人 工 知 能 78－9
（1991．9．12）

実例に基づく強化学習法

畝見 達夫
長岡技術科学大学 工学部 計画・経営系
unemi@voscc.nagaokaut.ac.jp

強化学習法で扱われる制御問題に適用可能な実例に基づく学習の機構を提案する．各学習サイクルにおける入力は実数のベクトル，出力はあらかじめ決められた有限個のシンボルの中の１つであり，学習の目的は正の強化入力をできるだけ頻繁に得ることである．学習システムは，生のまま記憶されたすべての入出力対を走査し，現状から近い将来に正の強化入力が得られそうな出力データを見つけ，出力とする．さらに，記憶容量を制限し，実行結果のフィードバックを利用した忘却を導入することによって，学習能力を向上させることができる．適応行動シミュレーションへの応用を通して，この学習機構が有効に働くことを示す．

Instance-Based Reinforcement Learning Method

Tatsuo UNEMI
Department of Planning and Management Science, Nagaoka University of Technology
1603-1 Kamitomioka-cho, Nagaoka, Niigata 940-21, Japan
unemi@voscc.nagaokaut.ac.jp

This paper proposes a reinforcement learning method based on an instance-based learning approach. The objective of the learner is to get positive reinforcement from the environment more frequently, the input on each learning cycle is a vector of real numbers, and the output is a symbol selected from *a priori* known finite set. By scanning all of the past experiences stored in memory verbatim, the learner searches a suitable output value which will likely lead to high positive reinforcement. Experimental results show these mechanisms work well to simulate an adaptive behavior, and some methods of forgetting is effective.

1 はじめに

本論文では，実例に基づく学習アルゴリズムを基に，強化学習で扱われる制御問題に適用可能な学習アルゴリズムを提案する．

計算機の処理能力の向上に伴い，学習研究のテーマにも変化が起きつつある．その1つが「記憶に基づく[1, 2]」あるいは「実例に基づく[3]」アプローチ，すなわち，多くの訓練例を加工せずに記憶し，問題解決の時点で，なんらかの尺度に従った類似知識検索を行なうことにより，未知の状況に対応しようとするものである．これらの手法は，ある程度の大きさの記憶容量と高速な計算能力を必要とする一方，コネクションマシン[4] などの大規模並列計算機上での実行に適するという点においても注目に値する．訓練例を一般化することによって規則あるいはパターン記述を生成するといった帰納推論[5] あるいは帰納学習[6] の方法とは様々な意味で対照的である[7]．

この種のアプローチは Samuel の先駆的な研究[8] 以来，綴りから発音への変換[1]，自然言語の翻訳[2]，診断システム[4]，簡単なロボット制御[9] ロボットマニピュレータの制御知識の獲得[10] などの応用例が報告されている．また，教師付きの分類問題を対象領域とするアルゴリズムについては，その性質が調べられている[3]．我々は先に，主に生体の環境適応行動をシミュレートする立場から，記憶に基づく学習の枠組に基づいた無限長離散時系列を対象とする学習機構を提案した[11]．ここでは，その学習機構を一般化する形で，学習アルゴリズムとしての確立を試みる．

我々が対象とする「無限長時系列学習問題」は，強化学習法 (reinforcement learning method) [12,13,14, 15,16,17] において対象とされてきた制御問題の一種である．すなわち学習は逐次的に行なわれ，学習システムは各サイクルにおいて得られる入力に対して出力を決定する．その際，出力と環境の状態に応じた強化入力が環境から与えられる．学習の目的は正の強化入力が得られる頻度を高く，負の強化入力が得られる頻度を低くすることである．強化入力には遅れがあってもよい．つまり，ほとんどのサイクルにおいて強化入力は価値の無い (あるいは中立な) ものであって，行為の系列が学習目的に沿うものであるかどうかは各サイクルで直ちに判定することはできず，意味のある強化入力が得られるまで待たなければならない．このような設定は時系列学習問題と通常の分類問題の違いを明確にするものであり，生体の適応モデルあるいは計画や制御への応用を考える上で重要である．

さらに，本論文では，学習システムの出力 (すなわち行為) に伴う環境の変化の文脈独立性を仮定する．つまり，各サイクルにおける入力から出力への写像を実現する関数を獲得すればよいような環境である．この仮定は生体の適応モデルを考える上で，あるいは複雑な実用的応用に対しては，制約として厳しすぎるが，多くの工学領域への応用を妨げるものではない．先に我々が報告した学習機構では，文脈に依存した環境変化を想定していたが，そのような環境に対応できる機構の解説については別の機会に譲り，本論文では制約の緩和について最後に考察として述べるに留めることとする．

以下，対象問題を定式化した後，基本的な学習アルゴリズムを示し，ついで拡張された2つのアルゴリズムについて述べる．1つの拡張は記憶容量に制限を加えるもの．2つ目は成功 / 失敗に基づくフィードバック機構の導入である．同時に各アルゴリズムについて計算機によるシミュレーション結果を示す．

2 無限長時系列学習問題

先の研究[11] で想定された学習主体は，餌を求めて2次元平面の仮想世界を歩き回る Beer の人工昆虫[18] のような「虫」であった．入力は虫の視覚に対応した10次元のベクトルで，各要素は視野に入った物体の種類と距離を反映する -9 から 9 の整数である．出力は虫の方向転換に対応し，-3 から 3 の整数を値とする．学習の目的は世界に配置されたゴミに触らずに，餌を食べることである．物体に触れた時にそれがゴミか餌かの情報が得られる．

本論文での議論の対象となる問題設定は以上の設定をつぎのように一般化することによって得られるものである．すなわち，

- 入力は数次元の数値ベクトルを1つの時点の要素とする無限長の離散時系列である．以下では t サイクル目における入力データ X_t を

  $$(x_{t1}, x_{t2}, \ldots, x_{tn})$$

  と表現する．x_{ti} $(i = 1, 2, \ldots n)$ は実数である．

- 出力はあらかじめ定められた数個の記号からなる集合 Σ_Y の1つの要素を 1時点の要素とする無限長の離散時系列である．すなわち，t サイクル目における出力データ Y_t は $Y_t \in \Sigma_Y$ である．

- 強化入力は正，ゼロ，負の3個の要素のうちの1つを1時点の値とする．すなわち，t サイクル目における強化入力 R_t は $R_t \in \{-1, 0, +1\}$ である．

- 学習システムの性能は，強化入力列の中に占める正の値の密度から負の値の密度を引いた値によって測られる．すなわち t サイクル目までのシステムの性能に対する評価値 P_t は，$P_t = \frac{1}{t}\sum_{i=1}^{t} R_i$ である．

出力をカテゴリ名，入力を分類対象の記述，強化入力を教師信号と見れば教師付きの弁別学習問題とみなすこ

ともできる．しかし，大きく異なるのは，意味のある（0でない正あるいは負の）教師信号が得られるのは特定の入出力対の直後に限られ，また，各サイクルの入力が前サイクルの入出力対に大きく依存するという点である．

3 基本となる学習アルゴリズム

まず，基本となる学習アルゴリズム IBRL0 について述べる．学習は逐次的に行なわれる．学習サイクルの概略的なアルゴリズムはつぎのとおり，一般的な逐次的学習（たとえば赤間の翻訳学習[19]）と同じである．本論文では以下のとおりアルゴリズムの記述に Pascal 風の記法を用いる．

アルゴリズム 1

```
program IBRL0;
var t : integer; (* 時刻 *)
t := 0;
repeat begin
   t := t + 1;
   InData := GetSensoryData;
   Rinfc := GetReinforcement;
   OutData := Policy(InData);
   PutActionData(OutData);
   ModifyMem(InData, OutData, Rinfc)
   end forever
```

以下本節では，まず記憶変更の手続き ModifyMem について述べた後，意思決定手続き Policy について述べる．

3.1 記憶変更の基本手続き

IBRL0 では，すべての経験を記憶するだけ記憶容量が十分豊富にあるものと仮定する．記憶は線形に並んだ無限長の配列を用いる．記憶変更の手続き ModifyMem は入出力データを新たな記憶場所に次々と割り当てる記憶手続きと，強化入力が 0 でない場合に過去の記憶データへ向かって強化入力の記憶を逆伝播させる手続きから構成される．すなわち，ModifyMem の定義はつぎのようになる．

アルゴリズム 2

```
var InMem: array [1..∞, 1..n] of real;
    OutMem: array [1..∞] of symbol;
    RifMem: array [1..∞] of real;
procedure ModifyMem(InData, OutData, Rinfc);
InMem[t] := InData;
OutMem[t] := OutData;
if Rinfc = 0 then RifMem[t] := 0
else begin
   R := Rinfc; I := t;
   while I > 0, |R| > ν and RifMem[I] = 0
   do begin
     RifMem[I] := R;
     R := γ · R;
     I := I − 1
   end
end
```

ここで n は 入力ベクトルの長さである．記憶手続きは，入力データ InData を InMem[t] に，出力データ OutData を OutMem[t] にコピーするだけの単純なものである．

逆伝播手続きでは，強化入力の値に応じて強化入力の記録値を割り当てる．ν は $0 < \nu < 1$ を満たす 0 に近い定数，γ は $0 < \gamma < 1$ を満たす 1 に近い定数である．後で述べる実験では $\nu = 0.05, \gamma = 0.85$ とした．強化入力が 0 の場合には，現サイクルの強化入力の記録値を 0 とする．さもなければ，その値に応じて 1, −1 いずれかの値を定め，徐々にその絶対値を減じながら記憶時刻を過去に遡って，順次，強化入力の記録値を割り当てる．この値は後で述べる意思決定手続きにおいて参照される．また，この逆伝播によって強化入力の遅れによる不都合が回避される．

3.2 意思決定の基本手続き

意思決定手続き Policy では，過去に記憶された入力データの中から，現在の入力データと類似性が高く，かつ期待される強化入力が大きいものを検索し，それと同じサイクルにおいて出力されたデータを現在のサイクルにおいて出力すべきデータとして選択する．ただし，適当なデータが検索できなかった場合には，既知の出力データ集合 Σ_Y の中からランダムに選び出す．Policy のアルゴリズムはつぎのように書ける．

アルゴリズム 3

```
function Policy(InData) : symbol;
if t ≤ 2 then Policy := RandomSelect(Σ_Y);
else begin
   find K in [1..t − 1]
      where RifMem[K] > 0 and which makes
         S(InMem[K], InData) · RifMem[K]
         maximum;
   if K was successfully found
      then Policy := OutMem[K]
      else Policy := RandomSelect(Σ_Y)
end
```

$S(X_p, X_q)$ は2つの入力データ X_p と X_q と間の類似度であり，次の式で定義する．

$$S(X_p, X_q) = 1 - \frac{1}{2n}\sum_{i=1}^{n}\frac{(x_{pi} - x_{qi})^2}{\mathrm{V}_{\mathrm{X}i}} \qquad (1)$$

ここで $\mathrm{V}_{\mathrm{X}i}$ は i 番目の要素についての分散である．分散はデータの2乗の平均と平均の2乗の差として算出できるので，各サイクルでデータとその2乗の平均値を保持することにより容易に算出できる．具体的にはつぎのような各サイクルでの計算により逐次的に求める．

$$E_t = \frac{E_{t-1}\cdot(t-1) + x_t}{t} \qquad (2)$$

ただし，少なくとも2つ以上のデータが得られた後でなければ，類似度の計算はできない．データがなければ E_t は不定となり，データが1つの場合は分散が0となるためである．このような不都合を避けるため，最初の2サイクルは記憶の検索を行なわずランダムに行動することとする．

類似度の最大値は定義より明らかに1となる．平均値はデータの分布に関係なく次式に示すとおり t が大きくなるに従って，つまり実行サイクルが進むに従って0に近くなる．

$$\begin{aligned}\mathrm{E}_{St} &= 1 - \frac{1}{2n}\cdot\frac{2}{t(t-1)}\cdot\sum_{i=2}^{t}\sum_{j=1}^{i-1}\sum_{k=1}^{n}\frac{(x_{ik} - x_{jk})^2}{\mathrm{V}_{\mathrm{X}k}} \\ &= 1 - \frac{1}{2n}\cdot\frac{2tn}{t-1} \approx 0 \qquad (3)\end{aligned}$$

このような一種の正規化を用いる理由は，応用領域の特質による差を吸収するためである．このことは，つぎに述べる強化入力値との組み合わせに基づく意思決定において必須の要件でもある．

現在の入力データと類似性が高く，かつ期待される強化入力が大きい記憶を検索するための尺度として，類似度とその記憶に割り当てられた強化入力の記録値の積を用いる．ただし，強化入力が負の値を持つ記憶は検索の対象から除外する．これにより，2つの要件を共に満たす記憶を見つけ出すことができる．あるいは積ではなく小さい方の値を採用しても良いが，それによって性能が大きく変わることはない．

3.3 基本アルゴリズムの性能

上で提案したような学習アルゴリズムの能力を形式的あるいは解析的に求めることは幾分，困難と思われるのため，シミュレーション実験により，その能力を検証することとした．実験には，先に報告した人工昆虫の環境適応実験に用いられたシミュレータを用いた．環境は図1に示すように餌とゴミとを交互に規則的に配置したトーラス状の2次元平面である．

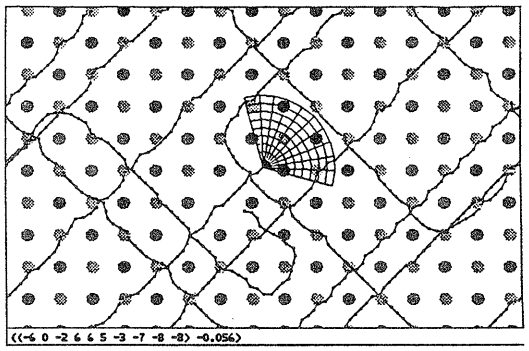


図1: Environment of simulation of adaptive behavior.

図中の灰色の小円は餌，黒の小円はゴミ，中央の扇型は虫の視野である．図中に描かれた折れ線は，典型的な虫の行動パターンを示す500サイクル分の足跡である．図の中央が最終(500サイクル目)の虫の位置である．虫が物体に触れると，その物体は虫が世界の半分の距離だけ遠ざかるまで一時的に消去される．

この問題設定における具体的な入力，出力，強化入力はつぎのとおりである．

- 入力データは虫の視覚に対応した10次元のベクトルとする．その各要素は，角度方向に区切られた細い扇型の中に見える物体の情報を表しており，最も近くにある餌の場合9，ゴミの場合 -9 となり，視野の最も遠くにある餌の場合1，ゴミの場合 -1 となる．この細い扇型の中に何も物体が無い場合は0となる．
- 出力データ集合 Σ_Y は

  $$\{-45, -30, -15, 0, 15, 30, 45\}$$

  とする．虫は1サイクルにあらかじめ決められた一定の距離だけ前進するが，その前に出力データにより決められる角度に従って向きを変える．角度の単位は度である．
- 強化入力は餌に触れたとき $+1$，ゴミに触れたとき -1 が与えられ，何も触れなければ0とする．

以上のような設定の下で異なる乱数系列について500サイクルずつのシミュレーションを100回行なった．図2に100回の実験から得られた学習曲線を示す．

縦軸は2節で定義した性能 P_t であり，曲線はその平均値および平均値±標準偏差の値の変化を表すものである．平均としては明らかに学習の効果が見られるが，標準偏差は一定して大きく，ほとんど学習の効果が見られない場合も観察された．乱数系列の違いは，初期の行動パターンを左右する．非常に有用な系列を早期に経験す

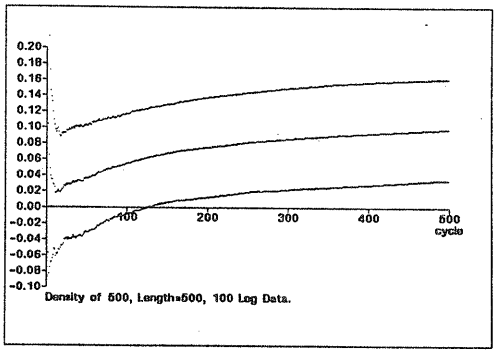


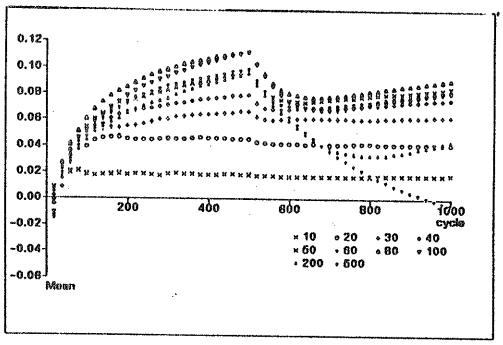
図 2: Learning curve of the basic algorithm IBRL0.

る場合もあれば，長期間に渡って学習の手がかりが得られない場合もある．さらに，このアルゴリズムでは，過去におけるすべての記憶が対等に検索対象となるため，稀にしか役に立たない過去の成功例に固執するような行動を取る可能性もある．上記のような学習性能の大きなばらつきは，これらの理由によるものと考えられる．

4 記憶容量の制限

IBRL0 では無限の記憶容量を仮定していたが，多くのサイクルに渡る実行に対しては，この仮定は計算量の上から考えると空間的にも時間的にも現実的でない．1つの単純な解決策は，記憶可能なサイクル数を制限することである．すなわち，記憶を環状の有限長のキューとし，最近の N サイクル分の経験のみを記憶する．N サイクル目以降は，N サイクル前の記憶を消去し，その記憶場所へ新たなデータを格納する．アルゴリズムとしては，t サイクル目のデータの格納場所を示すインデックスが t ではなく $(t \bmod N)+1$ となり，記憶に用いる配列の大きさが N となること以外，IBRL0 とまったく同じである．このアルゴリズムを **IBRL1** と呼ぶことにする．

実際，学習が行なわれれば，最近の行動ほど良い行動パターンを示すはずであるから，強化学習の枠組において，このような方略は悪くないと考えられる．また，学習中における環境あるいは入出力システムの変動に対しては，記憶容量を制限した方が柔軟に対応できるものと考えられる．基本アルゴリズムでは，過去におけるすべての記憶が対等に検索対象となるため，多くの場合に成功に導くような有用な記憶が保持される可能性が高まる反面，稀にしか役に立たない過去の成功例に固執するような行動を取る可能性もある．つまり，記憶容量をある程度制限した方が，平均としての性能はともかく，性能のばらつきを小さくするためには有益であると考えられる．しかし，記憶容量を制限し過ぎると，過去 N サイクル分の記憶中に意思決定に有用な記憶が含まれる確率が



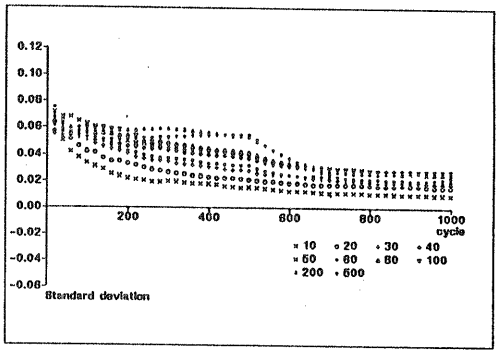


図 3: Learning curves of IBRL1.

低下するため性能を落すことになる．これらの性質はつぎに述べる実験によって確認された．

IBRL0 の場合とまったく同じ問題設定のもとで，いくつかの異なる記憶容量について同様のシミュレーションを行なった．ただし，環境変化に対する適応能力を調べるため，501 サイクル目以降，強化入力の値を反転させた．つまり，501 サイクル目以降は，餌を避け，ゴミを食べることが新たな目的となる．学習システムにとっては，入出力には変化がなく強化入力だけが反転することとなる．

図 3 に 1000 サイクル分の学習曲線を示す．上側の図は平均値の変化，下側の図は標準偏差の変化を表す．環境に変化が起きる 500 サイクル目までは，記憶容量を 50 程度に制限しても平均値としては容量を制限しない場合と同等の学習性能を示すが，更に容量を制限すると性能は低下する．しかし，下側の図を見ると，容量を制限した方が標準偏差が小さくなる傾向にあることがわかる．このことは，確実な学習性能を得るためには適度に記憶容量を制限した方がよいことを示している．

環境に変化が起きた直後は，当然ながら，それまでに経験に従って餌を食べ続けようとするため，性能 P_t は低下する．ところが，餌を食べると負の強化入力が得られるため，再びランダム探索行動を起こし，徐々に新た

な環境に適応していく．記憶容量が大きい場合には過去の経験の影響が長く残るため，新たな環境への適応は遅くなる．

5 フィードバック機構の導入

逆伝播学習[20]において学習システムが予測した結果と教師から教えられる正解との差を元に内部状態を変更するのと同じように，強化入力に基づいて出力決定に用いた記憶に何らかの修正を加える機構を導入すれば，学習機能を強化することがでよう．1つの方法は，記憶の信頼度を操作することである．すなわち，正の強化入力が得られたときには，そこで用いられた記憶の信頼度を高め，負の場合には弱める．ただし，強化入力には遅れがあり，意味のある強化入力の原因として過去の数サイクル分の出力が関与していると考えられるため，この信頼度変更の手続きは，強化入力の記録の手続きと同様，記憶を過去へ遡りながら変更を伝播させる必要がある．ただし，遠い過去まで遡るのは合理的でない．ここではつぎのようなアルゴリズムに従って，過去に行くほど変更の幅を小さくし，あらかじめ決められたサイクル数 m だけ遡ることとする．実験では $m = 50$ とした．

アルゴリズム 4

```
var RelMem: array [1..N] of real;
procedure Reflect(Rinfc);
for I in [1..m] do begin
   K := Index of OutData employed I cycles ago.;
   if Rinfc = +1 then
      RelMem[K] := RelMem[K]
         + ρ1 · ρ2^I · (1 − RelMem[K])
   else if Rinfc = −1 then
      RelMem[K] := RelMem[K]
         − ρ1 · ρ2^I · RelMem[K]
end
```

ここで ρ_1 および ρ_2 は $(0,1)$ の実数定数である．実験では $\rho_1 = 0.5, \rho_2 = 0.95$ とした．信頼度の初期値が $(0,1)$ の区間にあれば，値は絶えず 0 から 1 の間に保持される．信頼度は入出力データの記憶の時点で 0.5 に初期化する．手続き **Reflect** はアルゴリズム1 の中の，強化入力の読み込みの後に呼び出せば良い．

また，過去に採用された記憶のインデックス K を見つけるためには長さ m の環状のキュー LastQ を用意し，意思決定の時点でインデックスをキューに記録すれば良い．すなわち，アルゴリズム 3 に示した意思決定手続き **Policy** の中で，もし K が見つかれば K の値を，さもなければ K としてあり得ない値，たとえば -1 を LastQ$[(t \bmod m)+1]$ に代入する．こうしておけば，i サイクル前に採用された記憶のインデックスは LastQ$[((t+m-i) \bmod m)+1]$ を参照することによって得ることができる．

制限された記憶容量の下で，古い記憶から順に忘却するのではなく，信頼度の低い記憶から忘却するようにすれば，成功に結び付きやすい行動系列に含まれる記憶を残すことができ，より確実な学習性能が期待できる．また，4 節で述べたように，古い記憶を忘却することも有効であるから，これら 2 種類の基準を統合することが適切と考えられる．1つの実現法は古い記憶の信頼度を低くすることである．すなわち，サイクル毎に全記憶の信頼度を一定の割り合いで減衰させる．実験では減衰の係数 η の値を 0.98 とした．η と ρ_1 の値の比率によって，時間経過による効果と，フィードバックによる効果のバランスが決まる．

このような信頼度に基づく忘却のメカニズムを導入する場合には，過去に採用された記憶のインデックスを見つけるためのアルゴリズムに若干の工夫が必要となる．というのは，LastQ の中に登録された記憶が，参照しようとした時点で既に忘却されている可能性があるからである．LISP のような動的な記憶割り当てと自動ゴミ集め機能をもったプログラミング言語を用いれば，このような問題を回避することはさほど難しくはない．すなわち，個々の記憶要素を 5 つの要素，入力，出力，期待される補助入力，信頼度，忘却されたかどうかを示すフラグからなる構造体とし，記憶およびキューを，記憶要素を要素とする固定長のベクトルとする．忘却した時点でフラグをセットし，キューの中を見る場合には，フラグのセットされている記憶要素を無視すればよい．C や Pascal のような自動ゴミ集め機能をもたないプログラミング言語の場合にはつぎのような方法を用いるのが簡単であろう．すなわち，記憶と同じ大きさの整数配列を用意し，記憶した時刻 t を記憶と同じインデックスをもつ要素に代入する．キューに記録された記憶要素の記憶時刻はキューの順に整列しているはずであるから，キューを参照する時点で，記憶時刻を記録した配列の対応する要素を参照し，時刻が適合しないものは無視するようにする．実際に我々が実験で用いたプログラムでは，後者の手法をとっている．以上ようなフィードバック機構を導入したアルゴリズムを **IBRL2** と呼ぶこととする．

IBRL2 についても IBRL1 とまったく同じシミュレーション実験を行ない学習性能を確認した．

図 4 に学習曲線を示す．全体としては IBRL1 の場合と同様の変化を示すが，平均の評価は IBRL1 よりも高くなる傾向にある．特に記憶容量が 40 以下の場合にこの傾向は顕著であり，記憶容量が 10 の場合でも，IBRL1 の記憶容量 30 の場合と同等の能力を示す．記憶容量が大きな場合に IBRL1 との能力差が小さいのは，忘却の機会が少なく，フィードバックによる信頼性の調整が意思決定にほとんど影響しないためと考えられる．

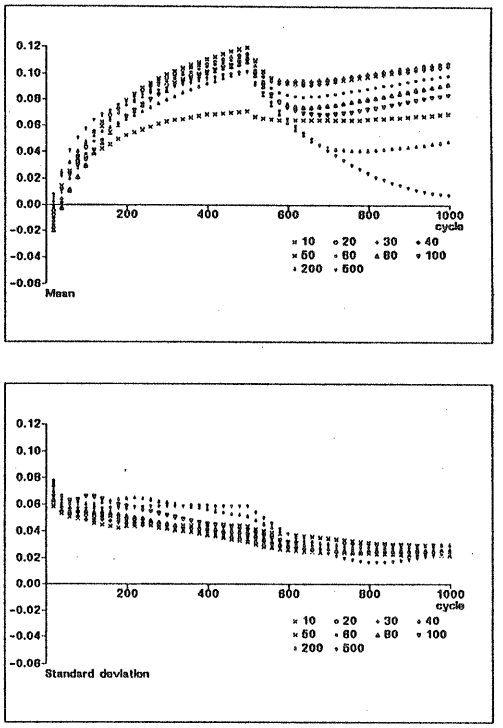


図 4: Learning curves of IBRL2.

6 考察

この節では考察として，上で提案した学習アルゴリズムの実行に要する計算量，パラメータ値の意味，および，適用可能な問題領域の拡張について述べる．

6.1 計算量

まず，基本アルゴリズム IBRL0 の実行に必要な計算量について考えてみよう．考慮すべき手続きは 2 つある．強化入力逆伝播手続きは，0 でない強化入力が得られたサイクルにおいて一定数 $k = \lceil \log_\gamma \nu \rceil$ 以下の記憶要素に変更を加える．実験で用いた値 $\gamma = 0.85, \nu = 0.05$ を当てはめると $k = 19$ である．意思決定手続き Policy は，すべての記憶要素について，現在の入力との類似度を計算し，その値と期待される強化入力の積が最大となるようなものを見つけ出す．すなわち，記憶された訓練例の数，すなわち，それまでの実行サイクル数 t と入力データの次元数 n に比例する計算コストが必要となる．0 でない強化入力が得られる頻度と逆伝播される要素数 k の積は訓練例の数に比べて非常に小さいと考えられるので，強化入力逆伝播手続きにおける計算コストは無視できる．結論として，基本アルゴリズムは 1 サイクル当たり $O(nt)$ の計算時間を必要とすることになる．組合せ的爆発を起こすことはないが，学習サイクルが進むに比例して 1 サイクル当たりの計算コストが大きくなるため，長期間に渡る実行には適さない．

表 1: CPU time spent by 100 trials of 1,000 cycles. (unit = second)

記憶容量	20	50	100	200	500
IBRL1	111.37	184.67	289.30	502.16	975.59
IBRL2	188.58	253.17	350.05	565.12	1025.39

記憶容量を制限する方法 IBRL1 では，容量に到達するまでのサイクルでは計算コストが学習サイクルが進むに比例して増大するが，それ以後は変化せず，記憶容量 N に比例した一定の計算時間となる．さらに，ベクトル演算機構をもつ SIMD 型の並列計算機上で実行すれば，類似度計算は各々の記憶要素に同じ演算をほどこすものであるから，演算装置の数に比例して計算時間を減らすことができ，最良の記憶要素を検索する箇所も，最大値を求めるアルゴリズムに帰着できるので，記憶容量に対して対数オーダーの計算コストに押えることができる[21]．

フィードバック機構を導入した IBRL2 の場合には，さらに，最小の信頼度をもつ記憶要素を見つけるための計算コストを考慮しなければならない．簡単にすべての記憶要素を調べる方法を取ると，類似度計算の場合と同様に記憶容量に比例した計算時間が必要となるが，ヒープソートアルゴリズムに用いられるような木構造を導入する[22] など，データ構造に工夫を加えるか，あるいは，並列計算機上で実行すれば，計算時間を対数オーダーに押えることができる．ただし，実験に用いたプログラムでは，アルゴリズムを簡単なものとするために，単純にすべての記憶要素の信頼度を調べる方法をとった．

実際に実験に要した CPU 時間は，表 1 のとおりである．

これは各々の記憶容量の下で，1000 サイクル分の実行を 100 回行なうためにかかった CPU 時間を秒で示したものである．使用した計算機は 20MHz の R3000 および R3010 を CPU にもつ SONY NWS-3460，プログラミング言語は C を用いた．上述のとおり，記憶容量に比例して多くの CPU 時間を要していることが分かる．また，1 サイクル当たりの平均 CPU 時間は記憶容量が 50 の場合で，IBRL1 では 1.85ms，IBRL2 では 2.53ms ほどであり，実時間性を要求される制御問題にも十分応用可能な速度であると考えられる．

6.2 パラメータ値の意味

上で提案したアルゴリズムには，設計者が設定するいくつかの実数パラメータが使われている．ここでは，そ

れらの値の意味について考察する.

IBRL0 および IBRL1 で用いられるパラメータは, γ と ν である. 学習性能にとっては, これらの個々の値ではなく, $k = \lceil \log_\gamma \nu \rceil$ が問題となる. 6.1節で述べたように, k は強化入力の逆伝播のステップ数である. すなわち, 強化入力が得られた時点から, 最大 k サイクル前の記憶要素までが, 逆伝播の影響を受け, それ以前の記憶は影響を受けない. もし k が, 強化入力を得るための原因となる行為のステップ数よりも小さければ, 学習能力は貧弱なものとなることが予想される. 逆に k が大き過ぎると, 正の強化入力を得る最短経路ではなく, たまたま発見した冗長な経路に行動パターンが収束する可能性が高くなると思われる. すなわち, k の値は, 正の強化入力を得てから, 次の正の強化入力を得るまでの標準的な最短サイクル数とするのが適当である. 実験で用いた問題におけるその値は 6 であるが, $k = 19$ を用いたため, 冗長な行動パターンに収束した場合がいくつか観察された.

IBRL2 では, 信頼性の変更にかかわるパラメータとして ρ_1, ρ_2, η が使われている. ρ_1 は, フィードバックによる影響力の大きさを, ρ_2 は, その際の時間的隔たりによる影響力の減衰係数を, η は, 時間経過による信頼性の減衰係数を, それぞれ表すものである. ρ_1 を 0 に近付けると, IBRL1 に似た挙動を示すようになる.

パラメータ値の変更に伴う学習能力変化の調査, およびパラメータ値の適応的変更の方法の開発は今後の課題である.

6.3 問題設定における制約の緩和

適用可能な問題の範囲を特定する, あるいは, 問題の性質と学習性能の関係を明らかにすることは, 学習アルゴリズムの研究にとって重要である. 適用可能な問題を特定するための要素は, 入出力と学習目標の特性である. 強化学習法の枠組では, 入出力は離散時系列であり, 学習目的は強化入力によって与えられる. ここで提案した学習アルゴリズムの適用可能問題を, この枠組の範囲内でさらに拡張する方向としてはつぎのようなものが考えられる.

1つは, 文脈依存性をもった入出力環境である. すなわち, 学習目的に沿った行為を決定するために, 2 以上の長さの入出力列をみなければならないような状況である. たとえば, 経済指標の予測問題を考えると, ある時点の経済指標だけを見たのでは予測は困難であり, ある程度の期間に渡る変動のパターンを観察する必要がある. つまり, 1 サイクル毎に類似データを検索するのではなく, データの系列の類似性を考慮しなければならない. 先に提案した学習法[11] で用いた想起表による類似データの管理法は, このような問題のための 1 つの解決策である. すなわち, 検索された類似データを想起表と呼ばれる記憶領域に登録し, 前後のデータとの類似性との加重和を逐次計算し, その記憶系列と現状との類似性とする方法である. もちろん, それまでの変動の履歴の情報が各時点の入力データとして与えられれば, 文脈依存性を改めて考慮する必要はない. しかし, 多くのサイクルにわたる類似度を各サイクル毎にまとめて計算するのでは明らかに効率が悪い. また, 前データとの差分などを入力に含める方法も考えられるが, 入力系列の特性についてある意味での制約を新たに加えることになる.

もう1つの拡張の方向は個々の入出力データの複雑化である. ここでは入力データとして実数ベクトルを仮定しているが, 文字列やフレームのような木構造をデータとして扱えるようにするためには, 事例ベース推論[23] の分野で研究されているような知識構造間の類似性を導入する必要がある. また, 本論文の問題設定は, 出力を既知の有限個のシンボルとしたことによって, 各サイクルの実行が分類問題のクラスに単純化されている. 出力データとして連続量を取る実数, あるいはそのベクトル, さらに, 文字列や知識構造などの構成的なデータが要求される場合には, 意思決定の過程で, 出力データを組み立てる仕掛けが必要となり, 問題は複雑になる.

強化入力に対する制約の緩和も重要である. アルゴリズム 3 では, 正の強化入力のみを参照しており, Barto らが対象としたような負の強化入力のみから学習するような問題[13] に対しては, そのまま適用することはできない. 実際に得られる強化入力の様々な分布に適応可能とするには, 入力データに対して採用したと同様の正規化を強化入力に対しても導入する必要がある. すなわち, アルゴリズム 3 の中で, 平均が 0, 最大値が 1 となるようななんらかの単調な変換を RifMem[K] に施した結果を用いればよいと考えられる.

7 関連する研究との比較

Moore のロボットマニピュレータ制御のための学習システム[10] は, ここで提案した学習アルゴリズムと同様に, 実例に基づく学習の枠組を, 離散時系列の制御問題に応用したものと見ることができる. Moore のシステムでは, 感覚 (sense), 行為 (action), 行動 (behavior) の 3 つ組からなる記憶要素を用い, 目標とする行動に合うような行為を, 最も近い既知データ (nearest neighbor) を検索することで決定するメカニズムを取っている. 目標行動は各ステップにおいて明示的に与えられる. この問題設定は, 目標行動と実際になされた行動との差を強化入力とみなすことによって, 強化学習法の枠組みで捉えることもできるが, 強化入力には遅れはなく, この意味では教師付の分類問題として捉える方が適切である. また, Moore は, 雑音や環境変化に対する頑健性をもたせるため, 近傍のデータの間で, 行動のデータを平滑化する手法を導入している. 平滑化によって求めら

れた値と観測結果が大きく異なるような記憶，および古い記憶は適当な閾値に基づいて削除される．この方法は，IBRL2 における信頼度配分の方法と同じような効果をもたらすものであるが，Moore の方法では信頼度のような数値情報は使われていない．さらに，最も近いデータまでの距離が閾値より遠い場合には，成功する確率の高い出力値を計算するメカニズムに従って探査行動をとる．これは本研究をはじめ多くの強化学習法で取られるようなランダムな選択よりももっともらしい探査行動をとるための１つの工夫である．探査行動の制御は強化学習法における重要な問題点の１つである．

Holland のバケツリレーアルゴリズム[24]では，ルールの連鎖によってタスクが成功した場合に，IBRL0 と同様の逆伝播メカニズムを使って信頼度の配分を行なっている．ただし，失敗に対してはそのような逆伝播を用いずに，活性化した時点で信頼度を少し減少させることによって同様の効果を実現している．これは，複数のルール系列が同時に活性化することを前提にしているためであり，競争に負けたルール系列は，特に逆伝播を行なわなくとも結果的に信頼度が下がるよう仕組まれている．このようなシステムにおいて，負の信頼度の逆伝播を明示的に用いると，失敗したすべてのルール系列に対して逆伝播を開始する必要が生じるため，計算効率の上から，あるいは認知モデルとしての妥当性の上から考えて適切でない．しかし，本システムでは活性化は即実行となるため，一度に活性化する記憶は唯一であり，負の信頼度の明示的な逆伝播を妨げる理由はない．

また，Holland は分類子 (classifier) システムの解説のために，虫を食べる動物の学習を例としてルールの変更過程を説明しているが，基本となる単純な規則があらかじめ与えられた状況から出発しており，我々が提案した学習アルゴリズムとは，前提条件を異にするものである．しかし，分類子システムに組み込まれている知識の構造化のメカニズムは，本システムの拡張を考える上でも重要な示唆を多く含むものである．

Sutton はこれまでに開発されてきた強化学習法の枠組を４つの段階に整理した[17]．それによれば，最も進歩した第４段階の Q-Learning と呼ばれる強化学習法では，入出力データをもとに環境に対する予測を行ない，その結果を利用して意思決定する．さらに予測の正誤をフィードバックさせ，予測能力を向上させる．Sutton の分類は，ニューラルネットのようなパラメータ調整によるある意味での環境のモデルを同定するシステムを想定したものであるが，本システムをその枠組みの上であえて位置付けるならば，Q-Learning に相当するものとみなすことができる．すなわち，強化入力の逆伝播によって割り当てられた期待値は，まさに入出力データをもとにした環境予測に対応し，信頼度の割り当ては，予測結果の正誤に基づくフィードバックに対応する．さらなる拡張として Sutton は，環境のモデルを利用してプランニングを行なう Dyna と呼ばれる学習機構を提案している．我々が先に提案した記憶に基づく時系列学習システムにおいても同様の考え方に沿って，「計画」の概念を導入した．出力系列の組み立てが必要な場合には，このような拡張が必要であると考えられる．

本研究と他の強化学習法との問題設定上の違いは，１つには 6.3節の最後に述べたとおり，強化入力値の分布に対する適応性を欠いていることである．もう１つは入力がビットベクトルではなく実数ベクトルであることである．上で提案したアルゴリズムは，そのままビットベクトルを入力とする場合にも適用可能である．すなわち，本研究で提案したアルゴリズムは他の強化学習法に比べて，強化入力への適応性は劣るが，入力の多様さの点では優れているといえる．

8 おわりに

記憶された実例をもとにした，離散時系列を対象領域とする制御問題に適用可能な学習アルゴリズムを提案した．本システムのように仮説としての規則を生成しないメカニズムは，一見，効率が悪いように思えるが，上で示したとおり，実時間性を要求される分野にも十分応用可能な時間で実行できる．さらに，線形システムのような対象の特性に対するモデルを特に想定しないため，単純なアルゴリズムであるにもかかわらず，より複雑な環境への適応が期待できる．

上で述べたような文脈に依存した環境への適応のメカニズムや，計画の導入によっておこる学習性能の変化を調査することは，今後の課題である．

謝辞

本論文の執筆に際して有益な助言を頂いた，東京工業大学の小林重信教授，北海道大学の赤間清助教授，九州工業大学の黒木秀一講師，ならびに，シミュレーション実験のための計算機設備について便宜をはかって頂いた長岡技術科学大学の吉谷豊教授，小池昭彦君に感謝いたします．

参考文献

[1] Stanfill, C. and D. Waltz: Toward Memory-Based Reasoning, *Communications of the ACM*, Vol. 29, pp. 1213–1228, (1986).

[2] 佐藤理史: 記憶に基づく翻訳 II, 情報処理学会人工知能研究会報告 70-3, (1990).

[3] Aha, D. W., D. Kibler and M. K. Albert: Instance-Based Learning Algorithm, *Machine Learning*, Vol. 6, pp. 37–66, (1991).

[4] Waltz, D. L.: Applications of the Connection Machine, *IEEE Computer*, Vol. 20, pp. 85–97, (1987).

[5] Angluin, D. and C. H. Smith: Inductive Inference: Theory and Methods, *Computing Surveys*, Vol. 15, pp. 237–269, (1983).

[6] Michalski, R. S.: A Theory and Methodology of Inductive Learning, in Michalski, R. S., J. G. Carbonell and T. M. Mitchell (Eds.), *Machine Learning: An artificial intelligence approach*, Palo Alto, CA: Tioga, (1983).

[7] 佐藤理史: Memory-based アプローチと規則学習, 「学習のパラダイムとその応用」シンポジウム論文集, 情報処理学会, pp 69–77, (1989).

[8] Samuel, A. L.: Some Studies in Machine Learning Using the Game of Checkers, *IBM Journal on Research and Development*, Vol. 3, pp. 210–229, (1959).

[9] Mason, M. T., A. D. Christiansen and T. M. Mitchell: Experiments in Robot Learning, *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 141–150, (1989).

[10] Moore, A. W.: Acquisition of Dynamic Control Knowledge for a Robotic Manipulator, *Proceedings of the Seventh International Conference on Machine Learning*, pp. 244–252, (1990).

[11] 畝見達夫: 記憶に基づく離散時系列の学習と環境適応シミュレーションへの応用, *Workshop on Learning 1991*, 北海道札幌市手稲, (1991).

[12] Waltz, M. D. and K. S. Fu: A Heuristic Approach to Reinforcement Learning Control Systems, *IEEE Transactions on Automatic Control*, Vol. 10, No. 4, pp. 390–398, (1965).

[13] Barto, A. G., R. S. Sutton and C. W. Anderson: Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 13, No. 5, pp. 834–846, (1983).

[14] Lynne, K. J.: Competitive Reinforcement Learning, *Proceedings of the Fifth International Machine Learning Conference*, pp. 188–199, (1988).

[15] Kaelbling, L. P.: Learning Functions in k-DNF from Reinforcement, *Proceedings of the Seventh International Conference on Machine Learning*, pp. 162–169, (1990).

[16] Whitehead, S. D. and D. H. Ballard: Active Perception and Reinforcement Learning, *Proceedings of the Seventh International Conference on Machine Learning*, pp. 179–188, (1990).

[17] Sutton, R. S.: Reinforcement Learning Architectures for Animats, in J.-A. Meyer and S. W. Wilson (Eds.), *From Animals to Animats – Proceedings of the First International Conference on Simulation of Adaptive Behavior*, The MIT Press, pp 288–296, (1990).

[18] Beer, R. D.: Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology, Academic Press (1990).

[19] 赤間清: 知識はいかに獲得され得るか, 認知科学の発展, Vol. 1, pp. 161–188, (1988).

[20] Rumelhart, D. E., G. E. Hinton and R. J. Williams: Learning Internal Representation by Error Propagation, in J. McClelland, D. E. Rumelhart and PDP Research Group: *Parallel Distributed Processing: Vol. 1 Foundations*, MIT Press, pp 318–362, (1986).

[21] Hillis, W. D. and G. L. Steele, Jr.: Data Parallel Algorithms, *Communications of the ACM*, Vol. 29, pp. 1170–1183, (1986).

[22] 小倉宏明, 畝見達夫: 暗記学習のための記憶管理方式について, 情報処理学会人工知能研究会報告 72-3, (1990).

[23] Riesbeck, C. K. and R.C. Schank: Inside Case-Based Reasoning, Hillsdale, NJ: Lawrence Erlbaum Associates, (1989).

[24] Holland, J. H., K. J. Holyoak, R. E. Nisbett and P. R. Thagard: Induction, MIT Press, (1986).