が分かり切った例を使います。まだよく理解できない概念を調べる場合、例は単純な物から始めるべきです。詰将棋でも同様の図を描くことができますが、簡単な問題でも候補手（分岐）が多くて大きな図になってしまい、作成作業が大変です。作成が大変だと放棄してしまい「図を使わずに理解しよう」との考えに到達します。図を見ながらの理解と、図を見ずに頭の中で漠然とイメージしながらの理解とは違うものです。

石４個から始める石取りには初期状態で石３個の石取り、さらにその中には初期状態で石２個の石取りがあります。詰将棋も同様に５手詰の中に３手詰があり、さらにその中に１手詰があります。このような入れ子構造には再帰的関数呼び出し(recursive function call)が有効です。

この石取りゲームでは「必勝」「必敗」がはっきりしています。詰将棋ソフトを自作する時は最初は「詰む」「逃げられる」でも良いと思いますが、正しい答を出すには詰む場合も点数をつけて序列をつけます。基本的に詰むまでの所用手数が点数となりますが、最後に攻め方の駒が余る場合は調整します。

開発の考慮点

詰将棋の問題集には問題とともにその解答が記されていますが、多くの問題には別解があります。そのため「７手詰」とされる問題なら、ミニマックスの規準を満たし、７手で詰む手順を全て列挙してテキストの答えがその中にあれば正解とします。正解手順が複数あり、一つだけ示せば良いのではないところが少々難しいかもしれません。１手詰の問題はその性質上別解はありません。３手詰は別解をよく見ます。

将棋・詰将棋には細かいルールがあります。最初は細部にはこだわらず本筋を捉えるようにする方が良いと思います。本来打ち歩詰めは禁止ですが、最初はこのルールは対応せず、後で判定機能を追加すればよいでしょう。その他に詰みの時点での駒余り、無駄な合駒による引き伸ばし等は最初は対応しないでもよいでしょう。問題によっては本来の正解と違うものが正解と判定されますが、妥協しましょう。双玉問題も最初は考えないで良いと思います。

細かいルールに対応するより、候補手の列挙と王手の判定、王手からの逃避の列挙のプロセスを工夫し磨き上げる方が有意義です。これらがしっかりしていれば、残る課題はミニマックス法に従う全体構成にそれを組み込むことだけとなり、完成は見通せるでしょう。

一方で将棋の細かいルールに完全に対応した詰将棋ソフトなら、新聞の連載棋譜の投了局面を入力して詰みまでの手順を調べさせることができるようになります。実現したら楽しいでしょう。

攻め方が王手を発見したらすぐそれを実行する形で次の局面を生成し、守備方の手番にする構成と、王手を全部リストにして、それを一つずつ処理して行く構成が考えられます。後者の方が見通しの良いプログラムになるはずです。待ち行列を利用した可変長のリストを扱う技術が必要になります。

全体構成が難しく感じられるなら、再帰、待ち行列、ミニマックス法を十分に理解できていないことが原因と思われます。互除法による最小公倍数の発見、二分樹検索や石取りゲーム等、簡単なプログラミング例題をこなして慣れることです。そしてまず１手詰を解くプログラムを書き、次に３手詰、さらに５手詰と発展させて様子をつかんでから一般化に挑戦することです。

再帰や待ち行列は有用な方法ですが何か間違いがあると始末が悪いものです。途中経過を報告するしくみを入れておくとよいでしょう。

人工知能への発展

かつては詰将棋を解くプログラムも人工知能の一種と見做されていました。最近は「人工知能は自動学習を行うもの」との定義を採用されるようになり、従来型のプログラミング手法で解決できる詰将棋は人工知能ではないとする人が増えています。

しかしこれまで述べて来たミニマックス法に準拠した逐次探索で詰将棋を解くプログラムと自動学習が全く無関係かというと、そうではありません。自動学習で機能を付加して性能を向上する方法があります。

将棋は相手の王を詰みにするゲームですが、すぐには捕まらないので途中では自陣を固める、相手の駒を取る、自分の駒を進めて成駒を作るなどを目的とします。将棋ソフトはこうしたを成果を評価して、詰みが遠い序盤中盤の局面の有利不利を判定し、なるべく良い手を指すようになっています。

詰将棋でも途中の成果を測定することができます。相手の駒を取れば戦力はその分落ちます。できれば強い駒を取りたいものです。一方で自分の駒は取られないようにします。「王は下段に落せ」という格言があるように相手の王を端の方へ逐うのが良い攻め方です。反対に攻められる方としては敵の駒の少ない方、広い盤の中央に逃げる手を考えるものです。少し将棋がわかる人なら一目で「詰まない」「逃げられない」と識別できる局面もコンピュータは几帳面に調査して判定します。駒の勢力圏（効き）や駒同士の連絡状況等を数値指標化して、個々の手が有望かそうでないか、傾向を把握できるプログラムを目指します。ここはおおよその判断で良いのです。評価判定関数の作成に自動学習エンジンを使うのも良いでしょう。

見通しの良い構成にするのに役立つのが全候補手の待ち行列です。待ち行列になっていれば要素を並び変えることができます。順序を変えただけでは正解が見つからなくなることはありません。開発の順序として、まず間違いなく王手と詰みを見つけ出す、反則手を指さないプログラムを完成させてから、改善の余地のある領域を見つけ、その部分を他としっかり区別した上で磨いて行くと良いでしょう。将棋ソフトを良い物にするには何度も改良を加えて行くことになります。一部を変更しても他に影響が及ばない、ことにコンピュータの「間違いを犯さない」機能が損なわれない構成を目指して下さい。

謝辞

本稿は 2020 年５月１６日開催予定の OSC 名古屋にてセミナーで発表する予定の内容をまとめたものです。この OSC 大会はコロナウイルス肺炎の感染拡大防止策の移動や集会の自粛要請で中止になってしまいました。ここに主催者のご厚意により文章での発表の場を頂きました。深く御礼申し上げます。

強い将棋ソフトは何度も改良を加えて実現するものです。使いやすいコンパイラ・インタプリタ、ライブラリその他開発機材を開発作成して万人が自由に利用できる形で提供して下さった方に感謝しなければなりません。

本稿が技術者ならびに情報技術を学ばれる皆様のお役に立つものでしたら幸甚です。

２０２０年６月　漆畑晶

参考資料

棋譜フォーマット

csa 棋譜フォーマット

http://www2.computer-shogi.org/wcsc12/record.html （旧版）

http://www2.computer-shogi.org/protocol/record_v22.html （2.2 版）

kif 棋譜フォーマット

http://kakinoki.o.oo7.jp/kif_format.html

詰将棋問題集

詰将棋の問題はインターネット上に多数発表されています。

日本将棋連盟のサイトでは毎日新しい問題が掲載されています。

日本将棋連盟

https://www.shogi.or.jp/

棋書情報サイト 棋書ミシュラン

http://rocky-and-hopper.sakura.ne.jp/Kisho-Michelin/index.htm

詰将棋集書籍が多数紹介されています。詰将棋問題集の手数別内訳があります。難問をお探しなら「大道将棋」の部の本をご検討下さい。

画像の直接出力

GUI の無い環境での画像表示

https://www.ospn.jp/osc2019-fukuoka/pdf/OSC2019_Fukuoka_Netpbm.pdf