JAPLA研究会資料 2008/6/28

JによるWhite & Blackパズルとその探索プログラム

西川　利男

１．はじめに

先月、先々月の２回の例会でＪによる８クィーンのパズルを取り上げた[1],[2]。今回はWhite & Black (WB)パズルを取り上げる。実はこの問題はAPLの著、"APL2 at a Glance"[3]の中のPuzzle-Solving Problemの章として、APL2のコーディング付きで紹介されているものである[3]。しかし、Ｊで行うにはいくつかの修正、変更が必要である。さらに、一般的な木の探索の２方法[4]のうち８クィーンではDepth-first Search(縦型探索)であったのに対して、WBパズルではBreadth-first Search(横型探索)であるなど、アルゴリズムとしてもおもしろい問題である。

２．White & Black (WB)パズルの遊び方

WBパズルとは次のように白(White)と黒(Black)の碁石(原著ではmarble,おはじき)を使って、はじめに次のように並べる。

```
WW_BB
```

そして白または黒の碁石と空いた場所（_で示す）と位置を入れ換えて並びかえる。つまり最初の手としては次の

```
_WWBB  W_WBB  WWB_B WWBB_
```

の４通りのパターンが可能である。

これを次々と繰り返し、最後に

```
BBWW_
```

というパターンに到達するにはどうすればよいか？、というパズルである。

２．White & Black (WB)パズルの解の探索プログラム

一般にこのようなパズルやゲームのプログラムでは、次々と変化する手の状態をどう表現するかが、プログラム設計の鍵となる。先の８クィーンではタテヨコの位置の座標値で表した。

幸いなことに、今回のWBパズルでは文字列をそのまま使って表現できる。

文献

[1] 西川利男「Ｊによる８クィーンーその1」JAPLA研究会資料2008/4/26

[2] 西川利男「Ｊによる８クィーンーその２、解の探索プログラム」JAPLA研究会資料2008/5/24

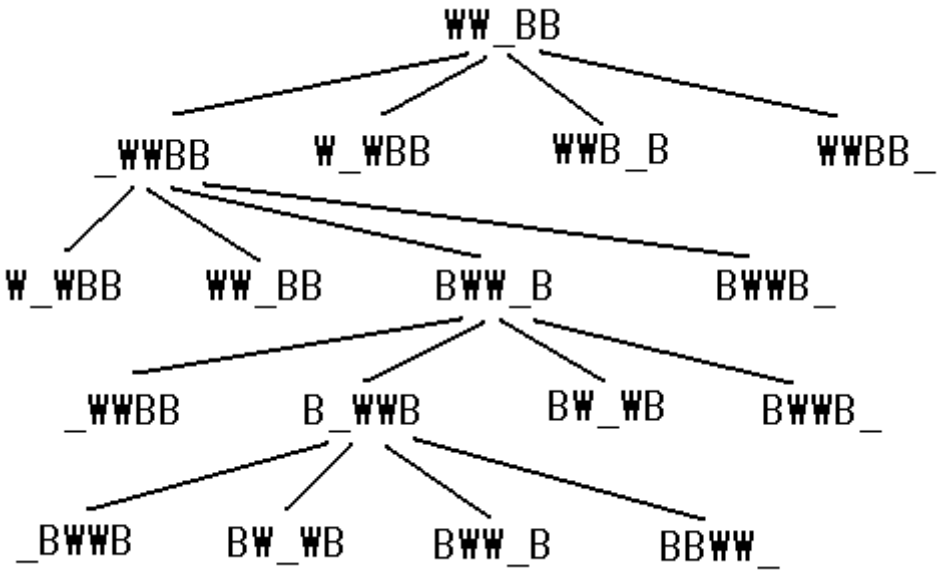
[3] J.Brown, S.Pakin, R.Polivka, "APL2 at a Glance", p.359-370, Pretice-Hall(1988).

[4] A.Aho, J.Hopcroft, J.Ullman, "Data Structures and Algorithms", p241-243, Addison-Wesley (1983).

すなわち、任意の一つの手は'W'(白), 'B'(黒), '_'(空き)の文字を使って、５文字から成る文字列として、たとえば次のように簡単に表現できる。

WW_BB

そして、いろいろな手の動きの過程は文字列の集合で表される。また、探索のルートは次のような木構造で図示される。



上の図から分かるように、

WW_BB → _WWBB → BWW_B → B_WWB → BBWW_

は一つの解である。

プログラムは次のようになる。一つの手の状態に対して、定義した動詞 move により次に可能な手を生成する。Jのコーディングは付録を参照のこと。

例えば'WW_BB'に対しては

```
   move <'WW_BB'
+-----+-----+-----+-----+
|_WWBB|W_WBB|WWB_B|WWBB_|
+-----+-----+-----+-----+
```

また、'_WWBB'に対しては

```
   move <'_WWBB'
+-----+-----+-----+-----+
|W_WBB|WW_BB|BWW_B|BWWB_|
+-----+-----+-----+-----+
```

のようになる。

一方、それまでの探索のルートを保存する探索データベース M を設け、手の動きのたびに更新していくようにする。

たとえば

(<_WWBB><WW_BB>)(<W_WBB><WW_BB>)(<WWB_B><WW_BB>)(<WWBB_><WW_BB>)

のようになる。

なお、ボックスの多重化のため、表示が見にくくなるのをさけるため、上の表示は新たに定義した動詞 ndisplay により表示している。

これらの準備をした上で、WBパズルの解の探索とは以下ような処理をおこなうことに他ならない。

つまりmoveにより可能な新しい手を求め、すでにデータベースMにあるものは除外して、次々と手を進める。このようにして、手を進めて行きつつ、最終のパターンBBWW_ が見つかったら「メデタシ！」となる。

解析のため、途中経過を出力するようにした探索プログラムsear1で、そのようすをみてみよう。なおNEWP0は動詞moveによる可能な手で、NEWP1は探索データベースMを元にNEWP0から既訪問を除外した新しい手である。このNEWP0を動詞joinxによりMに追加して探索データベースを更新する。

```
   sear1 BEGIN
I= 1
B:
+-----+
|WW_BB|
+-----+
NEWP0:
+-----+-----+-----+-----+
|_WWBB|W_WBB|WWB_B|WWBB_|
+-----+-----+-----+-----+
NEWP1:
+-----+-----+-----+-----+
|_WWBB|W_WBB|WWB_B|WWBB_|
+-----+-----+-----+-----+
M:
(<_WWBB><WW_BB>)(<W_WBB><WW_BB>)(<WWB_B><WW_BB>)(<WWBB_><WW_BB>)
I= 2
B:
+-----+
|_WWBB|
+-----+
NEWP0:
+-----+-----+-----+-----+
|W_WBB|WW_BB|BWW_B|BWWB_|
+-----+-----+-----+-----+
NEWP1:
+-----+-----+
|BWW_B|BWWB_|
+-----+-----+
M:
(<W_WBB><WW_BB>)(<WWB_B><WW_BB>)(<WWBB_><WW_BB>)(<BWW_B><_WWBB><WW_BB>
)(<BWWB_><_WWBB><WW_BB>)
I= 3
B:
+-----+
|W_WBB|
+-----+
NEWP0:
```

```
+-----+-----+-----+-----+
|_WWBB|WW_BB|WBW_B|WBWB_|
+-----+-----+-----+-----+
```

NEWP1:

```
+-----+-----+
|WBW_B|WBWB_|
+-----+-----+
```

M:

(<WWB_B><WW_BB>)(<WWBB_><WW_BB>)(<BWW_B><_WWBB><WW_BB>)(<BWWB_><_WWBB><WW_BB>)(<WBW_B><W_WBB><WW_BB>)(<WBWB_><W_WBB><WW_BB>)

（途中省略）

I= 24

B:

```
+-----+
|BWB_W|
+-----+
```

NEWP0:

```
+-----+-----+-----+-----+
|_WBBW|B_BWW|BW_BW|BWBW_|
+-----+-----+-----+-----+
```

NEWP1:

```
+-----+
|BW_BW|
+-----+
```

M:

(<WBB_W><W_BBW><WWBB_><WW_BB>)(<BBWW_><B_WWB><BWW_B><_WWBB><WW_BB>)(<BBW_W><B_WBW><BWWB_><_WWBB><WW_BB>)(<_BWWB><WB_WB><WBW_B><W_WBB><WW_BB>)(<_BWBW><WB_BW><WBWB_><W_WBB><WW_BB>)(<B_BWW><BWBW_><_WBWB><WWB_B><WW_BB>)(<BW_WB><BWBW_><_WBWB><WWB_B><WW_BB>)(<_BBWW><WBBW_><W_BWB><WWB_B><WW_BB>)(<BW_BW><BWB_W><_WBBW><WWBB_><WW_BB>)

I= 25

B:

```
+-----+
|WBB_W|
+-----+
```

NEWP0:

```
+-----+-----+-----+-----+
|_BBWW|W_BBW|WB_BW|WBBW_|
+-----+-----+-----+-----+
```

NEWP1:

M:

(<BBWW_><B_WWB><BWW_B><_WWBB><WW_BB>)(<BBW_W><B_WBW><BWWB_><_WWBB><WW_BB>)(<_BWWB><WB_WB><WBW_B><W_WBB><WW_BB>)(<_BWBW><WB_BW><WBWB_><W_WBB><WW_BB>)(<B_BWW><BWBW_><_WBWB><WWB_B><WW_BB>)(<BW_WB><BWBW_><_WBWB><WW

```
B_B><WW_BB>)(<_BBWW><WBBW_><W_BWB><WWB_B><WW_BB>)(<BW_BW><BWB_W><_WBBW
><WWBB_><WW_BB>)
I= 26
B:
+-----+
|BBWW_|
+-----+
Congratulation !!
I=26
Root to Goal:
(<WW_BB>)(<_WWBB>)(<BWW_B>)(<B_WWB>)(<BBWW_>)
```

3. 実行例

探索プログラム search1 の実行例をあげる。

```
   search1 BEGIN
Root to Goal:
(<WW_BB>)(<_WWBB>)(<BWW_B>)(<B_WWB>)(<BBWW_>)

Root to Goal:
(<WW_BB>)(<_WWBB>)(<BWWB_>)(<B_WBW>)(<BBW_W>)(<BBWW_>)

Root to Goal:
(<WW_BB>)(<WWB_B>)(<_WBWB>)(<BWBW_>)(<BW_WB>)(<BWW_B>)(<B_WWB>)(<_BWWB
>)(<BBWW_>)

Root to Goal:
(<WW_BB>)(<WWB_B>)(<W_BWB>)(<WBBW_>)(<_BBWW>)(<WBB_W>)(<WB_BW>)(<W_BBW
>)(<WWBB_>)(<_WBBW>)(<BW_BW>)(<B_WBW>)(<BBW_W>)(<BBWW_>)

*** end ***
```

付記

White & Black パズルの J プログラムの大半は５月中にすでに出来上がっていた。６月４日から１３日までのスペイン旅行から帰ってきて、その後２，３日は時差やらスペインぼけで J に触れる余裕もなく、まだ未完成で少々あわてた。グラナダのアルハンブラ宮殿やコルドバのメスキータのイスラム・だんだら模様が WB パズルと夢の中でダブったりと…、何とかプログラムが完成して、ほっとした。

プログラムの解説はあまり丁寧にはできなかったが、以下あげた J のリスティングにより検討されたい。

付録　プログラム・リスティング

```
NB. White and Black Marble Puzzle
NB.   cited from Brown, Pakin, Polivka, "APL2 at a Glance", p. 360
NB.   programmed by T. Nishikawa 2008/5/9
NB.   completed 2008/6/17 after the travel in Spain

wr =: 1!:2&2
rd =: 1!:1

BEGIN =: 'WW_BB'
END =: 'BBWW_'

NB. Usage => search1 BEGIN
search1 =: 3 : 0
B =. y.
M =. ,<,< y.
I =. 1
label_LOOP.
  if. 0 = #M do. '*** end ***' return. end.
  B =. {. , > {. M
  if. (<END) -: B
    do.
NB.  wr 'Congratulation !!'
NB.  wr 'I=',":I
     G =: |. > {. M
     wr 'Root to Goal:'
     ndisplay  G
     if. 0 < #rd 1 do. return. end.
     B =. y.
     I =. 0
    end.
  NEWP0 =. move ,> B
  MREM =. -. NEWP0 e. , >M
  NEWP =. MREM # NEWP0
  M1 =.  {. M
  if. 1 = #M1 do. M1 =. > M1 end.
  M3 =. }. M
  M =. M3, NEWP joinx M1
  I =. I + 1
goto_LOOP.
)
```

```
swap =: 3 : 0
:
D =. x.
'p q' =. y.
r =. p { D
D =. (q{D) p } D
D =. r q } D
)

move =: 3 : 0
D =. > y.
bl =. D i. '_'
nb =. (i. #D) -. bl
I =. 0
DN =. ''
while. I < #nb
 do.
  DN =. DN, < (>D) swap bl, (I{nb)
  I =. I + 1
end.
DN
)

joinx =: 3 : 0
:
z =. ''
Y =. > y.
Y =. }: ,Y,"(1) ' '
I =. 0
while. I < #x. do.
  z =. z, < ;: (,>I{x.),' ', Y
  I =. I + 1
end.
z
)

ndisplay =: 3 : 0
Z =. ''
I =. 0
while. I < #y.
  do.
NB.     wr I{y.
```

```
    DI =.  , (>'<', L:0 > I{y.),"(1)'>'
    Z =. Z, '(', DI, ')'
    I =. I + 1
  end.
wr Z
)

NB. Analysing Version ============================================

car =: {.
cdr =: }.


BE0 =: 'B_WWB'
BE1 =: 'BWW_B'
BE2 =: '_WWBB'
BE3 =: 'BWWB_'

NB. usage => sear1 BE1 !!OK!! 2008/5/9
NB. calling subroutines
NB.    move, swap
NB.    joinx, diam
NB.    ndisplay
sear1 =: 3 : 0
B =. y.
M =. ,<,< y.
I =. 1
label_LOOP.
  wr 'I= ', ":I
  if. 0 < #rd 1 do. 'break' return. end.
  if. 0 = #M do. '*** end ***' return. end.
  wr 'car M:'
  wr {. M
  wr 'B:'
  wr B =. {. , > {. M
NB.  if. (<END) -: B do. '*** end ***' return. end.
  if. (<END) -: B
    do.
     wr 'Congratulation !!'
     wr 'I=',":I
     G =: |. > {. M
     wr 'Root to Goal:'
```

```
      ndisplay  G
      if. 0 < #rd 1 do. 'break' return. end.
      B =. y.
      I =. 0
      wr 'NEW B:'
      wr < B
     end.
  wr 'NEWP0:'
  wr NEWP0 =. move ,> B
  MREM =. -. NEWP0 e. , >M
  wr 'NEWP1:'
  wr NEWP =. MREM # NEWP0
  wr 'M:'
  M1 =.  {. M
  if. 1 = #M1 do. M1 =. > M1 end.
  M3 =. }. M
  M =. M3, NEWP joinx M1
  ndisplay M
  I =. I + 1
goto_LOOP.
)
```