ウツリギの解析

柴原 一友　但馬 康宏　小谷 善行
東京農工大学

概要
本稿では，二人零和完全情報ゲームの一種であるウツリギの解を得る実験を行い，先手必勝であることを得た．また，より小さな盤面における解の傾向を見ることで，ゲームの特性について調査した．また，評価関数の違いによる探索速度の違いを調査し，その傾向を明らかにした．

Solving Utsurigi

Kazutomo Shibahara　Yasuhiro Tajima　Yoshiyuki Kotani
Tokyo University of Agriculture and Technology

Abstract
We experimented to obtain a solution of Utsurigi that is a kind of zero-sum two person games of perfect information and clarified the sure-fire method for winning of Black. Moreover, we investigated characteristics of the game by the tendency to solutions in the game of smaller boards. In addition, we investigated the relation between the evaluation function and the search speed, and clarified the tendency.

1．はじめに

近年，ゲームを解く研究が多くなされてきている[1]．ゲームを解くことは，ゲーム研究の最終目標の一つであるといえる．人間がコンピュータに勝てないレベルにあるゲームは数多くあるが，その中で解が得られていないゲームも多く存在する．また，昨今提案されたゲームも多く存在し，その解を得る研究も行われている[2]．

近年提案されたゲームにウツリギがある．ウツリギとは二人対戦型パズルであり，二人零和完全情報ゲームである．現在では「ダブルストン」という名前で学研から発売されている[3]．本稿では，その解を得る実験を行った．また，より小さな盤面における解を求め，その傾向を調べた．

2．ウツリギのルール

今回実験で使用したウツリギのルールは次のとおりである．

- NxN の盤面を使用する
- 黒と白の 2 人プレイヤーによる対戦である．
- 使用するピースは二種類存在し，それぞれ 1x1，1x2 である．
- 初期盤面は四隅にそれぞれのプレイヤーの 1x1 ピースを対角線になるように置く．
- ゲームは黒の手番から開始する．
- 手番側のプレイヤーは盤上のピースに対し，ピースの辺が隣接するようにピースを配置する．
- このとき，相手のピースに辺が隣接した場合は，そのピースを裏返し，自分側のピースに変える．
- ひとつピースを置いたら，相手に手番を移す．
- お互いがピースを置けない状態になったら終了とする．
- 終了のとき，盤上の升目の占有率が多いほうのプレイヤーの勝ちとする．

4x4 の盤面における初期配置を図 1 に示す．5x5，6x6 の場合でも同様の形で四隅に配置する．

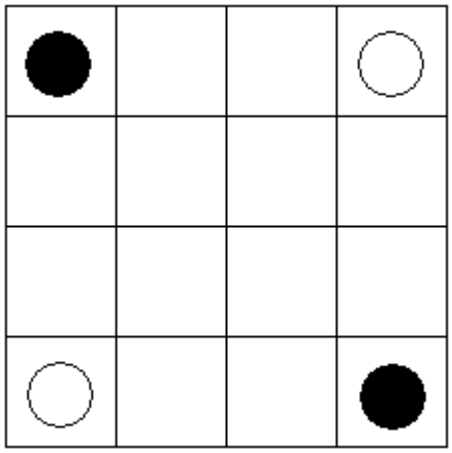


図 1　4x4 ウツリギの初期配置

市販されているダブルストンにおける正式なルールでは，6x6 の盤面であり，1x1 と 1x2 のピースを同数，つまり各 6 個使用する．

3. 実験

実験では，AND/OR 木探索を使用し，ウツリギの解を得た．反復深化は使用せず，単純にゲーム木の末端まで探索して解を得ている．対象は 4x4，5x5，6x6 の盤面である．探索は評価関数を用いて候補手のソートを行っている．使用した評価式を式 1 に示す．

E = 手番側の駒の占有数 — 相手側の駒の占有数　… 式 1

また，トランスポジションテーブルを使用している．過度のトランスポジションテーブルへの保存を避けるため，登録する局面は，それ以下の探索されたノード数が 10 より大きい局面に限っている．実験結果を表 1 にしめす．

実験の結果，ピースの配分によって解がかなりばらつくことがわかる．ピースの数によって戦略が大きく異なっていることが予想される．ピースの数を念頭においた戦い方が必要になる，頭を使うゲームであることが予想される．

表 1 ウツリギの解実験結果

盤のサイズ	1x2 ピース数	1x1 ピース数	解	ノード数
4x4	0	6	白必勝	177
	1	4	白必勝	10083
	2	2	黒必勝	249
	3	0	白必勝	324
5x5	0	11	黒必勝	362809
	1	9	黒必勝	1923406
	2	7	白必勝	657731
	3	5	白必勝	433356
	4	3	黒必勝	219951
	5	1	引き分け	135615

また，正式ルールでの実験も行った．盤面は 6x6 であり，ピースは大小各 6 個である．ただし，1x1 のピースはあらかじめ二つ盤面に配置されるため，所持しているのは 4 個だけである．

この実験の結果，黒必勝であることが得られた．必勝となる初手は，1x2 のピースの短い辺を盤面につける置き方である．総探索ノード数は 92807383 ノードであった．付録として，正式ルールで解いたときの出力結果（深さ 5 まで）と，解の例を示す．

4. 評価関数の変更

解を得る速度は候補手の並び替えに強く依存する．上に述べた評価関数を用いない場合，5x5 においても解を得ることは難しくなる．そこで，評価関数の性能をより向上させる方法について考え，5x5 の問題を解く実験を行った．

評価関数の変更にあたり使用するのは，取られることのないピースの占有量である．これは，図 2 のような状況のとき，左上隅の黒のピースは取られることはない．このようなピースは有利不利を確定する重要な要素であると考え，このピースの量を考慮した関数を作成した．ここで，取られることのないピースを固定ピースと呼び，取られる可能性のあるピースを非固定ピースと呼ぶ．

新たに使用する評価式を式 2 に示す．

E =（x ＊ 手番側固定ピース占有量 ＋y ＊ 手番側非固定ピース占有量） －
（x ＊ 相手側固定ピース占有量 ＋y ＊ 相手側非固定ピース占有量） …式 2

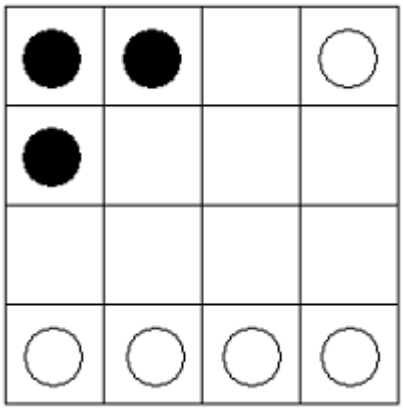


図 2 取られることのないピースの例

ここで，x と y は定数である．この式を用いて 5x5 の問題を解いた結果の，総探索ノード数を表 2 に，探索時間を表 3 に示す．

表 2 と表 3 において，なし(処理せず)と x=1,y=1 は同じ評価関数を用いた状態となるため，総探索ノード数に差異はない．しかし，固定ピースを求める処理が入っているかいないかの違いがあり，両者の探索時間は異なる．

表 3 から，固定ピースを求める処理は現時点で 10%程度の速度増加を招いていることがわかる．現時点では，局面を遷移する際に，指した手の周辺にあるピースが固定ピースであるかをひとつひとつ調べる方法を用いている．これを，ピースにそれぞれ背番号を付与し，各ピースの周辺の空き数を保存する方法などを用いることにより，処理速度の向上を行うことができるかもしれない．

表 2　新しい評価関数を用いたときの 5x5 の問題における総探索ノード数

	ピース数(1x1 ピース数,1x2 ピース数)					
	0,11	1,9	2,7	3,5	4,3	5,1
なし(処理せず)	362809	1923406	657731	433356	219951	135615
x=1,y=1	362809	1923406	657731	433356	219951	135615
x=2,y=1	517429	2662782	942862	576395	254259	172013
x=3,y=1	1382890	5414689	1885579	1173683	314038	233795
x=1,y=2	378246	1672324	392058	221573	228661	114704
x=1,y=3	407802	1814871	394388	180696	245653	119797
x=0,y=1	537852	2517576	492149	229682	295894	124828
x=1,y=0	11283371	80318141	138947468	113407472	4347857	2834392

表 3　新しい評価関数を用いたときの 5x5 の問題における探索時間

	ピース数(1x1 ピース数,1x2 ピース数)					
	0,11	1,9	2,7	3,5	4,3	5,1
なし(処理せず)	1.328	6.922	2.359	1.641	0.938	0.687
x=1,y=1	1.515	8.047	2.782	1.953	1.125	0.860
x=2,y=1	2.062	10.844	3.937	2.609	1.312	1.047
x=3,y=1	5.265	21.296	7.609	5.172	1.578	1.407
x=1,y=2	1.640	6.985	1.656	1.016	1.187	0.734
x=1,y=3	1.750	7.500	1.672	0.844	1.265	0.750
x=0,y=1	2.203	10.016	2.000	1.015	1.484	0.781
x=1,y=0	40.468	297.625	537.235	472.797	19.515	13.422

表 2 をみると，平均的に見て総探索ノード数が少ないのは x=1,y=2 の時である．つまり，非固定ピースに重みを置いたほうがよい結果になる．逆に固定ピースだけ(x=1,y=0)評価に使用する場合，探索ノード数は大幅に膨れ上がっている．一方，非固定ピースだけ(x=0,y=1)の場合は，それほど悪化しない．これにより，固定ピースは形勢を判断する主たる要素になりえないことがわかる．

表 3 をみると，一見してもっとも優れた解となっているのは，なし（処理せず）であるが，有利に働いた場合の速度向上が大きいため，平均でみるとわずかに x=1,y=2 の方が優れている．わずか 6 種類の結果であるため，明確によいとはいえないが，固定ピースを数える処理のオーバヘッドが小さくなれば，x=1,y=2 の方が高速に処理できることが予想される．

並べ替えをするときに，深さ 2 の探索を実行した結果で並べ替えた場合の総探索ノード数についても調べた．その結果を表 4 に示す．

表 4 深さ 2 の探索結果を用いて並べ替えを行ったときの総探索ノード数

	ピース数（1x1ピース数,1x2ピース数）					
	0,11	1,9	2,7	3,5	4,3	5,1
x=1,y=1	458943	2603936	1397093	1280284	532054	393916
x=2,y=1	920232	4485465	2491771	2185290	746066	553578
x=3,y=1	2068360	9678491	5474165	4792993	967809	696807
x=1,y=2	488782	1856271	810582	636471	504554	301312
x=1,y=3	515570	1993926	822931	541877	541935	287269
x=0,y=1	756836	2890004	1038147	677631	559449	309799
x=1,y=0	17476859	122209423	302952153	257226184	9370501	5057709

なお，この探索ノード数の値に並べ替えの際に行われた探索ノード数は含まれていない．しかしながら，ほぼすべてにわたって探索ノード数が上昇しており，探索ノード数の傾向は前の実験と変わっていない．よって，相手側が手を終えた段階の評価値の信頼性は低いと考えられる．

5. 評価関数の考察

固定ピースは決して取られることのないピースであるため，これが多いほうが有利であると考えるのが自然である．しかし，実際にはむしろ非固定ピースが多いほうが有利な結果になることが示された．この理由のひとつとして，図 2 のような局面を有利と判断してしまうことがあげられる．この局面では，実際には白の方が多く，範囲が広いため有利になり，黒はかたまった配置となるため不利となる．しかし，固定ピースを優先的に考えると有利と判断してしまう．

また，非固定ピースが多いほど，広範囲にピースが配置される状況になることも，非固定ピースの価値を高めている要因であると考えられる．囲碁などのように，手を抜いて範囲を広げていくことが，このゲームにおいても重要であると思われる．人間が実戦する場合においても，固定ピースにとらわれない戦い方が，勝つ上で重要であると考えられる．

深さ 2 の探索で並べ替えを行った場合，探索ノード数が増加する結果となった．これは，一手進むごとに評価値が変動しやすいため，相手の手が終わった状態では，相手に有利な状況が生まれやすいためではないかと考えられる．通常，相手の手を終えた段階で評価をする方が有効に働くが，将棋にたとえるならば，取り合い探索を導入していない状況であるために，相手に有利な状況へと進んでしまうのではないかと思われる．ただし，取り合いが落ち着く点を探すのは難しく，取り合い探索のような探索を単純に実装できるわけではない．

6. おわりに

実験の結果，4x4，5x5 のウツリギの解を得た．その解がピースの配分によって大きく異なっていることを示した．また，正式ルールの解が黒必勝であることを示した．新たな評価関数についても設計し，取られることのないピースを作ることはあまり重要ではないことを示した．今後，より大きな盤面を行うためには，今回実装しなかった，反転した盤面の同一化や，証明数探索の導入などがあげられる．

参考文献

[1] Herik, H. J. van den, J. Uiterwijk, and J. van Rijswijck: Game Solved Now and in the Future, Artificial Intelligence Journal 134:pp.277-312, (2002).

[2] 田中哲郎: ボードゲーム「シンペイ」の完全解析，ゲーム情報学研究会 2006-GI-15:pp.65-72, (2006).

[3] ダブルストン: http://shop.gakken.co.jp/shop/order/k_ok/goodsdisp.asp?code=918051

付録

付録 1 において，正式ルールにおける深さ 5 までの結果を示している．深さがひとつ深くなるごとにインデントがなされている．表記はまず，(a,b)で指定されており，横 a，縦 b の位置にピースを置くことを意味している．置くピースは 1x1，1x2 の二種類存在し，1x2 ピースの場合は向きが存在する．横向きにおく場合は，(a,b)と(a+1,b)に重なるようにピースを置くことを意味し，縦向きに置く場合は，(a,b)と(a,b+1)に重なるようにピースを置くことを意味する．見易さのため，白の一手目の頭にはハイフンをつけている．

また，付録 2 では，探索木の局面の中のいくつかの遷移例を図に示している．深さごとにインデントがなされており，同じ列の盤面は同じ深さの手である．右端(レイアウトとしては下端)にはその局面の最終的な解が示されている．

(1,2)に 1x2 を横に置く
-(1,4)に 1x2 を縦に置く
(1,3)に 1x2 を横に置く
(2,5)に 1x2 を縦に置く
(2,4)に 1x2 を横に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(4,1)に 1x2 を横に置く
(2,1)に 1x2 を横に置く
(5,1)に 1x2 を縦に置く
(3,2)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(2,5)に 1x2 を横に置く
(5,1)に 1x1 を置く
(2,5)に 1x2 を縦に置く
(6,2)に 1x1 を置く
(2,5)に 1x2 を縦に置く
-(1,5)に 1x2 を横に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(3,4)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(1,3)に 1x2 を縦に置く
(5,1)に 1x2 を縦に置く
(3,2)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(4,1)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(5,1)に 1x1 を置く
(2,6)に 1x2 を横に置く
(6,2)に 1x1 を置く
(2,6)に 1x2 を横に置く
-(2,5)に 1x2 を縦に置く
(2,3)に 1x2 を縦に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x1 を置く
(1,5)に 1x1 を置く
(3,4)に 1x2 を縦に置く

(5,1)に 1x2 を縦に置く
(3,2)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(4,1)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(5,1)に 1x1 を置く
(1,4)に 1x2 を縦に置く
(6,2)に 1x1 を置く
(1,4)に 1x2 を縦に置く
-(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(1,5)に 1x2 を横に置く
(3,5)に 1x2 を横に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x2 を横に置く
(4,1)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(5,1)に 1x2 を縦に置く
(3,2)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(1,5)に 1x1 を置く
(2,5)に 1x2 を横に置く
(5,1)に 1x1 を置く
(1,5)に 1x2 を横に置く
(6,2)に 1x1 を置く
(1,5)に 1x2 を横に置く
-(4,1)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(2,1)に 1x2 を横に置く
(1,5)に 1x2 を横に置く
(2,3)に 1x2 を縦に置く
(2,5)に 1x2 を縦に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く

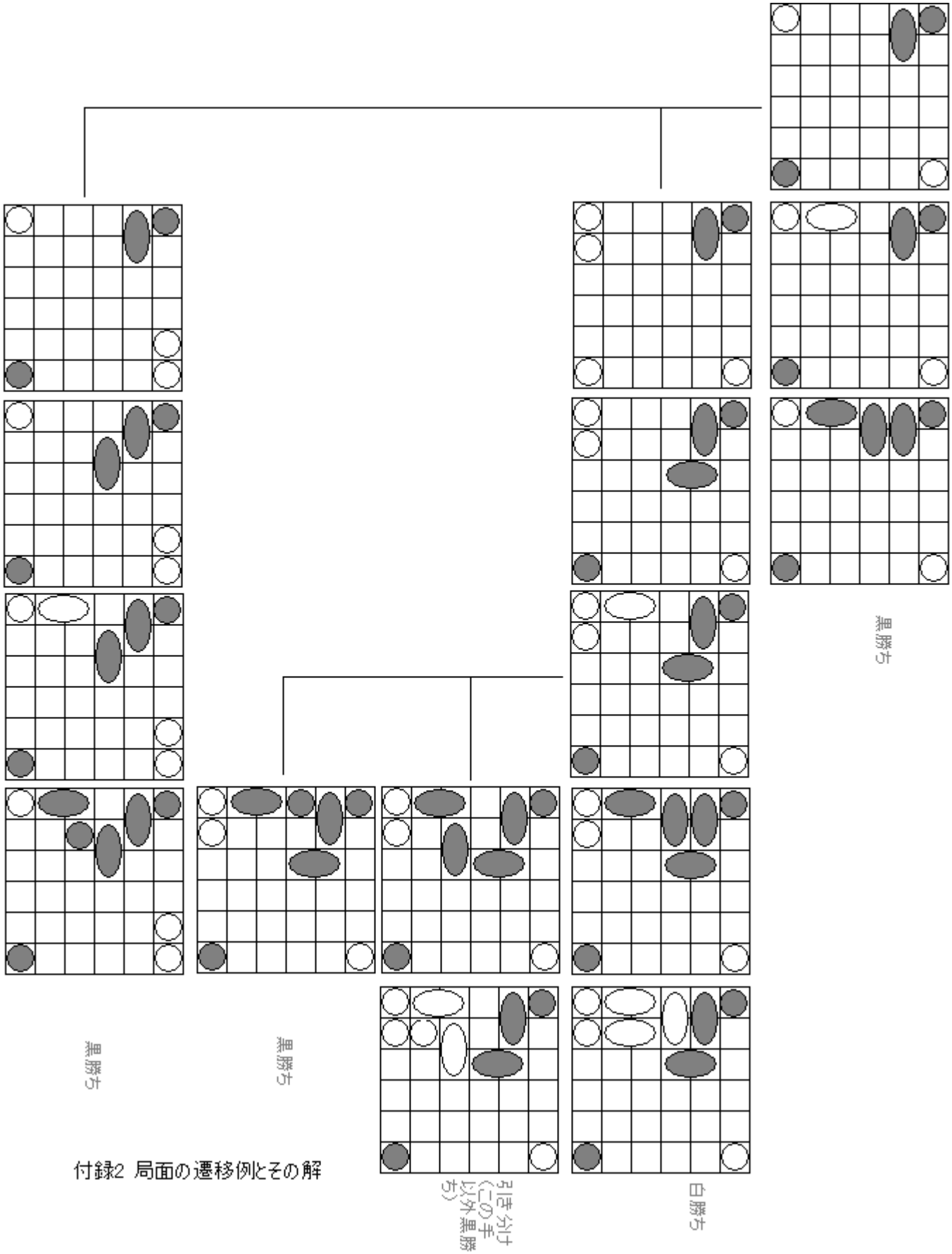
(6,4)に 1x2 を縦に置く
(1,5)に 1x1 を置く
(5,2)に 1x2 を横に置く
(2,6)に 1x1 を置く
(5,2)に 1x2 を横に置く
(6,2)に 1x1 を置く
(5,2)に 1x2 を縦に置く
-(5,1)に 1x2 を縦に置く
(3,2)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(4,3)に 1x2 を横に置く
(6,2)に 1x1 を置く
(4,3)に 1x2 を横に置く
(2,5)に 1x2 を縦に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x2 を横に置く
(1,5)に 1x2 を横に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(6,2)に 1x2 を縦に置く
(1,5)に 1x1 を置く
(6,2)に 1x2 を縦に置く
-(5,2)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(4,1)に 1x2 を横に置く
(5,1)に 1x1 を置く
(4,3)に 1x2 を横に置く
(2,5)に 1x2 を縦に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x2 を横に置く
(1,5)に 1x2 を横に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(4,1)に 1x2 を横に置く
(1,5)に 1x1 を置く
(4,1)に 1x2 を横に置く
-(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(5,1)に 1x2 を縦に置く

(5,3)に 1x2 を縦に置く
(1,5)に 1x2 を横に置く
(2,3)に 1x2 を縦に置く
(2,5)に 1x2 を縦に置く
(2,3)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(4,1)に 1x2 を横に置く
(5,2)に 1x2 を縦に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x2 を横に置く
(1,5)に 1x1 を置く
(5,1)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(5,1)に 1x2 を縦に置く
(5,1)に 1x1 を置く
(5,2)に 1x2 を縦に置く
-(1,5)に 1x1 を置く
(1,3)に 1x2 を縦に置く
(2,5)に 1x2 を縦に置く
(2,4)に 1x2 を横に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(4,1)に 1x2 を横に置く
(2,1)に 1x2 を横に置く
(5,1)に 1x2 を縦に置く
(3,2)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(3,2)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(2,5)に 1x2 を横に置く
(5,1)に 1x1 を置く
(2,5)に 1x2 を縦に置く
(6,2)に 1x1 を置く
(2,5)に 1x2 を縦に置く
-(2,6)に 1x1 を置く
(3,2)に 1x2 を縦に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x1 を置く
(1,5)に 1x2 を横に置く
(3,4)に 1x2 を縦に置く
(2,4)に 1x2 を縦に置く
(1,3)に 1x2 を縦に置く

(2,5)に 1x2 を横に置く
(3,4)に 1x2 を横に置く
(3,5)に 1x2 を縦に置く
(4,6)に 1x2 を横に置く
(3,6)に 1x2 を横に置く
(5,5)に 1x2 を縦に置く
(4,1)に 1x2 を横に置く
(2,1)に 1x2 を横に置く
(5,1)に 1x2 を縦に置く
(4,3)に 1x2 を横に置く
(5,2)に 1x2 を横に置く
(4,3)に 1x2 を横に置く
(6,2)に 1x2 を縦に置く
(6,4)に 1x2 を縦に置く
(1,5)に 1x1 を置く
(1,3)に 1x2 を縦に置く
(2,5)に 1x1 を置く
(3,4)に 1x2 を縦に置く
(3,6)に 1x1 を置く
(4,6)に 1x2 を横に置く
(5,1)に 1x1 を置く
(4,2)に 1x2 を横に置く
(6,2)に 1x1 を置く
(4,2)に 1x2 を横に置く
-(5,1)に 1x1 を置く
(2,3)に 1x2 を横に置く
(1,4)に 1x2 を縦に置く
(2,4)に 1x1 を置く
(1,5)に 1x2 を横に置く
(1,3)に 1x2 を縦に置く
(2,5)に 1x2 を縦に置く
(3,4)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(3,1)に 1x2 を横に置く
(4,2)に 1x2 を縦に置く
(4,1)に 1x2 を縦に置く
(2,1)に 1x2 を横に置く
(4,2)に 1x2 を横に置く
(4,3)に 1x2 を縦に置く
(5,2)に 1x2 を横に置く
(4,2)に 1x2 を縦に置く
(5,2)に 1x2 を縦に置く
(4,2)に 1x2 を縦に置く
(6,2)に 1x2 を縦に置く

(6,4)に 1x2 を縦に置く
(1,5)に 1x1 を置く
(1,3)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(2,4)に 1x2 を縦に置く
(4,1)に 1x1 を置く
(4,2)に 1x2 を縦に置く
(5,2)に 1x1 を置く
(4,3)に 1x2 を横に置く
(6,2)に 1x1 を置く
(1,3)に 1x2 を縦に置く
-(6,2)に 1x1 を置く
(2,3)に 1x2 を横に置く
(1,4)に 1x2 を縦に置く
(1,3)に 1x1 を置く
(1,5)に 1x2 を横に置く
(1,3)に 1x2 を縦に置く
(2,5)に 1x2 を縦に置く
(3,4)に 1x2 を縦に置く
(2,6)に 1x2 を横に置く
(4,6)に 1x2 を横に置く
(4,1)に 1x2 を横に置く
(2,1)に 1x2 を横に置く
(4,2)に 1x2 を横に置く
(4,3)に 1x2 を横に置く
(5,1)に 1x2 を縦に置く
(4,3)に 1x2 を横に置く
(5,2)に 1x2 を縦に置く
(4,3)に 1x2 を縦に置く
(5,3)に 1x2 を横に置く
(6,4)に 1x2 を縦に置く
(6,3)に 1x2 を縦に置く
(5,5)に 1x2 を横に置く
(1,5)に 1x1 を置く
(1,3)に 1x2 を縦に置く
(2,6)に 1x1 を置く
(2,4)に 1x2 を縦に置く
(5,1)に 1x1 を置く
(5,2)に 1x1 を置く
(4,3)に 1x2 を横に置く
(6,3)に 1x1 を置く
(6,4)に 1x2 を縦に置く

付録1 正式ルールの解決木



付録2 局面の遷移例とその解