

修士学位論文  
一般化三目並べの拡張における  
勝敗判定に関する研究

東北大学 大学院情報科学研究科  
システム情報科学専攻 篠原研究室  
博士課程前期二年の課程  
本田 耕一

2011年2月14日

# 目次

<b>第 1 章</b>	<b>序論</b>	<b>1</b>
1.1	パズル・ゲームの解析	1
1.2	計算機の利用	1
1.3	本論文の内容	2
<b>第 2 章</b>	<b>先行研究・関連研究</b>	<b>4</b>
2.1	一般化三並べ	4
2.1.1	勝ち型	5
2.1.2	負け型	5
2.1.3	未解明 Snaky	9
2.2	一般化三目並べの拡張	9
2.2.1	ゲーム OR	10
2.2.2	ゲーム AND	10
2.2.3	ゲーム $n$ 細胞 OR	11
2.3	ゲーム木探索	13
2.3.1	証明数探索	14
2.3.2	弱証明数探索	15
2.3.3	Proof tree tracing	16
<b>第 3 章</b>	<b>計算機による勝ち型判定</b>	<b>20</b>
3.1	証明数探索の利用	20
3.2	勝ち型の判定	21
3.3	探索の効率化	23
3.4	「どんな畳敷き戦略でも負け型と証明できない」ことの判定	29

<b>第 4 章</b>	<b>ゲーム OR, ゲーム AND, ゲーム <math>n</math> 細胞 OR における進展</b>	<b>30</b>
4.1	ゲーム OR の勝ち型判定 . . . . .	30
4.2	ゲーム AND の勝ち型判定 . . . . .	32
4.3	ゲーム $n$ 細胞 OR の先手 4 近傍限定時における上限 . . . . .	33
<b>第 5 章</b>	<b>ゲーム NOTAND</b>	<b>35</b>
5.1	ゲーム NOTAND の提案 . . . . .	35
5.2	ゲーム NOTAND の考察 . . . . .	35
5.2.1	勝ち型の証明 . . . . .	36
5.2.2	負け型の証明 . . . . .	36
5.3	ゲーム NOTAND の結果 . . . . .	37
<b>第 6 章</b>	<b>まとめと今後の課題</b>	<b>39</b>
6.1	証明数探索の利用 . . . . .	39
6.2	ゲーム OR・ゲーム AND . . . . .	39
6.3	ゲーム $n$ 細胞 OR . . . . .	40
6.4	ゲーム NOTAND . . . . .	40
6.5	一般化三並べ全般 . . . . .	40
<b>参考文献</b>		<b>41</b>

# 第1章

## 序論

### 1.1 パズル・ゲームの解析

我々が日常娯楽としているパズル・ゲームの多くは，離散数学において重要な性質を有しており，「娯楽数学」として素人，専門家を問わず数学を楽しむことのできる題材となっている [28]. 例えば，ハノイの塔では再帰アルゴリズムから得られる差分方程式を解くことで，「円盤の移動手順の最小回数は  $2^n - 1$  である」ことを求めることができる．また 15 パズルにおいては，1 から 15 の数字の置換を考えることにより，「任意の 2 枚のタイルを入れ替えた初期状態は解くことができない」ことを証明できる．

更にパズル・ゲームはそのルールを自然に拡張することができ，より数学的な問題とすることが可能である．例えば，天秤問題におけるコインの枚数やハノイの塔の杭の本数を，任意の自然数  $n$  と置いて一般化させた問題が研究の対象となっている [27].

近年，ますます様々なパズル・ゲームが研究対象とされるようになっているなかで，本論文は  $\circ \times$  ゲームあるいは三目並べ，五目並べを一般化したゲームである「一般化三並べ」とそれを拡張したゲームを対象とし，考察・解析を行う．

なお，本論文のタイトルでは「一般化三目並べ」としたが，本論文では過去の研究に倣い「一般化三並べ」と呼ぶこととする．

### 1.2 計算機の利用

パズル・ゲームでは状態をノード，行動を辺とした有向グラフ (ゲーム木) でパズル・ゲーム全体を表現可能である．数学的な完全解析が難しい，あるいは難しかったパズ

ル・ゲームにおいては、計算機を利用したゲーム木の探索がなされている。計算機を利用することにより、 $15 \times 15$  盤の五目並べは先手必勝であることが確かめられた [5]。その他にも、状態数が  $15!/2 = 653,837,184,000$  である 15 パズルの最適解の上限は 80 手であることが示され [6]、また状態数が  $(8! \times 3^8) \times (12! \times 2^{12}) / (2 \times 2 \times 3) = 43,252,003,274,489,856,000$  であるルービックキューブでも 2010 年 7 月に 20 手あることが示された [1]。また将棋では、2010 年 10 月に強豪プログラム 4 種の多数決からなる「あから 2010」が、清水市代女流王将（対局当時）に挑戦することが話題となり、結果は見事に計算機が勝利を収めた [2]。このように計算機性能の上昇と探索手法の効率化により、難解なゲームも徐々に解析できるようになってきている。

チェスや将棋を始め、本論文で扱う一般化三並べ等の二人ゲームにおいて必勝手順が存在することの証明には、自分の合法手のうちどれか一つで勝利を示し（OR 手順）、相手の合法手すべてで勝利を示す（AND 手順）AND/OR 木の探索が必要となる。そこで本論文では、詰将棋を始めとした AND/OR 木探索で成果を挙げている証明数探索を利用して一般化三並べの解析を行う。

### 1.3 本論文の内容

2 章で関連研究、先行研究について紹介する。まず本論文で対象とする一般化三並べについて「勝ち型」や「負け型」の定義と、その証明技法について解説する。そして、その拡張であるゲーム OR、ゲーム  $n$  細胞 OR、ゲーム AND を紹介し、既存研究における勝敗判定の結果も併せて述べる。ゲーム AND とゲーム  $n$  細胞 OR は、ゲーム OR の拡張ととらえることができ、一般化三並べとゲーム OR の関係と同様にゲーム OR の結果を他の 2 つのゲームの勝敗判定に利用することができる。

また 2 章では「勝ち型」の判定に必要な、ゲーム木探索のアルゴリズムである証明数探索について述べる。特に一般化三並べのグラフ構造である Directed Acyclic Graph (DAG) の AND/OR 木探索に有効な弱証明数探索と、類似した証明木の探索に有効な proof tree tracing を解説する。

そして、3 章で証明数探索を一般化三並べに適用することを考える。一般化三並べでは、盤の大きさが無限大というルールから、無限に決着がつかない場合が存在する。

従って一般化三並べの完全な探索では、探索中に終端ノードが存在せず、探索が終了しない。そこで、先手は有限の範囲のみ石を置けることとし、さらに後手を有利にした条件を付加して勝ち型判定を行うことを可能とする。

また、「どんな畳敷き戦略でも負け型と証明できない」ことの判定も AND/OR 木探索で可能であることを示す。

4 章では、ゲーム OR、ゲーム  $n$  細胞 OR、ゲーム AND における進展を述べる。ゲーム OR、ゲーム AND には計算機による勝ち型判定を用いた。ゲーム OR では既存研究以上には勝ち型を見つけることができなかったが、ゲーム AND では、66 組中 16 組新たに勝ち型を見つけることができた。またゲーム  $n$  細胞 OR では、先手が石を置ける場所を先手石の 4 近傍に限定された場合、17 細胞以上の動物を作ることができないということが証明する。

更に 5 章では、一般化三並べの新たな拡張として、作ってはいけない動物（禁止動物）を定めたゲーム NOTAND を提案する。この禁止生物の導入により後手にも必勝手順が出てくることや、勝ち型が負け型になる組合せが存在する可能性があり、より複雑なゲームとなる。残念ながら本論文では、そのような例を見つけることができなかったが、単独で勝ち型の組に対する結果を示す。

そして最後に 6 章でまとめと今後の課題を述べる。以上のことから得られる本研究の結果は以下の 4 つである。

- (1) 証明数探索を利用した一般化三並べの勝敗判定。
- (2) ゲーム AND における勝敗判定。
- (3) ゲーム  $n$  細胞 OR における先手 4 近傍限定時の上界の更新。
- (4) ゲーム NOTAND の提案とその勝敗判定。

## 第2章

### 先行研究・関連研究

本章では、本論文の準備として、一般化三並べとその拡張に関する既存研究について、また解析に用いる探索アルゴリズムについて述べる。

#### 2.1 一般化三並べ

二人ゲームとしてよく知られているゲームに○×ゲーム（三目並べ）や五目並べ、五目並べを競技用にルールを変更した連珠がある。これらのゲームはルールがシンプルで古くから世界中で親しまれている基本的なゲームである。

一般化三並べは、これらのゲームがフランク・ハラリイにより一般化されたゲームである [10, 12, 11]。三目並べは両者が最善を尽くした場合すぐに引き分けと分かってしまうが、ハラリイの一般化により数学的素材となる面白いゲームとなった。三目並べの一般化とは、三目並べのルールのうち以下の3点の変更をいう。

- (1) 盤の大きさを  $3 \times 3$  から無限大にした点。
- (2) 作る形を3連から任意の形を定めるとする点。
- (3) 定めた形の  $90^\circ$  ごとの回転とその反転は同一の形とするが、 $45^\circ$  ごとの回転とその反転は同一の形とは認めない点。

この3点を変更した一般化三並べは以下に定義される。

**定義 1 (一般化三並べ)** 盤の大きさが無限大の碁盤状の盤面に二者が交互に石を一つずつ置き、予め定められた連結した石で定義されるある動物（形）を、斜めは許さずに回転と反転を許して先に作った方が勝ちというゲームである。

ここで定められた形を**動物**と呼び、その動物を作るために占有しなければならないマスの個数を**細胞数**と呼ぶ。一般化三並べでは一般に動物としてポリオミノ<sup>1</sup>が用いられる。ポリオミノごとに、両者が最善を尽くしたとき先手必勝である**勝ち型**か、無限に続けても決着がつかない**負け型**に分類することができる。このゲームの性質上、後手の必勝手順は存在しない。なぜなら、もし後手の必勝手順が存在したならば、先手はどこかに初手を置いた後、後手の必勝手順を行えば良いからである。以下では、黒石を先手の石、白石を後手の石と定める。

### 2.1.1 勝ち型

両者が最善を尽くしたとき、先手必勝である形を勝ち型と呼ぶ。勝ち型には、図 2.1 で示されるように、1 細胞から 5 細胞動物までの合計 11 個が知られていて、これらの形は先手が必ず勝てる形である。なお証明に関しては、先手必勝手順を示すことや計算機によって計算させる方法などがある [4]。勝ち型に関する研究としては、勝つために必要な最小盤面や最小手数を求める研究がある [29]。

### 2.1.2 負け型

両者が最善を尽くしたとき、無限に続けても決着がつかない形を負け型と呼ぶ。負け型には、図 2.1 で示されるように、4 細胞以上から存在し、6 細胞以上になると、5 細胞以下の小さな負け型を含んでいるために明らかに負け型に分類されるものも多く存在する。さらに、7 細胞以上になると必ず小さな負け型を含んでいる。証明に関しては、後手の引き分け戦略として有効な**畳敷き戦略**を用いて示す方法が知られている。負け型に関しては、ハンディキャップの研究があり、これは先手が先にいくつ石を置けば勝ち型となるかを求めるものである [13, 16]。

---

<sup>1</sup>いくつかの正方形が辺で接しているもの。回転したり、裏返して一致しているものは同じとみなす。



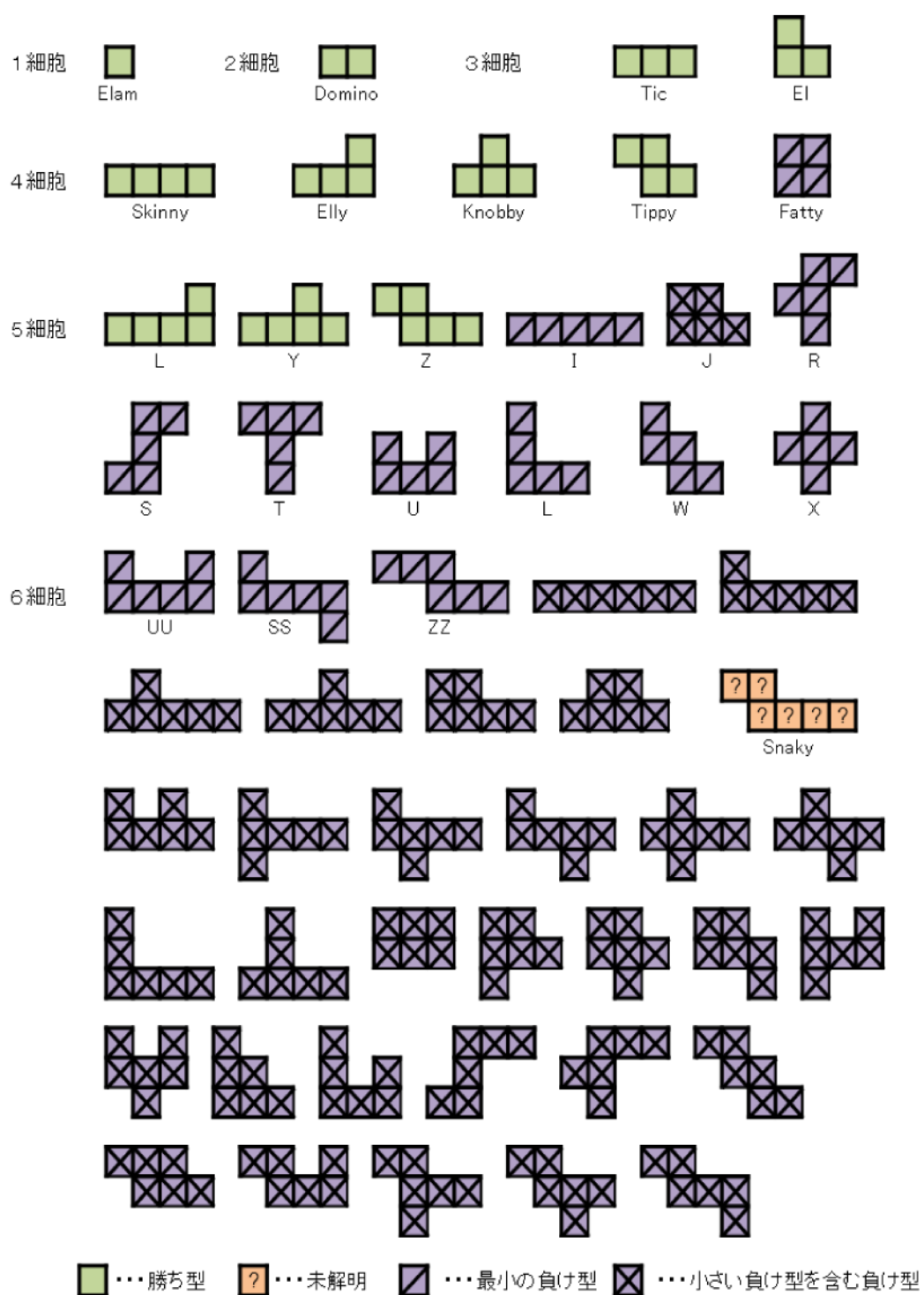


図 2.1: 一般化三並べの形の一覧

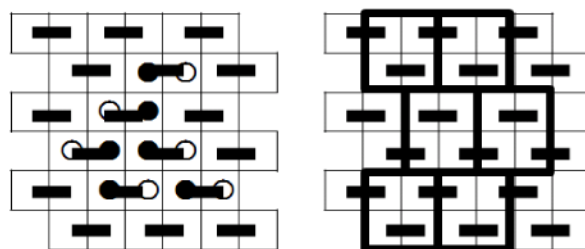


図 2.2: 畳敷きの例 (Fatty)

## 畳敷き戦略

畳敷き戦略は後手の引き分け戦略つまりは、負け型の証明として知られている手法である。盤面に2マス1組の「畳」を敷き詰め、後手の石は必ず先手の石が置かれた「畳」のもう一方のマ스에、1対1対応で置いていくという戦略である。

負け型であることの証明としては、盤面にある動物を作ろうとしたとき、どこにどの向きで作ろうとしても必ずその動物の中に畳が少なくとも1枚は含まれる事が確認できれば、後手の白石が必ず含まれてしまうためその動物は負け型と言える。そのため畳敷きは規則性を持っているものを考え、その規則性を考慮した十分な範囲内で必ず畳が含まれることを示せばよい。畳敷きの例として、Fattyが畳敷きにより防がれることを図2.2に示す。どの位置にFattyを作ろうとしても必ず畳が含まれてしまうことが確認できる。図2.3に主な畳敷きとその畳敷きにより負け型と証明できる動物を示す。

また畳敷き戦略の拡張として、部分的に動的戦略をとる畳敷き戦略が存在する。その例として、縦横7連と斜め8連を防ぐ後手の戦略を図2.4示す。その戦略とは、図2.4(a)の小盤面を考える。この小盤面は、図2.4(b)に示す9本の破線（縦3連、横4連、斜め3連）のいずれも先手に独占させないことができる。従って、図2.4(c)に示すように小盤面を敷き詰めると、縦横7連と斜め8連を防ぐことができるというものである。

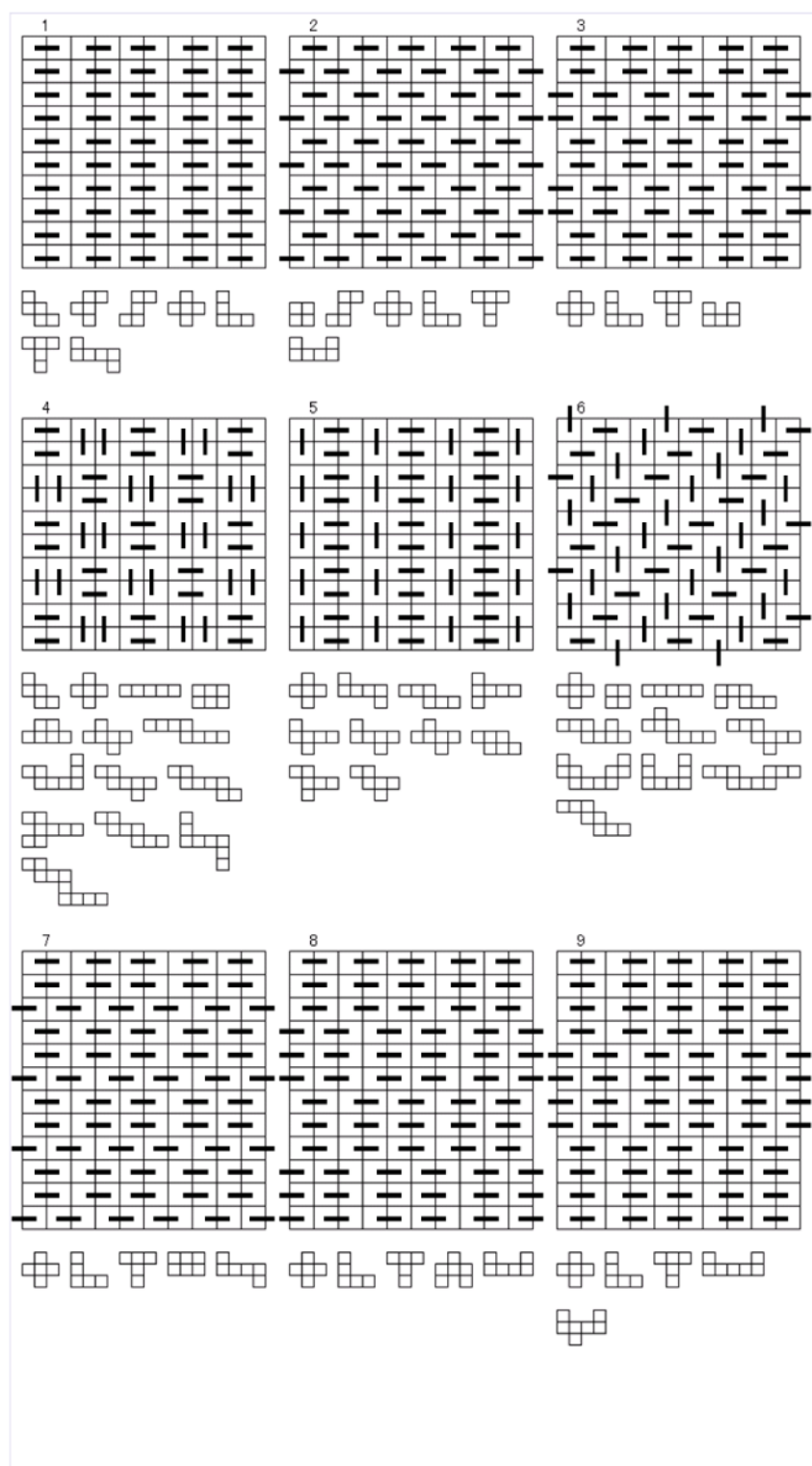


図 2.3: 主な畳敷きと負け型

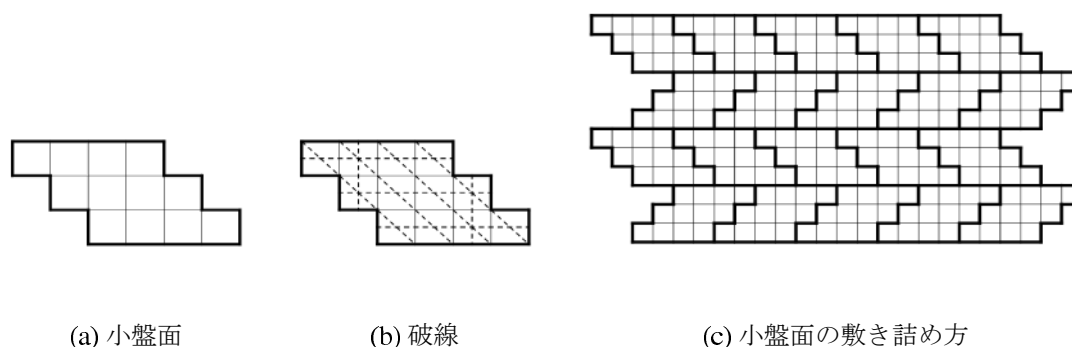


図 2.4: 縦横7連斜め8連を防ぐ畳敷き

### 2.1.3 未解明 Snaky

既存研究で勝敗について未解明な動物は1つだけ存在しており，図 2.1 で示されている6細胞の **Snaky** である．なお他の6細胞に関しては，図 2.1 で示されている通りすべて負け型となっている．一般化三並べでは特に未解明である **Snaky** に関して多く研究がなされており，後手が畳敷き戦略では負け型と証明できないことや，先手が黒石の4近傍に限定されると負け型なこと，ハンディキャップ数が1であることなどが知られている [9, 11, 14, 15, 18, 21].

## 2.2 一般化三目並べの拡張

一般化三並べの他にも，三目並べの変種は研究されており， $n$  目並べに関しては様々な制限を付加した問題に関する研究がある [20]. また一般化三並べの拡張も提案されており，盤のマス目を三角形，六角形にした問題に関する研究 [8, 17] や，作れば勝ちとなる動物を増やしたゲーム OR[32]，ゲーム  $n$  細胞 OR[31]，ゲーム AND[31] など，幅広く研究されている [7, 22, 23, 24, 25].

本論文では，一般化三並べを拡張したゲームのうち，ゲーム OR，ゲーム  $n$  細胞 OR，ゲーム AND に関して勝ち型判定を行ったため，以下ではその3つのゲームを説明する．

### 2.2.1 ゲーム OR

従来の一般化三並べでは単独の動物に対する勝敗判定や考察がなされてきたが、その自然な拡張の1つとして、作ればよい動物を増やしていくことが考えられる。ゲーム OR は、作れば良い動物を二体にし、そのうちの1つを先に作るというゲームで、次のように問題が定義される。

**定義 2 (ゲーム OR)** 盤の大きさが無限大の碁盤状の盤面に二者が交互に石を一つずつ置き、予め定められた連結した石で定義されるある動物の二つ組のどちらかの動物を、斜めは許さずに回転と反転を許して先に作った方が勝ちというゲームである。

ゲーム OR の二つ組に単独勝ち型の動物を含んでしまうと当然その二つ組は勝ち型であるため、負け型である動物を組み合わせなければ意味がない。ゲーム OR の勝ち型判定は、必ず勝てる部分的な局面（必勝パターン）を考え、その必勝パターンの手順を示す方法である。また負け型判定は、二つ組に共通する畳敷き戦略が存在するかどうかを調べる方法である。既存研究の勝敗判定の結果を、表 2.1 に単独負け型である動物のうち、他の負け型を含まない最小の 12 種類と未解明である Snaky の組合せ 78 種類について示す。表中にない 6 細胞以上の動物に関して、10 細胞動物どうしの二つ組は、すべて負け型であることが分かっている。

### 2.2.2 ゲーム AND

ゲーム AND は、定められた二つ組の両方の動物を先に作るとしたゲームである。従って、二つ組に単独負け型を組み合わせると負け型になるため、組み合わせる動物は勝ち型どうしを考える。その場合、単独で勝ち型をつくれればよいという点を考えれば、厳密には示されてはいないが先手必勝である可能性が高いため、ルールを付加して定義されている。

**定義 3 (ゲーム AND)** 盤の大きさが無限大の碁盤状の盤面に二者が交互に石を一つずつ置き、予め定められた連結した石で定義されるある動物の二つ組のどちらの動物も、斜めは許さずに回転と反転を許して先に作った方が勝ちというゲームである。ただし、二つの動物は 1 マス共有して作られなければならない。

表 2.1: ゲーム OR の勝敗

		×	○	○	×	×	×	○	×	×	○	○	○
			×	×	×	×	×	○	○	×	×	○	○
				×	×	×	×	○	×	○	×	○	○
					×	×	×	○	○	○	×	○	○
						×	×	×	×	×	×	×	△
							×	×	○	×	×	○	○
								×	△	×	×	○	○
									○	○	○	○	○
										△	△	△	△
											○	○	△
												×	△
													△

○ 勝ち型

× 負け型

△ 未解明























2つの動物を1マス共有して作れば勝ちということは、1マスを共有した新たな動物を全て考え、そのうちのどれか1つを作れば良いこととなる。従って、ゲーム AND はゲーム OR を拡張した問題といえる。よって勝敗判定もゲーム OR 同様に行うことができる。既存の結果として、表 2.2 に単独で勝ち型動物 11 種類の組合せ 66 組の勝敗判定を示す。

### 2.2.3 ゲーム $n$ 細胞 OR

ゲーム AND は二つ組によって定まる細胞数の等しい動物「 $m$  組」のゲーム OR であったが、ゲーム  $n$  細胞 OR は、「 $m$  組」を拡張し、細胞数  $n$  における  $n$  細胞動物全体を一つの組としたゲーム OR である。

**定義 4 (ゲーム  $n$  細胞 OR)** 盤の大きさが無限大の碁盤状の盤面に二者が交互に石を一つずつ置き、予め定められた細胞数  $n$  におけるすべての  $n$  細胞動物のうち任意の一つの動物を、斜めは許さずに回転と反転を許して先に作った方が勝ちというゲームである。

表 2.2: ゲーム AND の勝敗

											
	○	○	○	○	○	○	○	○	○	○	○
		○	○	○	○	○	○	○	○	△	△
			○	○	△	△	△	△	△	△	△
				○	△	○	△	△	△	△	△
					×	△	△	△	△	△	△
						△	△	△	△	△	△
							△	△	△	△	△
								△	△	△	△
									△	△	△
										×	△
											×

○ 勝ち型

× 負け型

△ 未解明

このゲームは  $n$  に対して勝ち型か負け型かが決まる。そして任意の  $n$  細胞動物を作ればよいということは、 $n$  個の石を任意に連結する仕方を考慮していることになる。つまり、ゲーム  $n$  細胞 OR は「 $n$  細胞動物を作ることができるか」、さらに「先手は黒石をいくつまで連結させることができるか」という問題として捉えることができる。

既存研究 [25] により、 $n$  細胞動物を囲い込む石の最小個数  $\varepsilon(n)$  は、式 2.1 で求められることが知られている。よって、13 細胞動物以下の生物は、(細胞数 - 1) 個以下の石で囲い込めないことから、 $n < 13$  では勝ち型である。

$$\varepsilon(n) = \lceil 2 + \sqrt{8n - 4} \rceil \quad (2.1)$$

また、先手を黒石の 4 近傍に限定した場合の上限として、19 細胞動物を作らせない後手の戦略があることが示されている。

## 2.3 ゲーム木探索

ゲームの状態空間は，全状態の集合  $N$ ，状態遷移（合法手）の集合を  $E$  とすると，有向グラフ  $G = (N, E)$  で表現することができる．このとき，ある状態をルートノードとして階層状にしたものを**ゲーム木**という．

一般化三並べの勝敗判定では，先手と後手が交互に石を置いていき，ある動物が勝ち型か負け型かを調べる．このような最終的な評価値が2種類しか存在しない二人ゲームのゲーム木探索は，**AND/OR 木**の探索として一般化できる．

**AND/OR 木**とは**AND ノード**と**OR ノード**の2種類のノードを持つ木である．各ノードの評価値は**真**か**偽**か**不明**である．子ノードを持つノードを**内部ノード**といい，子ノードを持たず，真または偽の評価値を持つノードを**終端ノード**と呼ぶ．**OR ノード**の子は必ず**AND ノード**であり，**AND ノード**の子は必ず**OR ノード**である．

内部ノードの評価値は以下のように定義される．

### (1) AND ノードの場合

- 少なくとも1つの子ノードが偽ならば，評価値は偽である．
- すべての子ノードが真ならば，評価値は真である．
- それ以外の場合，評価値は不明である．

### (2) OR ノードの場合

- 少なくとも1つの子ノードが真ならば，評価値は真である．
- すべての子ノードが偽ならば，評価値は偽である．
- それ以外の場合，評価値は不明である．

ノードの評価値が真であることを示すことを証明するといい，偽であることを証明することを**反証する**という．ノードが証明（反証）されたときには，ノードが真（偽）であることを証明する**証明木（反証木）**を持ち，複数の証明手段があれば証明木は複数存在する．証明木がわかれば，必勝手順を選ぶことが可能である．

**AND/OR 木探索**の目標はルートノードを評価することであり，より小さな証明木を見つけることが探索の性能となる．



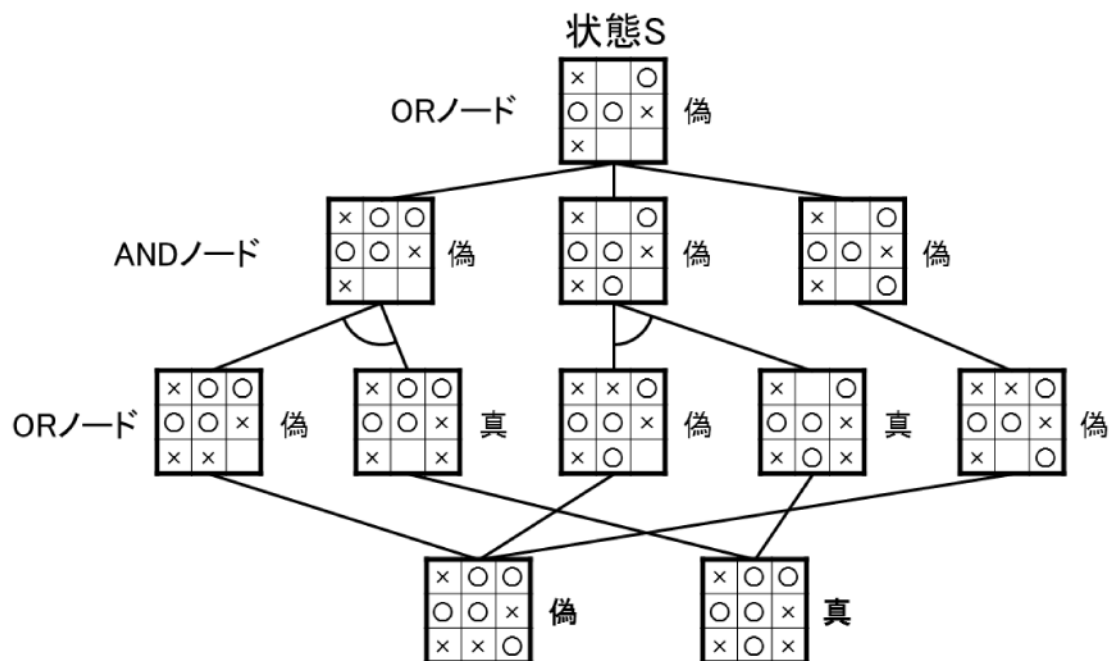


図 2.5: ○×ゲームのゲーム木

AND/OR 木の例として、状態  $S$  をルートノードに持つ○×ゲームのゲーム木を図 2.5 に与える．このゲーム木では、ある状態とその状態を回転・反転してできる状態を同一の状態としている．また状態には、定義に基づき評価値を付している．

### 2.3.1 証明数探索

証明数探索 (Proof-Number Search) [3] は、Allis らにより提案され、5 目並べの解析に使用された．証明数探索とは、**証明数**・**反証数**を用いた最良優先探索である．ノード  $n$  の証明数とは、 $n$  を証明するために最低限展開しなければならないノード数であり、反証数とは、 $n$  を反証するのに最低限展開しなければならないノード数である．

$n_1, n_2, \dots, n_K$  を内部ノード  $n$  の子ノードとすると、AND/OR 木の評価値の定義から、ノード  $n$  の証明数  $\mathbf{pn}(n)$  と反証数  $\mathbf{dn}(n)$  は次のように定義される．

(1)  $n$  が葉ノードの場合

- $n$  の評価値が真ならば、 $\mathbf{pn}(n) = 0, \mathbf{dn}(n) = \infty$ .

- $n$  の評価値が偽ならば,  $\mathbf{pn}(n) = \infty, \mathbf{dn}(n) = 0$ .
- $n$  の評価値が不明ならば,  $\mathbf{pn}(n) = \mathbf{dn}(n) = 1$ .

(2)  $n$  が内部 OR ノードの場合

- $\mathbf{pn}(n) = \min_{1 \leq i \leq K} \mathbf{pn}(n_i)$ .
- $\mathbf{dn}(n) = \sum_{1 \leq i \leq K} \mathbf{dn}(n_i)$ .

(3)  $n$  が内部 AND ノードの場合

- $\mathbf{pn}(n) = \sum_{1 \leq i \leq K} \mathbf{pn}(n_i)$ .
- $\mathbf{dn}(n) = \min_{1 \leq i \leq K} \mathbf{dn}(n_i)$ .

証明数探索では OR ノードでは証明数, AND ノードでは反証数が最小の子ノードを展開していく. そして証明数・反証数の定義から分かるように, 証明数・反証数は子ノードが複数の親を持たないことを仮定している. しかし, ゲーム木は親を複数持つ DAG (Directed Acyclic Graph) であることが多く, この定義をそのまま実際のゲーム木探索に使用すると証明数・反証数を 2 重カウントしてしまう問題がある.

また最良優先探索の場合, 展開したノードをすべて保持しておかなければならず, メモリが足りなくなるという問題もある. この問題に対しては, 多重反復進化法による df-pn[30] が提案され, ミクロコスモスと呼ばれる 1525 手詰めの詰将棋を解くなど非常に良い性能が示されている.

### 2.3.2 弱証明数探索

証明数・反証数の 2 重カウント問題の対策として長井の解決策 [30] が知られているが, 実装が難しく, 回避に時間がかかることから, 弱証明数探索 (Weak Proof-Number Search) が提案された [26]. 弱証明数探索は, 証明数・反証数の代わりに弱証明数・弱反証数を用いる探索である.

$n_1, n_2, \dots, n_K$  を内部ノード  $n$  の子ノード,  $k$  を  $n_1, n_2, \dots, n_K$  のうち証明 (反証) が済んでいない子ノードの数とすると, 弱証明数・弱反証数の定義は以下である.

(1)  $n$  が葉ノードの場合

- $n$  の評価値が真ならば,  $\mathbf{pn}(n) = 0, \mathbf{dn}(n) = \infty$ .
- $n$  の評価値が偽ならば,  $\mathbf{pn}(n) = \infty, \mathbf{dn}(n) = 0$ .
- $n$  の評価値が不明ならば,  $\mathbf{pn}(n) = \mathbf{dn}(n) = 1$ .

(2)  $n$  が内部 OR ノードの場合

- $\mathbf{pn}(n) = \min_{1 \leq i \leq K} \mathbf{pn}(n_i)$ .
- $\mathbf{dn}(n) = \max_{1 \leq i \leq K} \mathbf{dn}(n_i) + (k - 1)$ .

(3)  $n$  が内部 AND ノードの場合

- $\mathbf{pn}(n) = \max_{1 \leq i \leq K} \mathbf{pn}(n_i) + (k - 1)$ .
- $\mathbf{dn}(n) = \min_{1 \leq i \leq K} \mathbf{dn}(n_i)$ .

証明数と弱証明数では内部ノードの計算に違いがある。証明数では子ノードの証明数の和であるのに対し、弱証明数では子ノードの弱証明数の最大値に不明な子ノードの個数  $-1$  を加えている。このことから弱証明数は、局面の多くが合流することを仮定しており、オセロ等の DAG で表現される問題の探索に有効である。

図 2.6 にルートノード  $root$  を評価する弱証明数探索のアルゴリズムを示す。ここで、ノード  $n$  が、OR ノードのときの証明数・AND ノードのときの反証数を  $n.\phi$  とし、OR ノードの反証数・AND ノードの証明数を  $n.\delta$  とする。そして多重反復進化の際の  $n.\phi$ ,  $n.\delta$  の閾値を  $n.th\phi$ ,  $n.th\delta$  とする。また  $\Delta\text{Min}(n)$  はノード  $n$  の子ノードの  $\delta$  のなかで最小の  $\delta$  を返し、 $\Phi\text{Max}(n)$  は (最大の  $\phi$  + 未解決の子ノード数 - 1) を返す関数である。

### 2.3.3 Proof tree tracing

ノード  $n$  が証明されたときには、 $n$  の証明木が存在する。**proof tree tracing**[19] は、 $n$  の証明木をなぞることで、他のノードの証明が可能であるかを高速に調べる手法である。この手法を、証明数探索中に用いることで、探索の性能を更に上げることができる。

```

Function df-wpn(node root){
    /* ルートノードの閾値は  $\infty$  */
    root.th $\phi$  :=  $\infty$ ;  root.th $\delta$  :=  $\infty$ ;
    multiID(root);
    if root. $\delta$  =  $\infty$  then return ルートノードの勝ち;
    else return ルートノードの負け;
}

Function multiID(node n) {
    if n.th $\phi$   $\leq \phi$   ||  n.th $\delta$   $\leq \delta$  then return;
    /* 終端ノード */
    if n が終端ノード then
        if (n が OR ノード && n の評価値が真) ||
            (n が AND ノード && n の評価値が偽) then
                n. $\phi$  = 0; n. $\delta$  =  $\infty$ ;
            else
                n. $\phi$  =  $\infty$ ; n. $\delta$  = 0;
            end
        end
    /* 反復進化 */
    while 1 do
        n. $\phi$  :=  $\Delta$ Min(n);  n. $\delta$  :=  $\Phi$ Max(n);
        if n.th $\phi$  > n. $\phi$  && n.th $\delta$  > n. $\delta$  then return;
         $\phi$  が最小の子ノード child を選択;
         $\delta_2$  := (2 番目に小さい  $\phi$  を持つ子ノードの  $\delta$ );
        child.th $\phi$  := n.th $\delta$  + child. $\phi$  -  $\Delta$ Min(n);
        child.th $\delta$  := min(n.th $\phi$ ,  $\delta_2 + 1$ );
        multiID(child);
    end
}

```

図 2.6: 弱証明数探索のアルゴリズム

ノード  $n_{orig}$  を既に証明済みのノードとし,  $n_{sim}$  を  $n_{orig}$  に類似した証明すべきノードとしたとき, **proof tree tracing** は再帰的に定義できる. 図 2.7 にアルゴリズムを示す.  $n_{sim}$  が証明できた場合は真 (**true**), 証明できなかった場合には不明 (**unknown**) を返す.

この手法は, 詰将棋に置ける無駄な合駒の対策として考えられたが, 一般化三並べにおいても, 同様の手順で証明できる局面が多数存在するため **proof tree tracing** は有効な手法となる.

```

Function proof tree tracing(node  $n_{orig}$ , node  $n_{sim}$ ){
  /* 終端ノード */
  if  $n_{sim}$ が終端ノード then
    if  $n_{sim}$ の評価値が真 then return true;
    else return unknown;
  end
  /* 証明木をたどる */
  if  $n_{sim}$ が内部 OR ノード then
    /* OR ノードの場合は、必勝手をたどる */
    if (必勝となる手 $m$ が,  $n_{orig}$ に存在しない) || ( $m$ は $n_{sim}$ で非合法) then
      return unknown;
    else
       $n_{orig} := (m \text{ により生成される } n_{orig} \text{ の子ノード});$ 
       $n_{sim} := (m \text{ により生成される } n_{sim} \text{ の子ノード});$ 
      return proof tree tracing( $n_{orig}$ ,  $n_{sim}$ );
    end
  else
    /* AND ノードの場合は、全ての手をたどる */
    foreach  $n_{sim}$ の合法手  $m$  do
      if  $m$ は $n_{orig}$ で非合法手 then return unknown;
       $n_{orig} := (m \text{ により生成される } n_{orig} \text{ の子ノード});$ 
       $n_{sim} := (m \text{ により生成される } n_{sim} \text{ の子ノード});$ 
      if proof tree tracing( $n_{orig}$ ,  $n_{sim}$ ) = unknown then return unknown;
    end
    return true;
  end
}

```

図 2.7: proof tree tracing のアルゴリズム

## 第3章

# 計算機による勝ち型判定

### 3.1 証明数探索の利用

過去の研究では、一般化三並べの勝敗判定に計算機を用いずに証明してきたものが多かった。その理由として、計算機を利用する際に、一般化三並べのルールでは盤の大きさが無限大となっていることに起因する2つの問題が生じるからである。

- (1) 計算機で無限大の盤面を表現することができない。
- (2) 無限に決着がつかない場合の終端ノードの評価ができないため、負け型を証明することができない。

以上2点の理由から一般化三並べは全解探索を行うことができず、完全である探索を行うことができない。

しかしながら、ゲーム **AND** のような組み合わせる動物の数が多い場合には、人の手による解析が難しくなる。そこで本章では、2.3章で紹介した証明数探索を適用し、計算機を利用して一般化三並べとその拡張ゲームの勝敗判定を行うことを考える。

直感的には、勝ち型の判定だけならば盤を有限にするだけで解決できると考えられるかもしれないが、盤の大きさを有限にすることで解決といえるためには、「 $s \times s$  の盤で勝ち型であるならば、 $(s+1) \times (s+1)$  の盤でも勝ち型である。」ことが成り立つ必要がある。しかし、これは真偽が不明である。なぜならば、盤の大きさを有限にするということは、先手のみならず後手の無限に存在する合法手も制限してしまうことになるからである。

勝ち型判定の場合、先手は **OR** ノードとなり、後手が **AND** ノードとなる。よって先手の合法手の一部が制限されたとしても、残りの合法手の中に必勝である手が存在すれば、完全性は無いが勝ち型であることがいえる。しかし、制限された合法手に先手の必勝を防ぐ手が存在する可能性があり、後手ではすべての合法手において先手必勝であることを示す必要があるからである。

簡単に説明すると、 $s \times s$  の盤で先手の必勝手順が存在したとしても、 $(s+1) \times (s+1)$  の盤で同様の手順を行った場合に、後手のリーチになっている局面が存在する可能性がある。もし、そんな局面が存在して後手必勝であるならば、先祖の局面の評価値が偽に逆転し、最終的な判定が負け型になる可能性を含んでいるということである。

そこで次節では、完全性は無いが勝ち型を判定する方法と、その効率化を考える。併せて、畳敷き戦略では負け型と証明できないことを判定する方法を示す。

## 3.2 勝ち型の判定

先に述べたように、盤を有限にするだけでは勝ち型判定をすることができないため、さらに条件を付加することで判定を可能にする。

盤を有限にすることで生じる問題は、後手の手を制限してしまうことにあった。そこで、以下の条件を加える。

- (1) 後手のみ有限の盤の外側に置けることとする。
- (2) 盤外に置いた位置は無視して置いた個数だけを数える。
- (3) 盤外に置かれた石は任意の置き方を考えることができる。

盤外に置いた石の個数だけを数えることで、無限に存在する後手の盤外に石を置く手をまとめて考えることができる。すなわち計算機内部で保持する盤の大きさは、先手黒石を置ける範囲となる。なおかつ、盤外の石の任意の置き方を考えることで、すべての置き方を考慮していることになり、勝ち型の判定が可能となる。

ここで条件を付加した場合の、先手番における終端ノードの判定手順を図 3.1 に示す。手順中、盤外の白石の置き方は無限個存在するが、実装上は盤外の白石の個数が、



作ればよい動物の細胞数と動物の一部分である盤内の白石の個数との差より大きいことを調べればよい。

ただし、この勝ち型判定では後手が盤外に作る動物を先手は防ぐことができず、また後手は盤外の石を自由に置き換えることができるため、後手が有利になりすぎるという問題が残っており、より対等な条件での判定方法の考案が望まれる。

```

if 後手の直前手が盤外に白石を置く手 then
    if 盤外の白石の個数 = 作ればよい動物の細胞数 then
        終端ノード（後手の勝ち）；
    end
else
    if 盤内で作ればよい動物が完成した then
        終端ノード（後手の勝ち）；
    end
end
foreach 盤外の白石の置き方 do
    if 盤内の白石と併せて作ればよい動物が完成した then
        終端ノード（後手の勝ち）；
    end
end
if 盤に空きマスが存在する then
    内部ノード（探索を続ける）；
else
    終端ノード（後手の勝ち）；
end

```

図 3.1: 先手番における終端ノード判定の手順

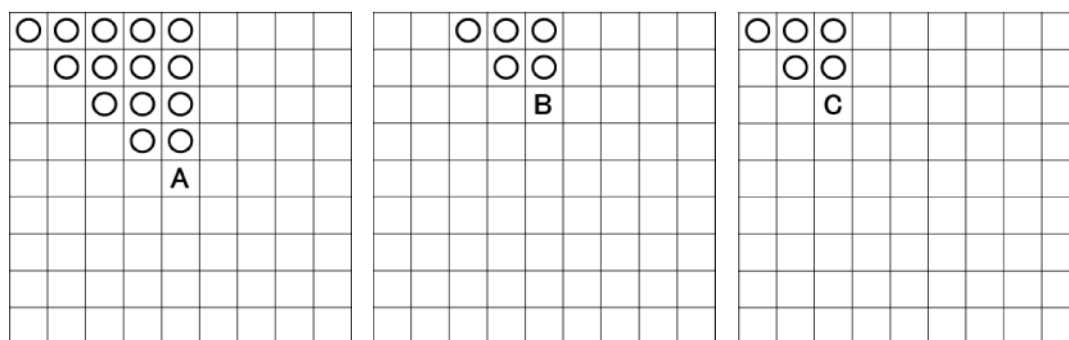
### 3.3 探索の効率化

探索を行う際に定めなければならない変数には、盤の大きさ（先手黒石を置ける範囲）、初手の位置がある。盤の大きさは大きい方が勝ち型と判定できる可能性が高くなるかわりに、探索コストが大きくなると考えられる。また初手の位置は、中心に置くよりも少しずらした方が効率良く盤面を利用できるかもしれない。

さらには、探索コストを減らすために先手の無駄でありそうな手を排除する必要がある。一般化三並べでは、既に置かれた石の近傍に置いていく手が有効であると考えられることから、先手の手を黒石の近傍で制限することが有効であると考えられる。

本節では、盤の大きさや初手の位置、後手の手の制限による探索コストの変化を調べ、勝ち型判定探索の効率化を考える。

まず、先手の初手をどこに置けば盤面を効率良く使うことができるかを調べる。初手の位置は、盤の中心・辺・角の3種類を調べる。盤の大きさを  $n \times n$  とし、盤の中心を原点  $(0,0)$  として各マス目に直交座標を与えたとき、盤の辺とは  $(0, n/4)$  のマス、盤の角とは  $(n/4, n/4)$  のマスを位置する。



(a) 盤の中心

(b) 盤の辺

(c) 盤の角

図 3.2: 初手の位置に対する後手の盤内の候補手

また  $9 \times 9$  の盤面での初手に対する後手の1手目の内、盤内に置く手を図 3.2 に示す。図 3.2 に示した位置と盤外の手について調べれば、後手の手を全て網羅している理由を簡単に述べる。初手の位置を原点  $(0,0)$  として各マス目から直交座標を与え



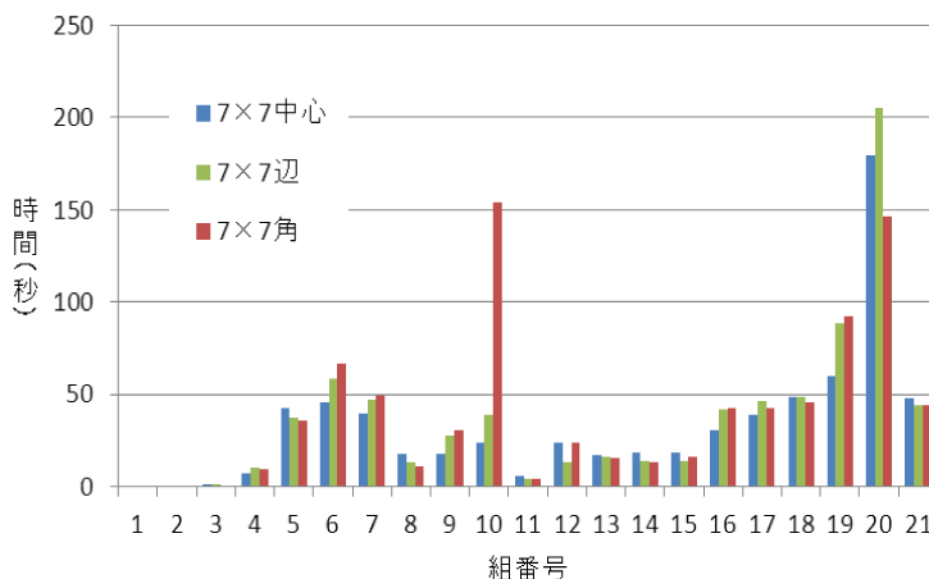


図 3.3: 初手の位置による探索時間の変化

表 3.1 は、この条件での勝ち型判定の結果を示したものである。○が勝ち型と判定できた組であり、×が勝ち型と判定できなかった組である。また、図 3.3 は中心に置いた場合に勝ち型と判定できた 21 組についての探索にかかった時間である。なお、組番号は表 3.1 の左上の組から 1 から順に付けた。

表 3.1 の結果から、先手の初手は盤の中心よりも、中心からずらして置く方が 2 組多く勝ち型と判定でき盤を効率良く使えていることが分かった。しかし、図 3.3 を見ると勝ち型判定にかかる時間は、中心に置いた方が少なくなる。これは初手を盤の中心に置くことで盤の回転や反転といった対称性を、他の位置に置いたときに比べて利用しやすく、探索を削減することができるためである。また、初手が辺の場合と角の場合を比べた場合は、平均すると角よりも辺に置いた方が少ないことが分かった。

この実験結果から以下では、中心に置いたときよりは時間がかかるが、より盤面を活用できる盤の辺  $(0, n/4)$  に先手の初手を置くこととする。

次に盤の大きさと手の制限についてであるが、一般に盤の大きさを小さくすると探索コストを減らすことができる。しかし、逆に小さくしすぎると勝ち型と証明できない。また、勝ち型と証明できない場合には先手の手を全て調べるため、勝ち型と判定

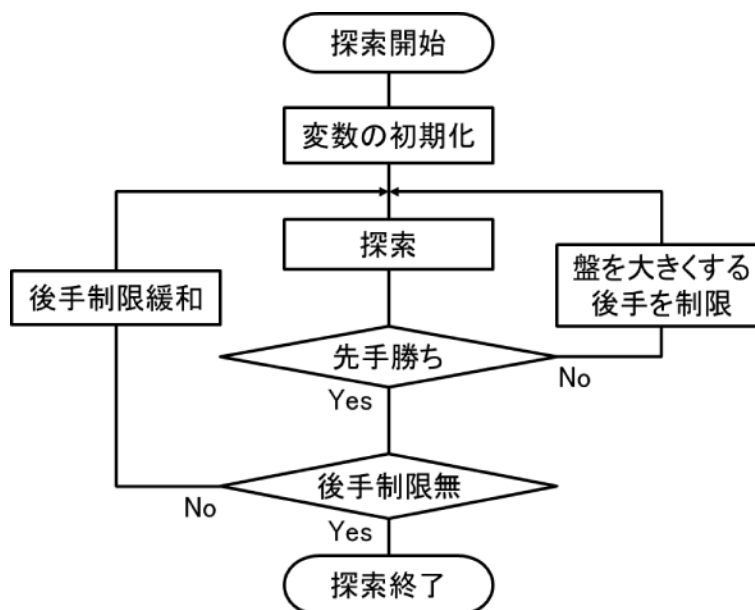


図 3.4: 勝ち型判定探索の手順

できる場合に比べて探索に大きな時間がかかってしまう．ここで，勝ち型判定では先手の手は制限可能である．逆に，勝ち型と判定できない場合は，探索における先手が **AND** ノード，後手が **OR** ノードとなるため後手の手を制限することができる．一般化三並べの場合，置ける範囲が決まっているため黒石から離れた位置に石を置く手は有効でないことから黒石の近傍を利用して手を制限することができる．しかし，探索前に探索コストが最小になるように盤の大きさや手の制限を定めることは難しい．

そこで，盤の大きさは探索の深さを表し，手の制限は分岐数を表していることに着目し，反復進化法のように反復的に探索時の変数を大きくしていくことを考える．まず盤を小さくし，後手の手も制限して探索を行う．勝てない場合は，後手の制限が無い場合でも勝てないため，盤を大きくして再度探索を行う．勝った場合は，後手の制限を緩めて再度探索を行う．もし後手の制限が無い場合で勝ちであるならば勝ち型であるといえる．

図 3.4 に勝ち型判定探索の手順を示す．

この探索をゲーム **OR** の勝ち型と未解明の 40 組（表 3.2）に対して行った際に要した実行時間を図 3.5(a) と図 3.5(b) に示す．比較対象として，盤の大きさを  $9 \times 9$  に固

























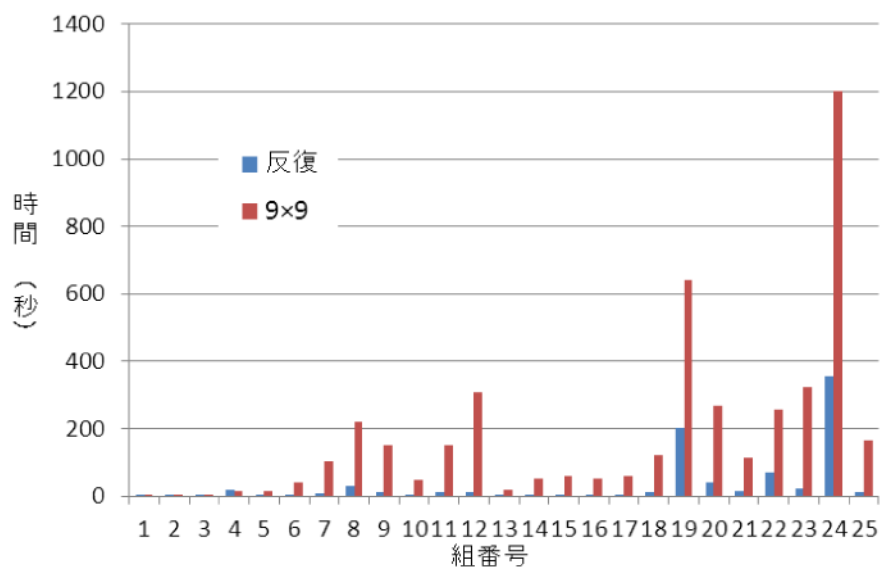
													
			1	2				3			26	4	5
								6	7			8	9
								10		27		11	12
								13	14	15		16	17
													28
									29			30	18
									31			19	20
									21	22	23	24	25
										32	33	34	35
											36	37	38
													39
													40

表 3.2: 組番号

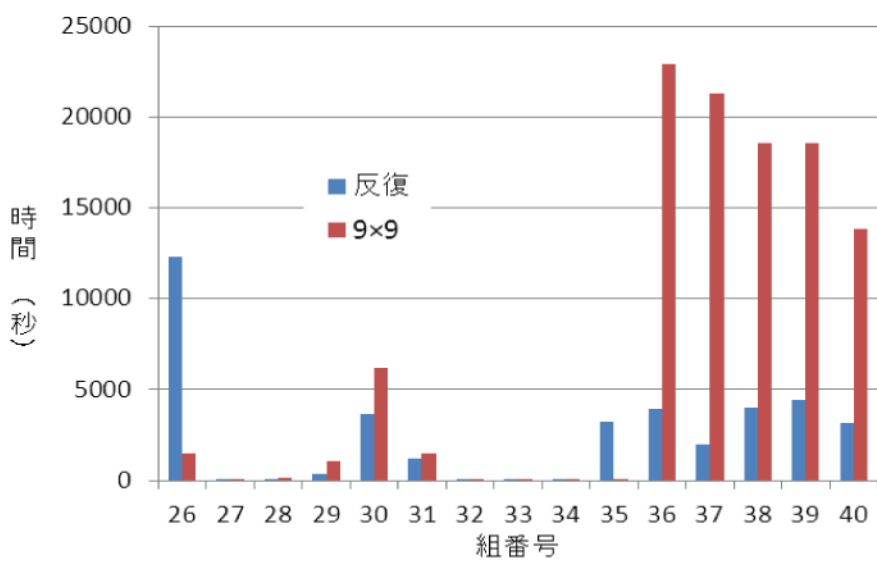
定し、後手の制限を付けない探索の実行時間も示してある. なお、先手の制限は黒石の4近傍とし、後手の制限は黒石の4近傍, 8近傍, 無制限の3段階を設けた.

図 3.5(a) は勝ち型と判定できた組 (表 3.2 1 番~25 番) であり, 図 3.5(b) はこの条件では勝ち型とは判定できなかった組 (表 3.2 26 番~40 番) についての結果である. 図 3.5(a) より, 勝ち型証明できた場合では, 予め十分大きな盤で探索を行うよりも, できるだけ小さな盤で勝ち型の証明を試みる方が探索時間が減少することが確認できた. さらに図 3.5(b) を見ると, 勝ち型証明ができなかった場合については, 予め後手を制限して探索をすることで, 26 番と 35 番の組を除き大きく探索時間が削減できている. また図 3.5(a) と図 3.5(b) を比較すると, 勝ち型と判定できた場合に比べ, できない場合の探索時間が非常に大きいことが分かる. 以上の結果から反復的に盤の大きさを拡大させ, 黒石の近傍での手の制限を緩和させていく探索が有効であることが確認できた.

盤の大きさを  $9 \times 9$  に固定した場合より判定に時間がかかった 26 番と 35 番について考察すると, 26 番の組は勝ち型であるが  $9 \times 9$  以下の盤での必勝手順が知られていない組である. この組は番の大きさが  $9 \times 9$  で後手が黒石の4近傍にのみ置けるとした場合では, 先手に必勝手順が存在するが, 後手が黒石の8近傍にも置けると防がれて



(a) 勝ち型と判定できた組



(b) 勝ち型と判定できなかった組

図 3.5: 勝ち型判定の実行時間

しまう。このような問題は勝ち負けが際どく証明木が大きくなる。従って探索時間が反復により蓄積し、盤の大きさを  $9 \times 9$  に固定した時よりも時間がかかったといえる。35 番の組は未解明の組であり、後手を制限した場合でも必勝手順が見つからなかったが、25 番と同様に後手に制限があると勝ち負けが際どくなることによる結果であると推察され、このような問題に対しては対策が必要になることが分かった。

### 3.4 「どんな畳敷き戦略でも負け型と証明できない」ことの判定

計算機では、負け型の判定はできないが、AND/OR 木の探索を用いてどんな畳敷き戦略でも負け型と証明できないことが示せる。例えば、未解明の Snaky は畳敷き戦略では防ぐことができない。この証明は、次の手順で行うことができる [32]。

- (1) 初期状態として、有限の盤面には畳は存在しないものとする。
- (2) 既に敷かれた畳（ペアとなっているマスの両方）を含まないように、目標とする動物を盤面に置く。
- (3) 置かれた動物を防ぐように、畳を重複がないように 1 枚敷く。
- (4) 手順 2,3 を繰り返していき、目標となる動物を防げ無くなった状態を終端状態として「勝ち」とする。もしくは畳を含まずに動物を置くことができない場合は「負け」とする。

この手順において、初期状態はルートノード、手順 2 の動物を置く状態は OR ノード、畳を敷く状態は AND ノードとして考えることができる。従って、この判定にも AND/OR 探索を利用することができる。AND/OR 木探索を行った結果が「勝ち」であるならば、その動物は畳敷きでは防ぐことができない。「負け」である場合には、盤面を広くして再度探索を行えば良い。

この手法はゲーム OR のように、作りたい動物が複数ある場合にも利用できる。この場合は、畳を含まないように動物を盤面に置く手順の際、複数のある動物のうち、どれか 1 体が畳を含まなければ良い。



## 第4章

# ゲーム OR, ゲーム AND, ゲーム $n$ 細胞 OR における進展

本章では, 3章の勝ち型判定を用いて, ゲーム OR, ゲーム AND の未解明である組が勝ち型となり得るかを調べた.

ゲーム  $n$  細胞 OR については, 先手が黒石の4近傍に限定された場合について, 先手が作れる動物の細胞数の上限を18から16に下がったことを示す.


























### 4.1 ゲーム OR の勝ち型判定

ゲーム OR では, 勝ち型と既に確認されている31組と, 未解明であり畳敷きでは負け型証明ができない9組に対し勝ち型判定を行った. この結果を表4.1に示す. 先手の制限は黒石のある8近傍まで置けるとした. 探索時の反復では盤の大きさは $5 \times 5$ から $9 \times 9$ まで縦横1マスずつ大きくしていき, 後手の制限は黒石の4近傍, 8近傍, 無制限の3段階を設定した.

表中の○で表される組は, 既に勝ち型と分かっていたが計算機では判定できなかった組であり, ◎で表される組は, 計算機でも勝ち型と判定ができた組である.

結果は残念ながら新たに勝ち型を見つけることができなかった. また, 勝ち型と分かっている3組についても計算機では勝ち型と判定できなかった. 勝ち型と判定できなかった組は,  $9 \times 9$ の盤面での必勝手順が知られていない組である. 今後はさらに大きな盤面で探索することや, 二つ組の必勝パターンを用いる等の工夫が必要である.

表 4.1: ゲーム OR における二つ組の勝ち型判定

													
		×	◎	◎	×	×	×	◎	×	×	○	◎	◎
			×	×	×	×	×	◎	◎	×	×	◎	◎
				×	×	×	×	◎	×	◎	×	◎	◎
					×	×	×	◎	◎	◎	×	◎	◎
						×	×	×	×	×	×	×	△
							×	×	◎	×	×	◎	◎
								×	△	×	×	◎	◎
									◎	◎	◎	◎	◎
										△	△	△	△
											○	○	△
												×	△
													△

○ 勝ち型 (既知)

◎ 勝ち型 (計算機)

× 負け型

△ 未解明

表 4.2: ゲーム AND における二つ組の勝ち型判定

	○	○	○	○	○	○	○	○	○	○	○
		○	○	○	○	○	○	○	○	◎	◎
			○	○	◎	◎	◎	◎	◎	◎	◎
				○	◎	○	◎	◎	◎	◎	◎
					×	◎	△	△	△	△	△
						△	△	△	△	△	△
							△	△	△	△	△
								△	△	△	△
									△	△	△
										×	△
											×

○ 勝ち型 (既知)

◎ 勝ち型 (計算機)

× 負け型

△ 未解明

## 4.2 ゲーム AND の勝ち型判定

ゲーム AND では、表 2.2 の△で示した未解明である 40 組に対し勝ち型判定を行った。この結果を表 4.2 に示す。ゲーム OR 同様、探索時の反復では盤の大きさは  $5 \times 5$  から  $9 \times 9$  まで縦横 1 マスずつ大きくしていき、後手の制限は黒石の 4 近傍、8 近傍、無制限の 3 段階を設定した。表中の○で表される組は、既に勝ち型と分かっていた組であり、◎で表される組は、計算機により新たに勝ち型と判定ができた組である。

計算機を利用することにより、新たに 16 組勝ち型であることが分かった。この結果から、組み合わせる動物の個数が多くなる場合に計算機が有用であることが確認できた。しかし、1 マス共有させた細胞数が 7 細胞以上になるとゲーム OR 同様解けない組合せが多くなった。

### 4.3 ゲーム $n$ 細胞 $\mathbf{OR}$ の先手 4 近傍限定時における上限

既存研究 [32] では、ゲーム  $n$  細胞  $\mathbf{OR}$  の先手 4 近傍限定時における上限は 18 個であった。本研究では、この上限を下げることを考える。

**定理 1** 先手が黒石を置ける位置を既に置かれた先手黒石の 4 近傍に限定したとき、後手は先手が初手に置いた石に対して、図 4.1(a) のように先手黒石を 17 個以上連結させないで囲い込む戦略が存在する。

**証明 1** 先手黒石の初手（図 4.1(b) 黒丸）に対して、後手初手は、図 4.1(b) に示す白丸に置く。後手 2 手目以降は、図 4.1(b) に示す応手で黒石を囲い込む。畳敷きのように、先手の領域内の手に対して線で結ばれたマスへ後手が白石を置いていく。

先手の手に対し 2 本の線が延びているマスもあるが、石を置いていく過程で必ず 1 対 1 に対応するか、既に白石が置かれている場合となる。既に置かれている場合は領域内のどこに置いても良い。

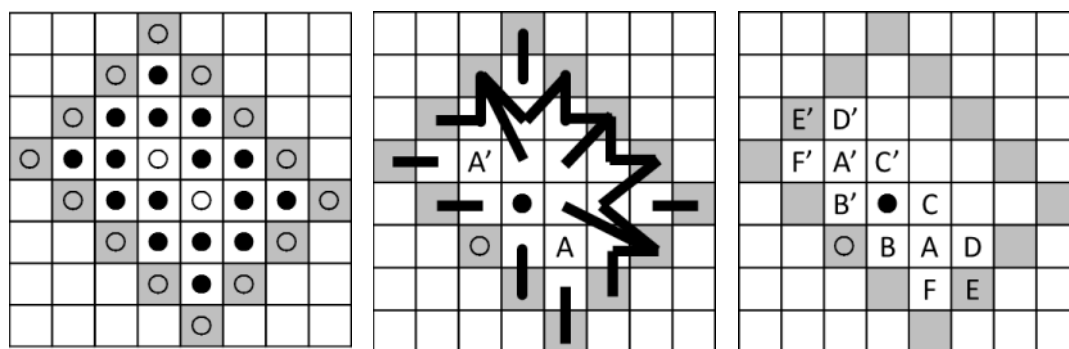
マス A ( $A'$ ) に先手が石を置いた場合、図 4.1(c) において

- (1) マス B ( $B'$ ) に先手黒石を置かれている。
- (2) マス C ( $C'$ ) に先手黒石を置かれている。

の少なくともどちらか一方を満たしている必要がある。(1) のみ、または (1) と (1) 両方を満たしている場合は、後手白石はマス E ( $E'$ ) に置く。(2) のみ満たしている場合は、後手白石はマス F ( $F'$ ) に置けばよい。□

この畳敷き戦略にも似ている囲い込み戦略により、先手黒石を 17 個以上連結させずに囲い込むことができた。

以上により、先手が黒石を置ける位置を既に置かれた先手黒石の 4 近傍に限定したゲーム  $n$  細胞  $\mathbf{OR}$  において、先手は 17 細胞以上の動物は作ることができず、負け型となる細胞数の下界に 17 が含まれることが判明した。その上限を 2 小さい 16 個に下げることができた。これにより、勝ち型となる細胞数の最大値は 12 から 16 の間であることが分かったが、下界は先手が最悪の手を続けた場合の結果であることから下界を上げることが必要になると考えられる。



(a) 後手が囲い込む領域

(b) 後手の戦略

(c) 戦略の分岐

図 4.1: ゲーム  $n$  細胞 OR における後手の手順

## 第5章

### ゲーム NOTAND

本章では、一般化三並べの新たな拡張として、**禁止動物**の導入を考える。これは連珠における禁手に似たルールである。連珠の場合は先手のみに禁手があるが、ここでは、先手後手ともに同じ禁止動物を定めることとする。これにより、禁止動物を導入した新たなゲーム NOTAND を提案する。

#### 5.1 ゲーム NOTAND の提案

禁止動物を導入した**ゲーム NOTAND**を以下のように定義する。

**定義 5 (ゲーム NOTAND)** 盤の大きさが無限大の碁盤状の盤面に二者が交互に石を一つずつ置き、予め定められた連結した石で定義されるある動物を、別に定められた禁止動物を作らずに（同時に作る場合も禁止する）、斜めは許さずに回転と反転を許して先に作った方が勝ちというゲームである。

#### 5.2 ゲーム NOTAND の考察

禁止動物を導入した場合、これまでのゲームにおける「石を置くことが自らの不利にはならない」という前提が無くなり、後手必勝がありえないということが自明でなくなる。また、単独で負け型である動物の組に禁止動物を導入することで後手が先手を防ぐ際に不利になることも考えられ、勝ち型になる可能性もあることが、このゲームの興味深い点となっている。

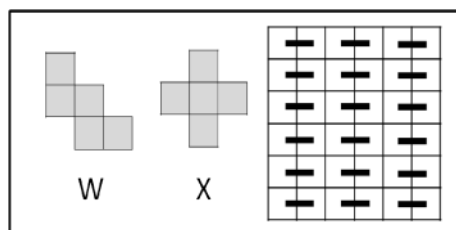


図 5.1: W と X に共通な畳敷き

従って、このゲームにおける動物の組の分類としては、「先手が必ず勝つことのできる組」、「先手が必ず負ける組」、「無限に続いて引き分けとなる組」の3つが考えられる。しかしながら、本論文では先手が必ず負けとなる組を見つけられていないことから、負け型の定義は、変えずに「無限に続いて引き分けとなる組」とする。

### 5.2.1 勝ち型の証明

禁止動物が完成する前に、必然的に作ればよい動物が完成していれば、勝ち型となる。これは、禁止動物が完成する1手前の形を考え（1細胞を削除した形）を全て列挙し、その全ての形に対して作ればよい動物が含まれていることを確認することで勝ち型と分類できる。

### 5.2.2 負け型の証明

禁止動物が作ればよい動物に含まれる場合は当然負け型となる。その他には、作ればよい動物と禁止動物が共通な畳敷きで防ぐことができれば、後手は禁止動物を作らずに先手を防ぐことが可能であるため負け型である。当然その二つの動物の作ればよい動物と禁止動物を入れ替えた場合についても負け型である。

単独で負け型動物が含まれるため次節で示す結果には含まれないが、例えばWとXは図5.1に示す通り、共通の畳敷きで防げるため負け型として証明できる。

### 5.3 ゲーム NOTAND の結果

ゲーム AND における勝敗判定の結果を表 5.1 に与える. 表 5.1(a) は, 作ればよい動物と禁止動物が共に単独勝ち型の組, 表 5.1(b) は, 作ればよい動物が単独勝ち型, 禁止動物が最小の単独負け型あるいは Snaky の組, 表 5.1(c) は, 作ればよい動物が最小の単独負け型あるいは Snaky, 禁止動物が単独勝ち型の組の勝敗判定の結果である. 勝ち型の組に書かれている数字は探索時に勝ち型と証明した局面の最長手数である. ただし証明数探索には最適性が保証されていないため, この数字は最適な必勝手順の手数ではなく, その上限である. 単独勝ち型動物どうしの 121 組の勝敗判定の結果, 57 組が勝ち型, 49 組が負け型となった. 禁止動物を単独負け型あるいは Snaky とにした場合は, 104 組が勝ち型, 39 組が未解明となり, 負け型は見つからなかった. また, 単独負け型あるいは Snaky を作ればよい動物, 単独勝ち型を禁止動物とした場合は, 74 組が負け型, 69 組が未解明となり, 勝ち型は見つからなかった. また勝ち型の多くは 10 手未満で勝つことができるが, Z と X の組では最長で 23 手での勝ち手順を見つけていた. 今後は未解明の解析と共に, 動物の種類数や組にする動物数を増やして勝敗判定をする必要がある.



表 5.1: ゲーム NOTAND における二つ組の勝敗判定

		禁止動物											
作ればよい動物		□	▢	▣	▤	▥	▦	▧	▨	▩	▪	▫	▬
	□	x	1	1	1	1	1	1	1	1	1	1	1
	▢	x	x	3	3	3	3	3	3	3	3	3	3
	▣	x	x	x	5	9	5	5	7	9	9	9	9
	▤	x	x	5	x	5	5	5	5	5	5	5	5
	▥	x	x	x	△	x	△	△	△	△	△	△	△
	▦	x	x	x	x	9	x	7	7	9	9	9	9
	▧	x	x	x	x	9	7	x	11	9	9	11	11
	▨	x	x	7	x	9	7	7	x	9	9	9	9
	▩	x	x	x	x	x	x	△	△	x	△	△	△
	▪	x	x	x	x	x	x	△	△	△	x	△	△
	▫	x	x	x	x	17	x	△	x	17	17	x	x

		禁止動物													
作ればよい動物		▬	▬	▬	▬	▬	▬	▬	▬	▬	▬	▬	▬	▬	▬
	▬	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	▬	3	3	3	3	3	3	3	3	3	3	3	3	3	3
	▬	5	5	5	5	5	5	5	5	5	9	5	5	5	5
	▬	5	5	5	5	5	5	5	5	5	5	5	5	5	5
	▬	△	△	△	△	△	△	△	△	△	△	△	△	△	△
	▬	9	9	7	7	7	7	9	9	9	9	9	7	7	7
	▬	15	7	11	7	15	7	7	7	9	7	7	7	7	7
	▬	7	7	9	7	7	7	7	7	9	7	7	7	7	7
	▬	△	△	△	△	△	△	△	△	△	△	△	△	△	△
	▬	△	△	△	△	△	△	△	△	△	△	△	△	△	△
	▬	17	19	21	15	23	17	17	19	17	17	17	17	17	17

(a) NOT(単独勝ち型) AND (単独勝ち型)

(b) NOT(単独負け型) AND (単独勝ち型)

		禁止動物											
作ればよい動物		□	▢	▣	▤	▥	▦	▧	▨	▩	▪	▫	▬
	▬	x	x	△	x	△	△	△	△	△	△	△	△
	▬	x	x	x	x	△	x	△	△	△	△	△	△
	▬	x	x	△	x	△	△	△	x	△	△	△	△
	▬	x	x	x	x	△	x	x	x	△	△	△	△
	▬	x	x	x	x	△	△	x	△	△	△	△	△
	▬	x	x	x	x	△	x	x	△	△	△	△	△
	▬	x	x	x	x	△	x	△	△	△	△	△	△
	▬	x	x	x	x	△	x	△	△	△	△	△	△
	▬	x	x	x	x	△	x	△	△	△	x	△	△
	▬	x	x	x	x	△	x	△	x	△	△	x	x
	▬	x	x	x	x	x	x	△	x	x	△	x	x

手数 勝ち型

x 負け型

△ 未解明

(c) NOT(単独勝ち型) AND (単独負け型)

## 第6章

### まとめと今後の課題

本研究では、証明数探索を一般化三並べの勝ち型判定に適用し、一般化三並べの拡張であるゲーム OR, ゲーム NOTAND での勝ち型判定を行った。また、新たにゲーム NOTAND を提案し、考察を行った。

#### 6.1 証明数探索の利用

証明数探索を一般化三並べに用いる際に、盤の大きさが無限というルールから生じる問題を、計算機が持つ盤の大きさを先手の黒石を置ける範囲とし、盤外に置いた白石の個数を数えることで解決した。また、盤の大きさや手の制限を反復的に変化させることで、勝ち型と判定できない場合の探索コストを削減させた。

しかし、本論文の手法だと後手が有利になりすぎているため、より対等な条件での判定を可能にすることが必要である。盤外に置いた白石は黒石で防がれることがないことより、任意の置き方を考える必要はなくある有限の範囲の置き方を考えれば十分であるとは考えられる。もしこの証明がつけば、現在の手法よりも対等な条件での判定が可能となるため新たな勝ち型が見つかることが予想される。

#### 6.2 ゲーム OR・ゲーム AND

ゲーム OR では、新たに勝ち型を見つけることができなかったが、ゲーム AND では、新たに 16 組の勝ち型を見つけることができた。今後の課題は、組にする動物の必勝パターンを用いるなど、その動物に特化させて探索を行うことで新たな勝ち型を見つけ

ることである．その結果により，勝ち型が存在する細胞数の上界が上がることとなる．

### 6.3 ゲーム $n$ 細胞 OR

先手が黒石を置ける位置を既に置かれた先手黒石の4近傍に限定した場合には，後手は先手の石を囲い込む戦略により，17細胞以上の動物は作ることができないという結果を得ることができた．しかしながら，先手の手が8近傍限定や無制限の場合の証明法については未だ不明であるため，今後は先手の手が8近傍限定や無制限になったときの戦略を考え，勝ち型となる  $n$  の最大値の下界や上界を調べる必要がある．

### 6.4 ゲーム NOTAND

禁止動物により自分の置いた石が自らに不利になる可能性があるということで，更に難解なゲームとなった．実際に，後手が勝ちとなる組や，負け型が禁止動物と組になることで勝ち型になることを確認することができれば非常に興味深い結果となるだろう．あるいは，そのような組が存在するかどうかを解析していくことが今後の課題となる．

### 6.5 一般化三並べ全般

本論文では，一般化三並べの拡張に対してまだまだ部分的な考察であったため，更に全体像を解明していく必要がある．そのためには，組み合わせる動物の細胞数や種類数を増やしていき，より網羅的な勝敗判定と考察が求められる．また将来的には，任意の論理式で表される動物の組に対しての勝敗判定が可能になると面白いだろう．そのためにも，畳敷き以外での負け型の判定法を始め，ゲームの特徴を考慮した新たな勝敗判定の手法の発見が求められる．

## 参考文献

- [1] God's Number is 20. <http://www.cube20.org/>.
- [2] 情報処理学会－コンピュータ将棋プロジェクト.  
<http://www.ipsj.or.jp/50anv/shogi/index2.html>.
- [3] L. V. Allis, M. Meulen, and H. J. Herik. Proof-number search. *Artificial Intelligence*, Vol. 66, No. 1, pp. 91–124, 1994.
- [4] L. V. Allis, H. J. van den Herik, and M. P. H. Huntjens. *Go-Moku and Threat-space Search*. University of Limburg, Department of Computer Science, 1993.
- [5] L.V. Allis. *Searching for solutions in games and artificial intelligence*. Ponsen & Looijen, 1994.
- [6] A. Brünger, A. Marzetta, K. Fukuda, and J. Nievergelt. The parallel search bench ZRAM and its applications. *Annals of Operations Research*, Vol. 90, pp. 45–63, 1999.
- [7] E. Fisher and N. Sieben. Rectangular polyomino set weak (1,2)-achievement games. *Theoretical Computer Science*, Vol. 409, pp. 333–340, 2008.
- [8] G. Fülep and N. Sieben. Polyiamonds and polyhexes with minimum site-perimeter and achievement games. 2008.
- [9] I. Halupczok and J. Schlage-Puchta. Achieving snaky. *Electronic Journal of Combinatorial Number Theory*, Vol. 7, No. 1, 2007.
- [10] F. Harary. Achieving the skinny animal. *Eureka*, Vol. 42, pp. 8–14, 1982.
- [11] F. Harary. Is snaky a winner. *Geombinatorics*, Vol. 2, pp. 79–82, 1993.

- [12] F. Harary and H. Harborth. Achievement and avoidance games with triangular animals. *J. Recreational Mathematics*, Vol. 18, No. 2, pp. 110–115, 1985–1986.
- [13] F. Harary, H. Harborth, and M. Seemann. Handicap achievement for polyominoes. *Congressus Numerantium*, Vol. 145, pp. 65–80, 2000.
- [14] H. Harborth and M. Seemann. Snaky is an edge-to-edge loser. *Geombinatorics*, Vol. 5, No. 4, pp. 132–136, 1996.
- [15] H. Harborth and M. Seemann. Snaky is a paving winner. *Bull. inst. Combin. April*, Vol. 19, pp. 71–78, 1997.
- [16] H. Harborth and M. Seemann. Handicap achievement for squares. *J. Combin. Math. Combin*, Vol. 46, pp. 47–52, 2003.
- [17] K. Inagaki and A. Matsuura. Winning strategies for hexagonal polyomino achievement. *WSEAS*, pp. 29–31, 2007.
- [18] H. Ito and H. Miyagawa. Snaky is a winner with one handicap. *HERCMA*, pp. 25–26, 2007.
- [19] Y. Kawano. Using similar positions to search game trees. *Games of no chance: combinatorial games at MSRI, 1994*, p. 193, 1998.
- [20] A. Pluhar. The accelerated k-in-a-row game. *Theoretical Computer Science*, Vol. 270, pp. 865–875, 2002.
- [21] N. Sieben. Snaky is a 41-dimensional winner. *Integers*, Vol. 4, p. 6, 2004.
- [22] N. Sieben. Wild polyomino weak (1,2)-achievement games. *Geombinatorics*, Vol. 13, No. 4, pp. 180–185, 2004.
- [23] N. Sieben. Polyominoes with minimum site-perimeter and full set achievement games. *European J. Combin*, Vol. 29, No. 1, pp. 108–117, 2008.

- [24] N. Sieben. Proof trees for weak achievement games. *Electronic Journal of Combinatorial Number Theory*, Vol. 8, , 2008.
- [25] N. Sieben and E. Deabay. Polyomino weak achievement games on 3-dimensional rectangular boards. *Discrete Math*, Vol. 290, pp. 61–78, 2005.
- [26] T. Ueda, T. Hashimoto, J. Hashimoto, and H. Iida. Weak proof-number search. In *Proceedings of the 6th international conference on Computers and Games*, pp. 157–168. Springer, 2008.
- [27] 伊藤大雄. パズル・ゲームで楽しむ数学 娯楽数学の世界, 第3章. 森北出版, 2010.
- [28] 伊藤大雄, 宇野裕之 (編) . 離散数学のすすめ. 現代数学社, 2010.
- [29] 伊藤大雄. ハラリーの一般化三並べ. 電子情報通信学会論文誌, Vol. J89-A, No. 6, pp. 458–469, 2006.
- [30] 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, 2003.
- [31] 八鍬友貴. 一般化三並べの勝敗判定に関する研究. Master's thesis, 東北大学 大学院情報科学研究科, 2010.
- [32] 八鍬友貴, 本田耕一, 篠原歩. 一般化三並べの変種 : . 負け型のペアは勝てるのか? 第14回ゲームプログラミングワークショップ, pp. 35–42, 2009.

## 謝辞

学部，大学院の3年間にわたり，このような研究の場を与えて頂くとともに，本研究の直接的な指導教員として多くのご教示，ご鞭撻を賜りました東北大学 大学院情報科学研究科 篠原 歩 教授，ならびに成澤 和志 助教に厚く御礼申し上げます。

また，本論文審査の副審査員を務めて頂きました，東北大学 大学院情報科学研究科 徳山 豪 教授，ならびに東北大学 大学院情報科学研究科 静谷 啓樹 教授には，御専門の立場からの確なご助言や貴重なご意見を賜りましたことを，心から御礼申し上げます。

また，本研究を進める上で多くのご助言，ご指導を頂いた篠原研究室の先輩である八鍬 友貴 氏に，心より感謝申し上げます。また，篠原研究室の皆様には日頃より多くの協力を頂き，本研究以外にもロボカップを始めとして様々なことを学び，経験する場を頂きましたことに深く感謝いたします。

最後に，東北大学 大学院情報科学研究科 博士前期課程までという長い間，進学の機会を与えてくださり，最後まで信じて見守ってくださった家族，私生活においていつも温かく支えてくれた友人一同に，深く感謝いたします。