Vol. 35 No. 12 情報処理学会論文誌 Dec. 1994

時刻入りの節を利用するRLSの学習

田島守彦† 実近憲昭††

本論文は筆者らが開発している学習システム RLS (Recursive Learning System) の新しい結果について述べている．筆者らはオセロゲームを学習の題材としていくつかの結果を発表済みである．これは，RLS 上に記述されたゲームプログラム RLS-Othello がゲームの対局を行い，対局後その打着手の系列から失敗の原因を解析して有用な概念を学習する試みである．補助的な学習手法として RLS は EBG を利用している．しかしこれまでの RLS-Othello の学習では EBG 用の知識の記述が不十分だったため，自分の悪手と将来の敵の良手との間の直接の関係等しか利用できず，従って複雑な盤面パタンのような高度な概念を得ることができなかった．この問題を解決するため，ここでは RLS-Othello の目標概念，領域理論，訓練例のそれぞれに時刻を導入し，棋譜を遡っての推論において，打着時と失敗認識時の間の全時間範囲における任意の時刻間の関係を記述できるようにした．これにより，多数の時間的関係を考慮しなければ学習できないような複雑なパタン（双翼）の概念が学習可能となった．本論文ではまず RLS の概要を紹介し，本論文で利用する学習事例を導入する．次いでこの事例に必要な EBG 用領域理論を設計し，領域理論以外の EBG の構成要素について述べ，これらを利用した対局と学習の例を示す．最後に本システムの考察を行う．

Learning of RLS that Uses Clauses Including Time

Morihiko Tajima† and Noriaki Sanechika††

This paper describes the new result produced by our learning system named RLS (Recursive Learning System). RLS-Othello is the game playing and learning system on RLS. It learns useful concepts by analyzing the cause of failure from a record of a game. RLS also utilizes EBG as an auxiliary means. Because of the shortage of the knowledge for EBG, RLS-Othello has not been able to learn very useful concepts including complex patterns. In this paper, we introduced variables denoting time into goal concept, domain theory, and training example, and enabled RLS-Othello to describe the relations between any two moments. By the improvement of RLS-Othello, RLS-Othello became to be able to learn the concept of "pair wing" which is one of the complex patterns of the game. In this paper, we introduce the outline of RLS firstly. Next, we introduce a learning case which is used in this paper. The designing of the domain theory needed for the case, the other elements of EBG, and an example of the game playing and learning follows it. We give consideration of the system in conclusion.

1. はじめに

筆者らは手続き的知識の学習システム RLS (Recursive Learning System) を開発しており，現在までにいくつかの成果を得ている[1),2)]．そこではオセロゲームを学習の題材としており，RLS上に記述されたゲームプログラム RLS-Othello がゲームの対局と学習を行う．補助的な学習手法として RLS は EBG（説明に基づく一般化）[3)] を利用している．

EBG に関する研究は多くが高速化学習に関するもので，Minton[4)] や沼尾[5)] などのマクロの選択的学習や Minton[6)] などの探索制御ルールの学習がある．これに対して筆者らの研究は，高速化が目的というよりも，事象の因果関係を利用して実行済みの手続きから有用な知識を得ることを目的とする．

ゲーム学習の分野ではチェッカーの局面の評価関数の学習[7)]，五目並べの中間目標の学習[8)]，チェスの盤上の特徴に対応する手の集合の学習[9)] や終局の分類学習[10)] などが研究されてきた．多くが，良悪が既知の多数の局面をデータとしそこから帰納的な推論を行って規則性を見いだすもので，対局後に棋譜に基づき打着手の系列から失敗の原因を解析して有用な概念を学習する試みはほとんどない．筆者らはそのようなゲーム学習に取り組んでいる．

† 電子技術総合研究所知能情報部推論研究室
Machine Inference Section, Machine Understanding Division, Electrotechnical Laboratory
†† 電子技術総合研究所情報アーキテクチャ部分散システム研究室
Distributed Systems Section, Computer Science Division, Electrotechnical Laboratory

文献 1) で実現できている学習には制限があった．すなわち，打着時点の悪手の概念を学習して以後の同様の打着に役立てるのであるが，悪手の概念には時刻（手番号）の要素が入っていないため，自分の悪手と将来の敵の良手との直接の関係しか記述できなかった．従って，得られる概念はマスの種類あるいは単純な部分パタンのような基本的なものにとどまり，複雑な盤面パタンのような高度な概念を得ることができなかった．本論文ではこれを解決するため RLS-Othello の目標概念，領域理論，訓練例のそれぞれに時刻を導入し，棋譜を遡っての推論において，打着時と失敗認識時の間の全時間範囲における任意の時刻間の関係を記述できるようにする．これにより，オセロゲームにおける複雑なパタンについての概念が学習可能となる．

本論文の構成は次のようである．2章で RLS の概要を紹介する．3章で本論文で利用する学習事例を導入する．4章でこの事例に必要な EBG 用領域理論を設計し，5章において領域理論以外の EBG の構成要素について述べる．これらを利用した対局と学習の例を6章で示し，7章で本システムの考察を行う．

2. EBG を利用する RLS

RLS は一般的な手続き的知識の学習を指向するシステムである．RLS を簡単にまとめておく．ある動作のための手続き的知識は型をもつ知識片から合成される．型には基本型，逐次型，条件型，各種構造型（順次試行，最善枝選択，最善枝から順次試行，各枝の総合）がある．知識片にはそれの実行意図が書かれており，実行意図が満たされないとシステムはその手続き実行を失敗と認識する．すなわち知識片のいずれかに何らかの欠陥があると判断して学習モードに入る．

図1に学習モデルを示す．RLS の学習とは，失敗した手続きに対応する学習用のメタ知識（これも RLS の知識表現形式により記述されている）をメタ知識生成器 (MKG) により生成してその手続き的知識を修正することである．必要に応じてより高位の学習も行う．補助的手段として一般的学習手法も用いる．現行の RLS では EBG を採用している．

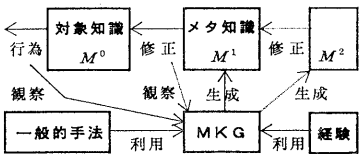

図 1 学習モデル
Fig. 1 The learning model.

RLS は EBG の動作を繰り返し実行することで，長期間にわたる手続きの実行結果からの学習を可能にする．RLS-Othello の EBG は次のようなものである．自分が打着した手から悪手 (bad_move) の概念を得ることを目標にする．補助的な目標概念としては良手 (good_move) も用いる．打着後の局面から作られる訓練例とあらかじめ用意される領域理論を用いて目標概念を学習する．

RLS および RLS-Othello はオブジェクト指向論理型言語 ESP で記述し，MELCOM PSI/UX で動作させた．

3. 複雑なパタンの事例

図 2 に本論文で使用する事例を示す．RLS-Othello は自分の石を増やす戦略に基づいて打着すると仮定する．また隅を取る手に高い価値があるという知識を持っているものとする．RLS-Othello 側が白，相手が黒である．図(a)，(b)，(c)，(d)はそれぞれ，35手済みの局面，棋譜，36手済みの局面，47手済みの局面である．白 e8 により出現する局面(c)において左側および下側の辺および中辺（b3～b6 および c7～f7) が真っ白になっているが，これは双翼 (pair wing) と呼ばれる非常に悪い部分パタンのなかでも最悪のものであることが知られている[11]．なぜなら，次の手，黒 b7 のあとで白は左下隅を取れるのだが，その結果として黒は a7，b8 に入ることが可能になり，続い

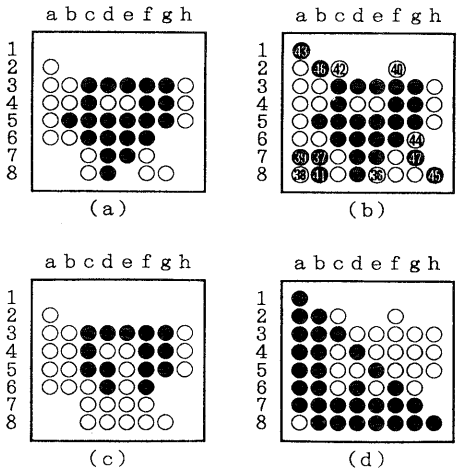

図 2 事例
Fig. 2 A case.

て黒が a1 および h8 の両隅を取ることができるからである．従って 36 手目の白 e8 が悪手であることを学習する必要がある．双翼はよく現れる典型的悪形であり，また最も複雑な悪形なので，学習事例として最適である．

4. 時刻入り領域理論の設計

4.1 基本的なアイデア

従来のわれわれの研究では目標概念は

bad_move(Move)

という述語であった．EBG では基本的に次の節で示される関係のみを利用した．ここで「∧」は連言を，「←」は含意を意味する．

自分の手 m が悪手←将来の敵の手 n が良手
∧m が n の原因である （1）

そこでは「m が n の原因である」には直接的な原因となる領域理論のみが用意された．従って何段にもわたるような高度な推論を用いることができなかった．これを解決するためにわれわれは，2種の引数を追加した次の述語を目標概念として用いることにする．

bad_move(T, P, M)

ここで T, P, M はそれぞれ手番号 (1, 2, …, n)，打着者 (b または w)，手 ([I|J]，ここで I, J＝1, 2, 3, 4, 5, 6, 7, 8, pass すなわち盤上の位置の座標またはパス)を意味する．そして(1)式の代わりに次の知識を利用する．

自分の手 m が悪手←将来の敵の手 n が良手
∧n の原因が m の時点に存在する． （2）

さらに節(2)の最後のリテラルは次に図式的に示すような再帰的な知識を利用する．

n の原因が m の時点に存在する
←(n が)取れた原因
取れた原因
←敵石（の列）が残っていた∧軸石があった
（軸石を）取れた原因
←生き残った原因←自手があった
∧軸石がなかった

この図式について説明する．基本的なアイデアは，m の時点で既に n の原因（本事例では最悪の双翼）が存在していることを示し，その原因を m の時点で導かれる概念を構成するための材料とすることである．n に打着できたということは敵の石（の列）を取れたということである．取れるためには取られるべき敵石（の列）が残っていて，かつ反転するために反対側の自石（軸石と呼ぶことにする）をもっている必要があ

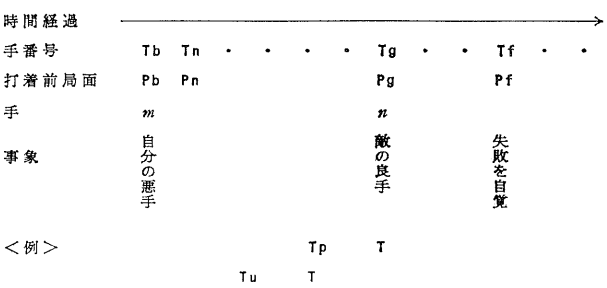

図 3 推論の時点
Fig. 3 Moments for inference.

たとえば図4の(8)における M がここでの手番号 T の n ならば図4(9)の軸石の手番号が Tp である．図4(11)で軸石を調べるとすると，今度は図4(8)の M がその軸石になり，図4(12)の「過去の自手」の手番号が Tu となる．

表 1 領域理論中の述語
Table 1 Predicates in the domain theory.

目標概念	
bad_move(N,P,M)	N手目のPの手Mは悪手である
good_move(N,P,M)	N手目のPの手Mは良手である
訓練例	
move(N,P,M)	MはN手目のPの手である
move(P,M)	MはPの手である
pre_state(N,L,S)	SはN手目打着前のLの状態である
マスの種類	
corner(L)	Lは隅のマスである
hoshi(L)	Lは星のマスである
la(L)	LはAのマスである
lb(L)	LはBのマスである
lc(L)	LはCのマスである
at(L)	Lは@のマスである
sa(L)	Lはaのマスである
マスの相対的位置	
near_la(E,Corner,L)	Lは辺E上でのCornerに近いAである
near_lb(E,Corner,L)	Lは辺E上でのCornerに近いBである
near_lc(E,Corner,L)	Lは辺E上でのCornerに近いCである
far_lc(E,Corner,L)	Lは辺E上でのCornerに遠いCである
far_lb(E,Corner,L)	Lは辺E上でのCornerに遠いBである
far_la(E,Corner,L)	Lは辺E上でのCornerに遠いAである
near_at(E,Hoshi,L)	Lは中辺E上でのHoshiに近い@である
near_sa(E,Hoshi,L)	Lは中辺E上でのHoshiに近いaである
far_sa(E,Hoshi,L)	Lは中辺E上でのHoshiに遠いaである
far_at(E,Hoshi,A)	Lは中辺E上でのHoshiに遠い@である
対局者関係	
opponent(P,O)	PはOの敵である
空間的関係	
corner_pair(C1,C2)	C1とC2は隅の対である
edges2(E1,E2)	E1とE2は隣接辺である
mid_edge(L,M,E)	EはLとMが属する中辺の位置である
direction(L,N,D)	DはLからNへの方向である
next(L,D,N)	Nは位置Lの方向Dの次の位置である
neighbor(L,N)	NはLに接する位置である
時間的関係	
just_before(T,U)	TはUの直前の手番号である
earlier_than(T,U)	TはUよりも前の手番号である
中間的利用用述語	
c_can_take(Tb,T,P,M,..)	TbにTでPがMを取れた原因がある
state_continue(Tb,T,L,S)	Tbの次からTまで位置Lの状態がSである
upset(Tb,T,P,D,M)	Tで方向Dの隣Nの敵石を反転した
has_pivot(Tb,T,P,D,M)	マスMから方向D上のマスにPの軸石がある
unchanged(Tb,T,P,N)	マスNの石は過去のPの手では反転されなかった
no_pivot(Tb,Tu,P,N,N2)	N2からNへの方向にはPの軸石はなかった
future(Clause)	Clauseが将来成立つことを示すメタ述語
補助述語	
assume_true	真と仮定

マスの種類	
	隅
星	C
@	A
a	B
a	B
@	A
星	C
	隅

る．敵の石が残っているためにはそれ以前の自分の手がその敵石を取ってしまっていてはいけないが，このためには反転するのに必要な軸石が存在しないことが必要である．このように原因追求を必要な深さまで再帰的に繰り返すことで，n の原因となった m の時点での部分パタンを推論する．

図3に時間経過と推論の時点の例を示す．Tb が m の，すなわち m を悪手と判定すべきだった時点（**基準時点**と呼ぶことにする），Tf が失敗を自覚（認識）した時点（**失敗認識時点**と呼ぶことにする）の手の番号である．打着前局面とは，その手の打着前の局面を示す．敵側の良手の手番号を Tg とする．

4.2 述語の種類

上で述べたような推論を行うために必要な領域理論を次のように設計する．オセロゲームの領域における理論を節形式で記述する．**表1**に事例における学習に直接関係する述語を示す．各種類ごとに説明する．

（a） 目標概念と訓練例

EBG の動作に必要な目標概念および訓練例である．これらについては5章で詳述する．

（b） マスの種類と相対的位置

オセロ盤の各マスの種類を用意する．また辺ならば隅からの相対的な関係，中辺ならば星からの相対的な関係について，特に述語を用意する．

（c） 対局者関係

黒番と白番が互いに相手であることを示す．

（d） 空間的な（盤上の）関係

盤上の重要な部分あるいは位置の間の関係を示す．

（e） 時間的な関係

手番号の時間的な関係を示す．

（f） 中間的利用用述語

学習のための推論の中間段階で必要となる各種述語である．4.3 節で利用する場面を詳述する．

（g） 補助述語

探索を打ち切るために便宜上真と仮定するために導入した引数なしの述語である．

4.3 節の種類

図4に事例の学習で使用する領域理論の主要部分を，また**図5**にそのなかのマスを取れた原因の部分（図4の8行目以下）の詳細を示す．

```
( 1) 良手 ← 重要なパタンを生成 ∨・・・・・・・
( 2) 重要なパタン ← 隅の対 ∨・・・・・・・

( 3) 悪手 ← 致命的なパタンを生成 ∨・・・・・・・
( 4) 致命的なパタンを生成 ← 将来の敵良手 ∧ 良手の原因
( 5) 良手の原因 ← 良パタンの原因∨・・・・・・・
( 6) 良パタンの原因 ← 隅の対を取れた原因 ∨・・・・・・・
( 7) 隅の対を取れた原因 ← 隅1を取れた原因 ∧ 隅2を取れた原因
( 8) Mを取れた原因 ← Mが空 ∧ 列Lを取れた
( 9) 列Lを取れた ← 隣のマスNの敵石を反転した ∧ Kに軸石があった
                ∧ 軸石まで敵石列
(10) Nの敵石を反転した ← Nに敵石 ∧ Nは過去の自手では反転されなかった
(11) Kに軸石があった ← 過去にKに打着した ∧ Kを取れた原因
(12) Nは過去の自手では反転されなかった ← 過去にNの隣N2に打着
                ∧ N2からNへの方向に軸石がなかった
                ∧ Nは（別の）過去の自手では反転されなかった
(13) 軸石がなかった ← 空マスまで敵石列
```

図4 領域理論の主要部分
Fig. 4 Main part of the domain theory.

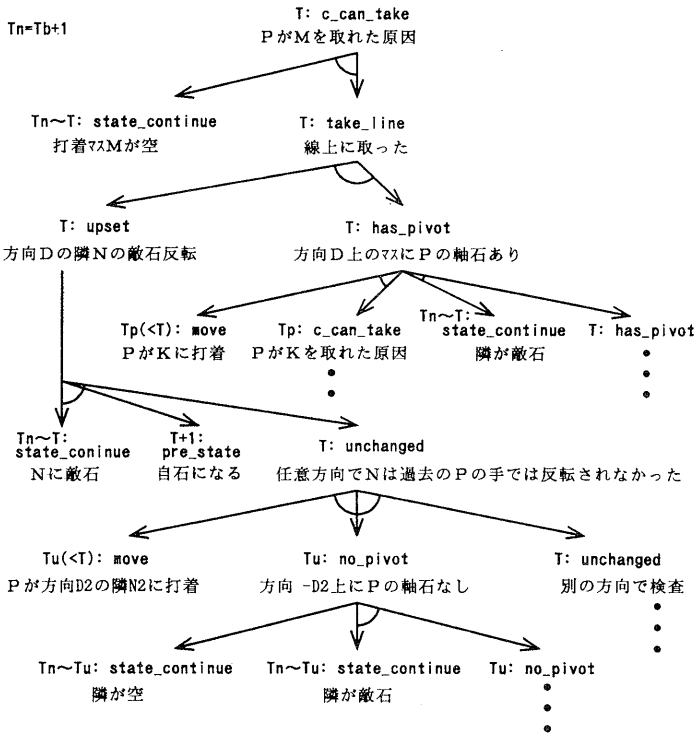

図5 マスを取れた原因
Fig. 5 Cause of being able to take a square.

図4について説明する．選言記号以降の部分は実際には別の節で表現される．この理論を利用して最終的に得られる盤面部分パタンは，図中の 8, 9, 13 番の節中の下線の部分の連言として得られる．盤外などでの例外処理については省略した．各節について説明する．

(1)敵の良手についての条件．
(2)重要なパタンについての知識．
(3)自分の悪手についての条件．
(4)将来敵に良手ができ，その原因が今自分が生成したパタンにあるなら，致命的なパタンを生成したのである．
(5)良パタンを生むことは良手のひとつの条件である．
(6)隅の対を取れたというのは良パタンのひとつの条件である．
(7)それぞれの隅について原因を調べる．
(8)一般的にあるマスを取れたことに対する条件はゲームの規則から得られる．
(9)隣の敵石，および同方向の自分の軸石が必要である．
(10)隣に石があったということは，過去の自分の石による反転の機会にも反転されなかったということである．
(11)軸石に対しても(8)を再帰的に調べる．
(12)過去に反転されなかったことは，過去のすべての自分の手に対して軸石がなかったということである．
(13)敵石（の列）の先が空ならば軸石はなかった．

図5では各段階で使用している述語を，それが成立する時点と共に示してある．ここで述語 state_continue/4 について説明する．これはマスの状態が2つの時点の間同じ状態を持続することを示す述語で，基準時点で既に悪手の原因があったことを示すのに必要な重要な述語である．これは pre_state/3 と just_before/2 および自身 state_continue/4 により再帰的に定義する．よってこの述語の呼び出しは多数のリテラルの展開を引き起こす．そしてその展開結果が悪手概念を定義する部分パタンを構成する．

4.4 仮　　定

領域理論は基本的には上記のようであるが，実際に利用する際にはいくつかの仮定をもうける必要がある．

（1） 探索の限度

再帰的に「マスを取れた原因」を追求しているが，現実的な処理のためには探索を制限する必要がある．追求するマスを限ることでこれを行う．ここでは次の2点を基準としている．

（a） マスの種類

RLS-Othello では盤の最外周とそのひとつ内側のマス，すなわち隅，星，辺，中辺のみが重要であるという知識を先験的に与えている．それ以外のマスでは再帰的な探索を行わない．そこには軸石はあるものと仮定している．このために仮の真を示す．assume_true/0 という述語（命題）を用意している．

（b） マス間の距離

一般的に言えばある打着は盤上のすべてのマスに何らかの影響を及ぼす．特に多くの敵石を反転した場合には，原則的にはそれらすべての敵石についてなぜそれが敵石の状態で残っていたかの原因を追究することができる．しかしここでは簡単のため隣接する石にのみ注目する．図4の9節目および 12 節目でこの制限を行っている．

（2） 将来の事象

時刻付きの領域理論を利用して学習される概念には一般に将来の事象が含まれる．しかし打着時には将来の事象の真偽は不明であるから，何らかの仮定をするほかはない．われわれは将来の事象はすべて真と仮定し，最終的な概念の記述（以後**最終記述**と呼ぶ）から将来の事象に関するリテラルを除去する．特に state_continue/4 の展開から各時刻でのリテラルが多数生じるが，その大半が除去される．

4.5 便宜上の述語

前述の assume_true の必要性について述べる．われわれは述語 p を真と仮定する場合に

```
p :- assume_true.                    (3)
```

と書くことにした．通常の証明では特に必要のないこの述語は，EBG で学習される概念の記述に p のリテラルが入ってしまうのを回避するためである．もちろん assume_true/0 は最終記述から除去する．

5. その他の EBG の要素

本事例での，領域理論以外の EBG の各要素を示す．

5.1 目 標 概 念

表1で既に示したが，目標概念としては次の2種類がある．

bad_move(N, P, M)	N手目のPの手Mは悪手である

good_move(N, P, M) N手目のPの手Mは良手である.

このうち, bad_move/3 が最終的に手続きに変換される. good_move/3 はその補助に利用される. われわれはこれらを, それぞれ「**利用可能な概念**」,「**有用な概念**」と呼んでいる.

これらには時刻（手番号）と打着側が変数として入っている. これらは EBG の動作時に必要になる. 対局時に必要になるかどうかは学習時点では不明だが, 不要かも知れない. 本事例では不要である.

5.2 操作性基準

RLS は EBG の動作を受け持つ ebg と名付けられたクラスをもつ. これは与えられた目標概念, 訓練例, 領域理論から概念を ESP の証明機構を利用することにより学習する. ebg で証明できることがすなわち操作できることである. ESP での各種の評価可能述語はここでも評価可能である. そのほか, ごく基本的な空間的あるいは時間的な関係に関する述語はこのクラスで評価できるようにしてある. 学習される概念はすべてクラス ebg で評価できる節の形式で記述される. クラス ebg で特別に評価する述語は, 表1に記したもののうち以下のものである.

neighbor(L,N), direction(L,N,D), next(L,D,N), just_before(T,U), earlier_than(T,U).

5.3 訓 練 例

RLS がもつモジュール observe_position により基準時点の局面から pre_state/3 から成る単位節および move/3 から成る単位節を生成する. それぞれ盤面の各マスの状態, そのときの打着手を示す. ある基準時点で生成された pre_state/3 および move/3 は基準時点が後退しても後の EBG 用にすべて残しておく. 文献1)の事例と異なり, 各基準時点では失敗判定時点と基準時点の間のすべての状態や手を訓練例として利用するためである.

6. 学 習 例

3章で導入した事例について, その対局実行および学習の例を示す.

6.1 対 局 実 行

図6に全体の流れを示す. RLS-Othello は

$v=d+10c$

が最大の手を打着する（ただし d, c はそれぞれ石数差, 獲得隅数差）. 自分の石が少なくなると RLS-Othello はこの手続きの実行を失敗とみなし, 自分のもつ手続き的知識の何らかの部分（知識片）に欠陥があると判断する. RLS は失敗を認識すると学習モードに入る.

6.2 学 習

図6に沿って説明する. 開始時の基準時点は失敗認識時点であるが, 学習が成功するか一定の限界に至るまで対局を遡って行く. 訓練例と与えられている領域理論から EBG が実行される. 利用可能な概念が得られたら後処理（6.3節で後述）に進み学習は終了する. 利用可能な概念は得られなくても有用な概念が得られればその情報を次の EBG 動作用に格納する. 利用可能な概念が得られなければ基準時点を1手過去に後退させ学習を続行する.

本事例の開始局面を与えて RLS を実行すると RLS

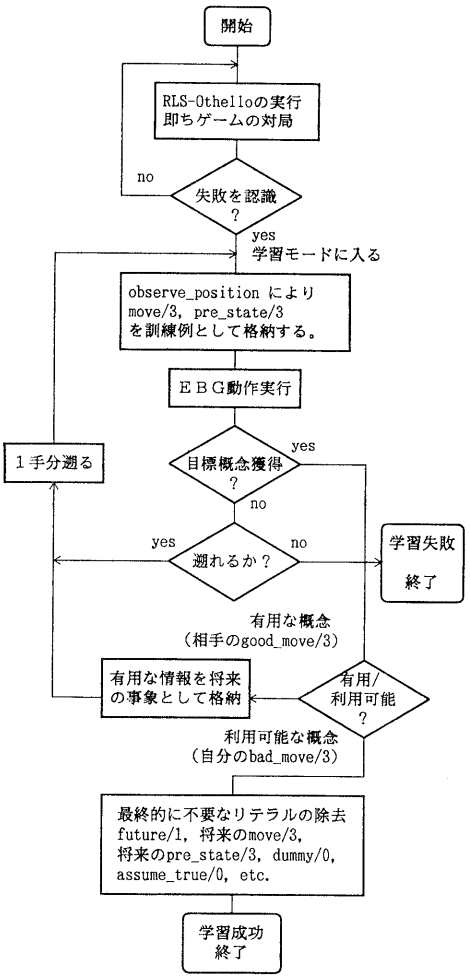

図 6 学習の流れ
Fig. 6 Flow of the learning.

は，図3における Tg の時点で敵の良手として，敵側(黒)が2か所の隅を獲得したということを有用な概念(good_move/3)として獲得する．さらに RLS の実行を続けると RLS は，敵にその良手が生じた原因を推論することにより利用可能な概念(bad_move/3)を獲得する．**図7**に学習の結果得られた概念を示す．得られた概念 bad_move/3 の本体部は 238 のリテラルから成り，非常に長い．

本体の最初の部分に move/3 に加えて move/2 があるが，これは基準時点を特定するために必要な便宜上の措置である（move/2 は基準時点のものしか同時には存在しないようにしてある）．この節では変数Aが基準時点である．リテラルの多くはstate_continue/4 を展開した結果生じたものである．

PSI/UX による実験について説明する．目標概念としては文献1)で利用した bad_move/1 と good_move/1 をも bad_move/3 と good_move/3 と併用して実験した．領域理論としては文献 1) で利用したものと本例に必要な4章で示したものを両方含んだものを利用した．本事例を実行したところ，述語呼び出しの回数は 7,553 回，EBG に要する学習時間は1分7秒であった．これは手を遡ることで繰り返されるすべての EBG での数値を合計したものである．

6.3 後 処 理

図7の結果は基準時点以外の将来の事象を多数含んでおり，そのまま利用することができない．図6に示したように，RLS は学習の最終段階で最終的に不要なリテラルの除去を行う（図には示していないが，実際にはこのあとで手続き的知識への変換を行う）．

図8に最終的に得られる概念 bad_move/3 を示す．期待どおり，最悪の双翼を生成する手が悪手であることを示している．節の本体部は 116 個のリテラルを含む．まだ数は相当大きいが，最悪の双翼という複雑なパタンを表現しているのでこの程度の数はやむを得ない．

除去された主なリテラルは，将来時点で真偽が決定されるようなリテラルである．基準時点の局面とその次の時点の局面（これは手の仮打ちで容易に実現できるので）から決定できるリテラルが残される．除去されたリテラルは次の述語をもつものすべて

future/1, dummy/0, assume_true/0 (dummy/0 は

```
bad_move(A,B,C):-move(B,C),move(A,B,C),opponent(B,D),future(good_move(E,D,F)),
move(E,D,F),corner_pair(F,G),move(H,D,G),earlier_than(H,E),edges2(I,J),
pre_state(E,F,v),just_before(K,E),pre_state(K,F,v),just_before(H,K),
pre_state(H,F,v),just_before(L,H),pre_state(L,F,v),just_before(M,L),
pre_state(M,F,v),just_before(N,M),pre_state(N,F,v),just_before(O,N),
pre_state(O,F,v),just_before(P,O),pre_state(P,F,v),just_before(Q,P),
just_before(A,Q),pre_state(Q,F,v),corner(F),near_lc(I,F,R),direction(F,R,S),
pre_state(E,R,B),pre_state(K,R,B),pre_state(H,R,B),pre_state(L,R,B),
pre_state(M,R,B),pre_state(N,R,B),pre_state(O,R,B),pre_state(P,R,B),
pre_state(Q,R,B),just_before(E,T),pre_state(T,R,D),lc(R),next(F,S,R),
next(R,S,U),la(U),near_la(I,F,U),pre_state(E,U,B),pre_state(K,U,B),
pre_state(H,U,B),pre_state(L,U,B),pre_state(M,U,B),pre_state(N,U,B),
pre_state(O,U,B),pre_state(P,U,B),pre_state(Q,U,B),next(U,S,C),lb(C),
near_lb(I,F,C),pre_state(E,C,B),pre_state(K,C,B),pre_state(H,C,B),
pre_state(L,C,B),pre_state(M,C,B),pre_state(N,C,B),pre_state(O,C,B),
pre_state(P,C,B),pre_state(Q,C,B),next(C,S,V),lb(V),far_lb(I,F,V),
pre_state(E,V,B),pre_state(K,V,B),pre_state(H,V,B),pre_state(L,V,B),
pre_state(M,V,B),pre_state(N,V,B),pre_state(O,V,B),pre_state(P,V,B),
pre_state(Q,V,B),next(V,S,W),la(W),far_la(I,F,W),pre_state(E,W,B),
pre_state(K,W,B),pre_state(H,W,B),pre_state(L,W,B),pre_state(M,W,B),
pre_state(N,W,B),pre_state(O,W,B),pre_state(P,W,B),pre_state(Q,W,B),
next(W,S,X),lc(X),far_lc(I,F,X),move(M,D,X),earlier_than(M,E),pre_state(M,X,v),
pre_state(N,X,v),pre_state(O,X,v),pre_state(P,X,v),pre_state(Q,X,v),at(Y),
neighbor(X,Y),direction(X,Y,Z),pre_state(M,Y,B),pre_state(N,Y,B),
pre_state(O,Y,B),pre_state(P,Y,B),pre_state(Q,Y,B),pre_state(L,Y,D),
move(Q,D,A1),earlier_than(Q,M),neighbor(Y,A1),hoshi(A1),direction(Y,A1,S),
mid_edge(Y,A1,I),direction(A1,Y,B1),pre_state(Q,A1,v),next(A1,B1,Y),
near_at(I,A1,Y),next(Y,B1,C1),sa(C1),near_sa(I,A1,C1),pre_state(Q,C1,B),
next(C1,B1,D1),sa(D1),far_sa(I,A1,D1),pre_state(Q,D1,B),next(D1,B1,E1),at(E1),
far_at(I,A1,E1),pre_state(Q,E1,B),next(E1,B1,F1),hoshi(F1),pre_state(Q,F1,v),
dummy,next(X,Z,Y),next(Y,Z,G1),assume_true,pre_state(H,G,v),pre_state(L,G,v),
pre_state(M,G,v),pre_state(N,G,v),pre_state(O,G,v),pre_state(P,G,v),
pre_state(Q,G,v),corner(G),near_lc(J,G,H1),direction(G,H1,I1),
pre_state(H,H1,B),pre_state(L,H1,B),pre_state(M,H1,B),pre_state(N,H1,B),
pre_state(O,H1,B),pre_state(P,H1,B),pre_state(Q,H1,B),pre_state(K,H1,D),lc(H1),
next(G,I1,H1),next(H1,I1,J1),la(J1),near_la(J,G,J1),pre_state(H,J1,B),
pre_state(L,J1,B),pre_state(M,J1,B),pre_state(N,J1,B),pre_state(O,J1,B),
pre_state(P,J1,B),pre_state(Q,J1,B),next(J1,I1,K1),lb(K1),near_lb(J,G,K1),
pre_state(H,K1,B),pre_state(L,K1,B),pre_state(M,K1,B),pre_state(N,K1,B),
pre_state(O,K1,B),pre_state(P,K1,B),pre_state(Q,K1,B),next(K1,I1,L1),lb(L1),
far_lb(J,G,L1),pre_state(H,L1,B),pre_state(L,L1,B),pre_state(M,L1,B),
pre_state(N,L1,B),pre_state(O,L1,B),pre_state(P,L1,B),pre_state(Q,L1,B),
next(L1,I1,M1),la(M1),far_la(J,G,M1),pre_state(H,M1,B),pre_state(L,M1,B),
pre_state(M,M1,B),pre_state(N,M1,B),pre_state(O,M1,B),pre_state(P,M1,B),
pre_state(Q,M1,B),next(M1,I1,N1),lc(N1),far_lc(J,G,N1),move(O,D,N1),
earlier_than(O,H),pre_state(O,N1,v),pre_state(P,N1,v),pre_state(Q,N1,v),at(O1),
neighbor(N1,O1),direction(N1,O1,Z),pre_state(O,O1,B),pre_state(P,O1,B),
pre_state(Q,O1,B),pre_state(N,O1,D),earlier_than(Q,O),neighbor(O1,A1),
direction(O1,A1,I1),mid_edge(O1,A1,J),direction(A1,O1,P1),next(A1,P1,O1),
near_at(J,A1,O1),next(O1,P1,Q1),sa(Q1),near_sa(J,A1,Q1),pre_state(Q,Q1,B),
next(Q1,P1,R1),sa(R1),far_sa(J,A1,R1),pre_state(Q,R1,B),next(R1,P1,S1),at(S1),
far_at(J,A1,S1),pre_state(Q,S1,B),next(S1,P1,T1),hoshi(T1),pre_state(Q,T1,v),
next(N1,Z,O1),next(O1,Z,U1)
```

図7 EBG により得られた概念
Fig. 7 The concept obtained by EBG.

```
bad_move(A,B,C):-move(B,C),move(A,B,C),opponent(B,D),corner_pair(E,F),
edges2(G,H),just_before(A,I),pre_state(I,E,v),corner(E),near_lc(G,E,J),
direction(E,J,K),pre_state(I,J,B),lc(J),next(E,K,J),next(J,K,L),la(L),
near_la(G,E,L),pre_state(I,L,B),next(L,K,C),lb(C),near_lb(G,E,C),
pre_state(I,C,B),next(C,K,M),lb(M),far_lb(G,E,M),pre_state(I,M,B),next(M,K,N),
la(N),far_la(G,E,N),pre_state(I,N,B),next(N,K,O),lc(O),far_lc(G,E,O),
pre_state(I,O,v),at(P),neighbor(O,P),direction(O,P,Q),pre_state(I,P,B),
neighbor(P,R),hoshi(R),direction(P,R,K),mid_edge(P,R,G),direction(R,P,S),
pre_state(I,R,v),next(R,S,P),near_at(G,R,P),next(P,S,T),sa(T),near_sa(G,R,T),
pre_state(I,T,B),next(T,S,U),sa(U),far_sa(G,R,U),pre_state(I,U,B),next(U,S,V),
at(V),far_at(G,R,V),pre_state(I,V,B),next(V,S,W),hoshi(W),pre_state(I,W,v),
next(O,Q,P),next(P,Q,X),pre_state(I,F,v),corner(F),near_lc(H,F,Y),
direction(F,Y,Z),pre_state(I,Y,B),lc(Y),next(F,Z,Y),next(Y,Z,A1),la(A1),
near_la(H,F,A1),pre_state(I,A1,B),next(A1,Z,B1),lb(B1),near_lb(H,F,B1),
pre_state(I,B1,B),next(B1,Z,C1),lb(C1),far_lb(H,F,C1),pre_state(I,C1,B),
next(C1,Z,D1),la(D1),far_la(H,F,D1),pre_state(I,D1,B),next(D1,Z,E1),lc(E1),
far_lc(H,F,E1),pre_state(I,E1,v),at(F1),neighbor(E1,F1),direction(E1,F1,Q),
pre_state(I,F1,B),neighbor(F1,R),direction(F1,R,Z),mid_edge(F1,R,H),
direction(R,F1,G1),next(R,G1,F1),near_at(H,R,F1),next(F1,G1,H1),sa(H1),
near_sa(H,R,H1),pre_state(I,H1,B),next(H1,G1,I1),sa(I1),far_sa(H,R,I1),
pre_state(I,I1,B),next(I1,G1,J1),at(J1),far_at(H,R,J1),pre_state(I,J1,B),
next(J1,G1,K1),hoshi(K1),pre_state(I,K1,v),next(E1,Q,F1),next(F1,Q,L1)
```

図 8 最終的に得られた概念
Fig. 8 The concept finally obtained.

完全にプログラミング技巧上のものである）および次の述語をもつものの一部である．

just_before/2,
earlier_than/2,
move/3, pre_state/3.

次の2種類の述語をもつリテラルは最終的には不要なのにもかかわらず除去されない．

（1） マスの種類についての述語の重複

特に重要なマスの位置の関係についての述語がある．near_x/3 や far_x/3 などで，これらが概念の表現に入っていれば，例えば near_lc(G, E, J) が入っていれば lc(J) は不要であるが，このシステムでは特に除去していない．隣接関係などを用いる一般的な表現も残っている．

（2） 使われない変数用述語

opponent/2 は相手を指定するための述語であるが，ここでの概念の表現に相手の変数は不要であった．システムはこれらのチェックをしていない．

7. システムの考察

時刻（手の番号）を導入することで事象の厳密な時間的関係を利用できるようになった．このようにして一般的な領域理論を用いて高度な悪パタンに関する知識を学習することが可能である．本論文では複雑な悪形の典型例として双翼を示した．翼[11]（片側）は本事例の場合の半分の解析で学習できた．また双翼と性質の似たグライダ[11]も good_move の知識を高度にすることで同様に学習できる．4.1 節（2）はさらに変形することも可能で，例えば「敵の手 n が良手」に対し

敵の手 n が良手
←将来に自分が悪手 m' しか打てない∧…

を用意して，悪手と良手が交互に現れる鎖を延長し，さらに高度な推論を行うことも可能である．

領域理論には事象の因果関係のほかに空間的あるいは時間的な関係を表す節が使われる．これの種類や数をどの程度用意するかは問題に依存するが重要である．探索の制御は注目する石をあらかじめ選択することで行っているが，その範囲はより広げるのが望ましい．特に隣接石にのみ限っているのは不当な制限なので，より柔軟に必要な探索を行うべきである．

学習された概念が不要なリテラルを含んでいるなど，最終記述が冗長であるのは，打着時の判定に余分の時間がかかるので問題である．リテラル間の依存関係の知識を用意すればこの冗長性を除去できるが，依存関係の知識を利用するコストとのトレードオフを考慮する必要がある．

このシステムでは探索が制限されているため，盤面パタンはそれほどむやみに増えないが，多数の新概念が無秩序に蓄積されることにより効用問題が生じることが予想される．これの解決には得られた概念を一般化した戦略を作成することで，個々の概念の節を不要とするような，知識の再構成のための別の学習機構を用意する必要がある．

8. 結 び

RLS が補助的に利用している EBG の機構における，目標概念，領域理論，訓練例，の各述語に時刻（手番号）を導入することにより，オセロゲームの双翼のような複雑な概念を学習することができた．

このシステムはさらに改良の余地がある．現行の領域理論では，盤上の重要な部分が与えられている．つまりそのようなパタンが，あるいはそのような打着が可能であった原因をさらに追求すべきかどうかは領域理論に初期状態として与えられている．現在の RLS-Othello で隅／星／辺／中辺を注視できているのはこれが理由である．しかし，実際の問題ではそのような注視すべき場所をも学習する能力が必要である．

大規模な領域理論を効率的に扱うため，RLS のも

つ再帰的な学習の階層構造に基づきさらに上位のメタ知識により学習用知識を制御し，領域理論を分割し，併用したり選択的に使用することを考えており，このための装置を開発済みである[12]．今後，領域理論の適切な分割法と利用法についても研究を進める予定である．

謝辞 本研究に対してのご支援ご討論に対し当所推論研究室の諸氏に感謝します．本研究は新世代コンピュータ技術開発機構(ICOT)との共同研究です．ICOT の関係各位に感謝します．

参考文献

1) 田島守彦，実近憲昭：宣言的知識の利用による RLS の拡張，情報処理学会論文誌，Vol. 34, No. 5, pp. 820-830 (1993).
2) 田島守彦：手の価値を評価する手法を用いるゲーム対局プログラムの研究，電子技術総合研究所研究報告，第 954 号 (1993).
3) Mitchell, T. M., Keller, R. M. and Kedar-Cabelli, S. T.: Explanation-Based Generalization: A Unifying View, *Machine Learning*, Vol. 1, pp. 47-80 (1986).
4) Minton, S.: Selectively Generalizing Plans for Problem-Solving, *Proc. of IJCAI '85*, pp. 596-599 (1985).
5) 沼尾正行，丸岡 孝，志村正道：説明の部分構造抽出による高速化学習，人工知能学会誌，Vol. 7, No. 6, pp. 86-94 (1992).
6) Minton, S. et al.: Acquiring Effective Search Control Rules: Explanation-Based Learning in the PRODIGY System, *Proc. of Int. Workshop on Machine Learning*, pp. 122-133 (1987).
7) Samuel, A.: Some Studies in Machine Learning Using the Game of Checkers. II—Recent Progress, *IBM J. Res. Dev.*, Vol. 11, pp. 601-617 (1967).
8) Murray, A. M. and Elcock, E. W.: Automatic Description and Recognition of Board Patterns in Go-moku, *Machine Intelligence*, Vol. 2, pp. 75-88 (1968).
9) Pitrat, J.: A Chess Combination Program which Uses Plans, *Artif. Intel.*, Vol. 8, pp. 275-321 (1977).
10) Quinlan, J. R.: Learning Efficient Classification Procedures and Their Application to Chess End Games, Michalski, R. S. et al (eds.) *Machine Learning*, pp. 463-482, Springer-Verlag (1984).
11) 井上 博：逆転の発見，(株)企画室ネコ，東京 (1977).
12) 田島守彦：RLS における EBG 用領域理論の柔軟な構造，第 47 回情報処理学会全国大会論文集，2 N-8 (1993).

(平成 6 年 5 月 11 日受付)
(平成 6 年 9 月 6 日採録)

田島 守彦 (正会員)

1947 年生．1969 年東京大学工学部計数工学科卒業．同年電気試験所(現電子技術総合研究所)入所．現在知能情報部推論研究室主任研究官．入所以来，拡張閾値論理回路，並列計算機システム，論理型言語による知識処理の研究を行う．現在ゲーム学習などの機械学習を中心とする人工知能の研究に従事．工学博士．情報規格調査会 SC 22/Prolog WG 幹事．電子情報通信学会，人工知能学会各会員．

実近 憲昭 (正会員)

1939 年生．1965 年東京大学工学部電気工学科卒業．同年電子技術総合研究所入所．論理システムの研究に従事．1989-1992 年 AI 言語研究所(株)にてオブジェクト指向論理型言語の開発を行う．現在，電子技術総合研究所，分散システム研究室所属．主任研究官．ゲームを題材とする人工知能の研究に興味を持つ．電気学会会員．