Vol.2013-MPS-95 No.4
2013/9/26

Compact Genetic Algorithmを導入した学習分類子システムによる分類子数の削減

中田　雅也[1,2]　ピエール・ルカ・ランチ[3]　田島　友祐[1]　高玉　圭樹[1]

概要：本論文では，学習分類子システム (Learning Classifier System: LCS) において，学習する分類子数を削減するために，Compact Genetic Algorithm を用いた確率モデル型分類子生成法を提案する．提案分類子生成法は，1) 分類子がもつ部分解の存在確率をモデル化することで不要な分類子の生成を抑制し，2) 従来の確率モデル型分類子生成法が適用困難であった強化学習問題クラスに適用可能である．教師あり学習問題 (multiplexer 問題) と強化学習問題 (block world 問題) において，提案分類子生成法を導入した LCS を適用したところ，次の知見を得た．まず，1) 提案 LCS は従来 LCS よりも，少ない学習回数で最適解を学習可能であり，2) 従来 LCS が学習した分類子数に対し，提案 LCS は最小でも 54%(最大で 76%) 削減した分類子数で学習可能であることを示した．

1. はじめに

学習分類子システム (Learning Classifier System: LCS)[10] は，機械学習と進化計算を組み合わせた環境適応システムであり，教師あり学習 (Supervised Learning) や強化学習 (Reinforcement Learning: RL)[14] が扱う幅広い機械学習問題に適用可能である [1], [11]．ここで，LCS における知識獲得技術は一般化 (generalization) と呼ばれ，複数の環境状態に照合する汎用的な分類子を学習することで実現される．現在主流の LCS は，LCS を構成する 2 種類の機械学習手法 (教師あり学習と強化学習) に応じて分類される．まず，教師あり学習を用いた LCS である Supervised Learning-based LCS(UCS)[2] は，環境適応に最低限必要な分類子のみ学習するため，教師あり学習問題においては少ない分類子数で学習可能である [2]．しかし，UCS は教師データが設計不可能な未知環境を扱う強化学習問題に適用できない．一方で，強化学習を用いた LCS である Accuracy-based LCS(XCS)[7], [18] は，LCS が扱う全問題クラス (教師あり学習問題と強化学習問題) に適用可能である．しかし，XCS は全状態行動空間を網羅的に学習するため [7]，膨大な分類子数を必要とする問題がある．

本論文では，上記の従来 LCS の限界を克服するために，少ない分類子数で全問題クラスに適用可能な LCS の構築を目指す．このために，本論文は XCS 内の分類子生成法 (Rule-discovery mechanism) に着目し，不要な分類子の生成を抑制することで，XCS における分類子数を削減することを考える．ここで，分類子生成法は進化計算によって実現され，一般的に遺伝的アルゴリズム (Genetic Algorithm: GA)[8] が用いられる．特に，XCS では，学習不十分な分類子が削除されることを防ぐため，毎世代に 2 つの子個体 (分類子) のみ生成 (2 個体のみ削除) する定常状態 GA(Steady-State GA)[17] が用いられる [7]．しかし，定常状態 GA は，毎世代に 2 つの親個体のみ考慮して子個体を生成するため，正しい分類子を獲得するまでに，多様な分類子を生成する．その結果，XCS は，全状態行動空間における最適解を獲得するために，膨大な分類子数を生成する必要がある．

そこで，我々は，分布推定アルゴリズム (Estimation of Distribution Algorithm: EDA)[12] の一手法である Compact Genetic Algorithm (cGA)[9] を用いた確率モデル型分類子生成法を提案する．提案手法は，親個体がもつ部分解の存在確率を確率ベクトルでモデル化することで，有望な部分解を持たない分類子 (不要な分類子) の生成を抑制可能とする．したがって，提案手法は，複数の親個体を考慮して子個体を生成する点で定常状態 GA と異なる．加えて，提案手法は，XCS のための分類子生成法であるため，従来 LCS(Compact Classifier System:CCS[13], Attribute-Feedback UCS:AF-UCS[16]) の確率モデル型分

[1] 電気通信大学大学院
Graduate School, the University of Electro-Communications, Chofu, Tokyo, Japan
[2] 日本学術振興会　特別研究員 (DC1)
Research Fellow of Japan Society for the Promotion of Science(DC1)
[3] ミラノ工科大学
Politecnico di Milano, Milano, Italy

ⓒ 2013 Information Processing Society of Japan

類子生成法が適用困難であった強化学習問題に適用可能である．具体的には，CCS や AF-UCS の分類子生成法は，評価関数もしくは教師データを用いて確率モデルを生成するが，提案手法では，教師データ等を用いずに確率モデルを生成するため強化学習問題に適用可能である．

本論文の構成は次の通りである．以下，第 2, 3 章では，XCS と cGA のメカニズムについて述べ，第 4 章で提案手法について説明する．第 5，6 章では，multiplexer 問題と block world 問題における実験結果を示す．最後に，第 7 章で本論文をまとめる．

2. Accuracy-based LCS (XCS)

2.1 分類子

XCS で用いられる分類子 (classifier) は IF(条件)-THEN(行動) ルールで表現され，環境状態と照合する条件部 (condition : C) と，照合後に実行する行動部 (action : A) からなる．一般的に条件部と行動部は 0,1 のビット列で形成されるが，特に条件部は任意の値を示す#(don't care) を組み込むことで，複数の環境状態に適応可能な汎用的な条件部が形成される．各分類子は予測値 (prediction)，誤差値 (error) および適応度 (fitness) の評価値を持ち，条件部と行動部が共に等しい分類子をマクロ分類子 (macro-classifier) として集約し，集約した数を重合度 (numerosity) と呼ぶ．分類子の上限数 (max population size) をパラメータ N と定め，j 番目の分類子 cl_j の予測値，誤差値，適応度，重合度をそれぞれ p_j, ϵ_j, F_j，num_j と記す．

2.2 メカニズム

環境より入力される状態 (state) は，一般的に 0,1 のビット列からなり，$[P]$ 内の各分類子の条件部と照合され，照合した分類子は照合集合 (match set : $[M]$) を形成する．ここで，$[M]$ 内に存在する分類子がもつ行動部の種類の数がパラメータ θ_{nma} の値未満の場合，条件部が入力状態に照合し，かつ，$[M]$ 内に存在しない行動をもつ分類子を θ_{nma} の値以上に上回るまで生成する．ここで，被覆により生成された分類子の条件部は，入力状態と同一に設定されるが，各ビットに対して確率 $P_\#$ で#に置き換えられる．次に，式 (1) に従って各行動の予測報酬を計算し予測配列 (prediction array : $P(a)$) に記憶後，これを選択確率として行動選択をおこなう．ただし，式中の $[M]\,|_{a_i}$ は，$[M]$ 内で行動部に行動 a_i をもつ分類子の集合である．最後に，照合集合から選択された行動を行動部に持つ分類子をまとめて行動集合 (action set : $[A]$) を形成し，選択された行動が実行される．

$$P(a_i) = \frac{\sum_{cl_k \in [M]|a_i} p_k \times F_k}{\sum_{cl_k \in [M]|a_i} F_k} \tag{1}$$

環境から得られた報酬 r に基づいて $[A]$ 内の各分類子の予測値，誤差値および適応度を式 (2)，(3) のように更新する．

$$p_j \leftarrow p_j + \beta(r + \gamma\, max\, P(a) - p_j) \tag{2}$$

$$\epsilon_j \leftarrow \epsilon_j + \beta(\mid P - p_j \mid - \epsilon_j) \tag{3}$$

ここで，予測値は，前ステップの報酬 r と予測配列中の最大予測報酬 $maxP(a)$ を用いて計算される．パラメータ $\beta(0 \leq \beta \leq 1)$，$\gamma(0 \leq \gamma \leq 1)$ はそれぞれ学習率 (learning rate)，割引率 (discount factor) と呼ばれ，それぞれ学習の更新速度と将来の報酬を考慮する度合いを制御する．そして，更新後の誤差値 ϵ_j に基づき分類子の絶対的な正確さ κ_j と相対的な正確さ κ'_j を算出する [18]．最後に適応度 F_j が式 (5) に示すように更新され，強化部における一連の処理が完了する．ここで，ϵ_0 および ν は誤差 ϵ_j から絶対的な正確さ κ_j を計算する際のパラメータであり問題に応じて適切に設定する．

$$\kappa_j = \begin{cases} 1 & if \epsilon_j \leq \epsilon_0 \\ \alpha(\epsilon_j/\epsilon_0)^{-\nu} & otherwise. \end{cases} \tag{4}$$

$$F_j \leftarrow F_j + \beta(\kappa'_j - F_j) \tag{5}$$

定常状態 GA による分類子の生成と削除を通して，分類子集団 $[P]$ を進化させる．したがって，XCS で用いる分類子生成法は，定常状態 GA を用いた方法である．具体的には，前ステップ行動集合 $[\mathrm{A}]_{-1}$ を形成する各分類子が，それぞれ前回 GA の進化対象に設定された時から経過したステップ数の平均値がパラメータ θ_{GA} の値を上回る場合に GA が実行される．まず，分類子生成法が適用される．具体的には，$[\mathrm{A}]_{-1}$ 内の各分類子の適応度を用いたトーナメント選択 (トーナメントサイズ τ)[4] より 2 つの分類子が親個体として選択される．次に，子個体として，それぞれの親個体の分類子と同様の条件部，行動部および各評価値を持つ分類子が 2 つ生成され，交叉 (crossover) および突然変異 (mutation) がそれぞれパラメータ χ，μ の確率で適用される．特に，XCS で用いられる突然変異 (niche mutation) [7] では，分類子の条件部を#もしくは入力状態の属性値に変異させる．最後に，子個体である 2 つの分類子は分類子集団に追加される．このとき，$[P]$ 内の分類子数がパラメータ N を超えた場合，適応度の低い分類子が削除 (deletion) され，N を超えないときは削除されない．

3. Compact Genetic Algorithm (cGA)

cGA はビットストリング型の確率モデルを用いた GA である．cGA では，0,1 のバイナリで表現される環境状態をもつ環境を対象とし，GA における母集団の代わりに確率モデルを用いる．cGA における確率モデルは，遺伝子長と同じ長さの確率ベクトルを用い，環境内の最適解を確率ベクトルで表現する．ここで，各確率は対応する遺伝子座が 1 である確率を意味する．

ⓒ 2013 Information Processing Society of Japan

まず，確率ベクトルにおける各確率は 0.5 として初期設定される．そして，確率ベクトルから子個体を生成するが，子個体の各遺伝子座の属性値 (0 もしくは 1) は対応する確率より決定される．cGA では，2 つの子個体を生成後，評価関数より子個体を評価し適応度 (Fitness) を得る．その後，適応度が高い子個体を winner，低い子個体を loser と定め，両者の各遺伝子座の属性値の違いを用いて，確率ベクトルを更新する．具体的には，各遺伝子座において，winner と loser が異なる属性を持つ場合のみ，対応する確率を更新する．winner が 1 で loser が 0 である場合，確率の値を更新率 $(1/n)$ だけ増加させる．一方で，winner が 0 で loser が 1 である場合，確率の値を $1/n$ だけ減少させる．なお，同一の属性値を持つ場合，対応する確率は更新されない．ここで，n は任意に設定可能である

4. 提案手法

提案手法は，cGA を基にした確率モデル型分類子生成法であり，子個体を確率ベクトルより生成する．提案手法の特徴は，1)XCS の分類子生成法 (定常状態 GA) と比較して，複数の親個体から子個体を生成すること，2) 従来の確率モデル型分類子生成法と比較して，強化学習問題に適用できることである．cGA との差異は，1) winer と loser を判別せずに 2 つ以上の個体から確率を計算すること，2) 確率ベクトルを保有ならびに更新せず毎世代確率ベクトルを構築すること，さらに，3) 生成した子個体に対し，確率ベクトル用いた突然変異を適用することである．そして，XCS で用いる GA を用いた分類子生成法との差異は，毎世代に複数の分類子を考慮して，子個体を生成することである．このため，提案手法は，a) 複数個体からの確率ベクトルの生成 (上記 1 と 2 に相当)，b) 確率ベクトルに基づく突然変異 (上記 3 に相当) の 2 つのメカニズムにより構成される．以下，各メカニズムについて説明する．

4.1 複数個体からの確率ベクトルの生成

4.1.1 概要

cGA は，子個体の正しい適応度が評価関数より獲得できることを前提としてる．そのため，cGA は 2 つの個体を比較することで，winner と loser に正しく判別可能であり，正しい部分解 (属性値) を識別できる．しかしながら，提案手法は，強化学習問題に適用するために，評価関数や教師データを用いずに正しい部分解を識別することが求められる．しかし，XCS より学習した分類子の適応度は，学習途中では必ずしも正しい値に収束しているとは限らない．これは，XCS に導入する提案手法では，winner と loser が一意に決定できないことを意味する．そのため，提案手法では，複数の分類子に存在する部分解の存在確率を算出し，存在確率の高い部分解を有望な部分解として扱う．

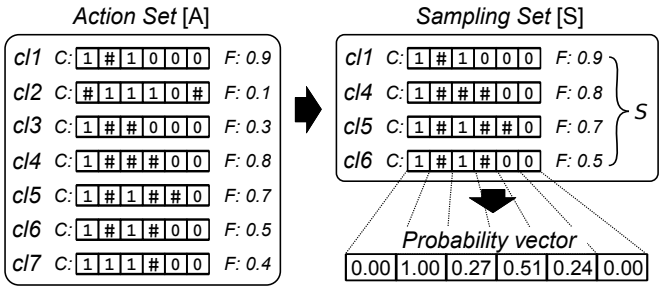

図 1 複数個体からの確率ベクトルの生成の概要図

4.1.2 メカニズム

図 1 に複数個体から確率ベクトルを生成する過程の概要図を示す．提案手法は，XCS と同様に行動集合 $[A]$ 内の分類子を基に分類子を生成する．一般的に，適応度 F が高い分類子は有望な部分解を持つ可能性が高いが，$[A]$ には適応度が低い分類子が多数存在している．提案手法では，適応度が低い分類子がもつ誤った部分解を抽出することを防ぐため，高い適応度をもつ分類子から構成される標本集合 (sampling set)$[S]$ を生成する．

具体的には，$[A]$ からトーナメント選択 [4] により選択された高い適合度をもつ分類子 (親個体) を，$[S]$ に格納する．このとき，選択する親個体の数を，$[A]$ 内の分類子数の sr (sampling rate) %と定める．ここで，sr は選択する親個体の数を調整するパラメータであり，$0 < sr \leq 1.0$ (100%) までの任意の値に設定する．例えば，sr=0.3 に設定した場合，分類子数が 10 である $[A]$ から 3 (=10×0.3) つの分類子を親個体として選択し，$[S]$ に格納する．ただし，親個体は既に選択した親個体を除く分類子から選択する．

$[S]$ を生成後，$[S]$ 内の分類子から確率ベクトルを計算する．提案手法では 0,1 のバイナリで表現される環境を扱うが，各確率は対応する遺伝子座が#(generalized bit) もしくは 0,1(specified bit) であるかを表現する．具体的には，各確率は#である確率を意味する．図 2 に，具体的な確率ベクトルの算出アルゴリズムを示す．i 番目の遺伝子が#である確率 $prob_i$ は，$[S]$ 内の分類子の条件部 C から対応する遺伝子座 C_i から算出される．このとき，各分類子の適応度 F の重みをつけて計算する．例えば，図 1 において，3 番目の遺伝子座は，cl4 のみが generalized bit (#) である．そのため，3 番目の確率は 0.27 $(= 0.8/(0.9+0.8+0.7+0.5))$ と算出される．

その後，cGA と同様に確率ベクトルから 2 つの子個体を生成し，分類子集団 $[P]$ に追加する．ここで，子個体の各遺伝子座の属性値は，対応する確率により generalized bit(#) もしくは specified bit(0,1) に設定される．specified bit となった場合は，入力状態 σ の対応する属性値 σ_i が設定される．ここで，$[P]$ 内の分類子数がパラメータ N を超えた場合，適合度の低い分類子が削除 (deletion) され，N を超えないときは削除されない．

```
1: for all classifier cl in [S] do
2:   for i = 0 to Length of C of cl do
3:     if C_i is equal to # then
4:       prob_i += cl.F
5:     end if
6:   end for
7:   FitnessSum += cl.F
8: end for
9: for i = 0 to Length of C do
10:   prob_i /= FitnessSum
11: end for
```

図 2 確率ベクトルの算出アルゴリズム

4.2 確率ベクトルを用いた突然変異

4.2.1 概要

学習途中では分類子は正しい部分解を持っておらず，確率ベクトル内の各確率は十分に収束しない．そのため，生成される子個体の条件部は，確率ベクトルが示唆する有望な部分解 (属性値) が反映されていないことが考えられる．そこで提案手法では，確率ベクトルが示唆する部分解を子個体に反映させるために，確率ベクトルに基づく突然変異を導入する．具体的には，提案する突然変異は，各遺伝子座の属性値が確率ベクトルの示唆する属性値と異なる場合，示唆された属性値へ変異させることで，分類子の条件部を変異させる．

4.2.2 メカニズム

図 3 に提案する確率ベクトルに基づく突然変異のアルゴリズムを示す．提案する突然変異は, AF-UCS で用いられる突然変異 [15] と同様に, 1) 通常の突然変異 (niche mutation) と 2) 確率ベクトルに基づく突然変異について，確率 P_C で適用する突然変異を決定する [15]．例えば，$P_{Complete}$=0.0 である場合，確率ベクトルに基づく突然変異が常に実行される．実行する突然変異を決定後，分類子の条件部の各遺伝子座について，突然変異確率 μ で突然変異が実行される．AF-UCS は，確率ベクトル内の確率の最大値を考慮して分類子を変異させるが，提案手法は，確率ベクトル中の対応する確率 $prob_i$ に応じて，遺伝子座の属性値を#もしくは対応する入力状態の属性値 σ_i に変異させる．

```
1: for i = 0 to Length of C do
2:   if RandomNumber ≤ P_C then
3:     Run niche mutation
4:   else
5:     if RandomNumber ≤ μ then
6:       if RandomNumber ≤ prob_i in vector then
7:         C_i ← #
8:       else
9:         C_i ← σ_i
10:      end if
11:    end if
12:  end if
13: end for
```

図 3 確率ベクトルに基づく突然変異のアルゴリズム

5. 教師あり学習問題における実験

教師あり学習問題における LCS の一般的なベンチマーク問題として，Multiplexer 問題 [18] を用いる．提案手法を導入した XCS(XCScGA) と従来手法である XCS を Multiplexer 問題に適用し，後述する評価基準を比較することで，提案手法の有効性を評価する．

5.1 Multiplexer 問題

Multiplexer 問題は，入力と出力との間の高い非線形性と，入力の適切な一般化により入出力関係を縮約できる構造を持つため，LCS の一般化能力の評価に用いられる．multiplexer 問題で与えられる入力であるビット長 l は $k+2^k$ であり，k は任意の整数である．また，出力はビット列 $b_0b_1\cdots b_{l-1}$ の最初から k ビット ($b_0 \sim b_{k-1}$) を 10 進数に変換した値 d を加えた b_{k+d} ビットの値となる．例えば，$k=2$ とした時に与えられる入力 110001 のアドレスは b_0 $b_1 \rightarrow 11$ を変換した値 $d=3$ と定まり，b_{2+3} のビット値である 1 が出力となる．

ここで，教師あり学習問題として扱う Multiplexer 問題は，教師データとして入力に対して正しい出力を LCS に与えることが可能であるが，XCS では教師データを扱う機構を有していないため，XCS が実行した行動を報酬の有無により評価する [18]．具体的には，任意の入力状態 (入力) に対して正しい行動 (出力) を実行した場合は報酬として $r=1000$ を与え，間違った場合は 0 を与える．また，本論文では k=5 とした 37-Multiplexer 問題を用いる．

5.1.1 評価基準とパラメータ設定

評価基準は，1) 正答率 (Performance)，2) 分類子集団 [P] 中の分類子数 (Population size) を用いる．正答率は高い値であるほど，正確に一般化された分類子を学習できていることを示している．分類子数は少ない値であるほど，分類子集団中に不要な分類子が少ないことを意味する．学習回数は 1 試行につき学習回数を 500,000 回とする．このとき，各評価基準は学習回数 5000 回ごとの移動平均で示し，試行数 10 回の平均をとる．パラメータ設定は先行研究と同様に設定する [6]．具体的には，N=5000, ν=5, β=0.2, ϵ_0=10, θ_{nma}=2, θ_{GA}=25, χ=0.8, μ=0.04, $P_{\#}$=0.65, $\tau=0.4$ とする．XCScGA は，sr=0.2 (20%)，P_C=0.2 と設定し，その他のパラメータは XCS と同様に設定する．

5.2 実験結果

図 4, 5 に, 37-Multiplexer 問題における XCS と XCScGA の正答率ならびに [P] 内の分類子数を示す．

図より，XCS は学習回数 350,000 回程度で正答率が 1 に達しているが，XCScGA は 100,000 回程度で正答率が 1 に達しており，極めて少ない学習回数で正しい分類子を学習

ⓒ 2013 Information Processing Society of Japan

できている．XCS の分類子数については，学習回数 10,000 回程度で最大値となり，学習回数の経過に伴って次第に減少している．一方で，XCScGA は，学習回数 10,000 ～ 35,000 回の間は XCS よりも大きい値であるが，その後急激に分類子数が減少している．これは，XCScGA が，学習初期では XCS よりも多様な分類子を生成し大域的探索を行った後，[P] 内の不要な分類子の数が減少していること意味する．また，学習終了後の XCS の分類子数は 1125 であるの対し，XCScGA の分類子数は 267 であることから，XCScGA は XCS の分類子数を約 76.3%削減している．

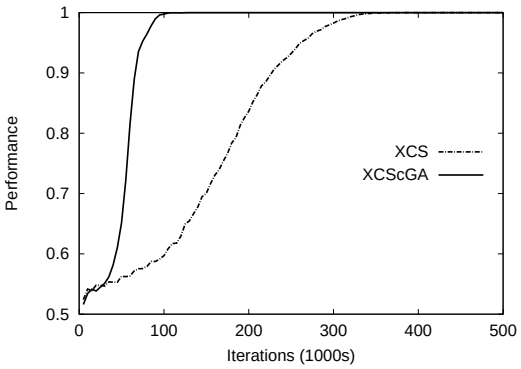

図 4 37-Multiplexer 問題における XCS と XCScGA の正答率

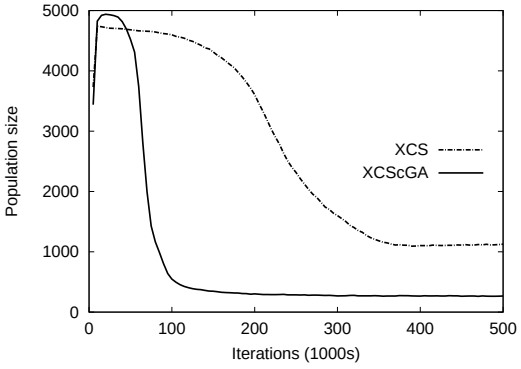

図 5 37-Multiplexer 問題における XCS と XCScGA の分類子数

6. 強化学習問題における実験

強化学習問題における LCS のベンチマーク問題として，Block world 問題 [18] を用いる．提案手法を導入した XCS(XCScGA) と XCS を Block world 問題に適用し，後述する評価基準を用いて提案手法の有効性を評価する．

6.1 Block world 問題

Block World 問題は，*animat* と呼ばれるエージェントが各セル間を遷移し食料 (報酬) の獲得を目的とする問題である．フィールドは障害物 (Obstacle：”T”)，通路 (Empty position：” ”)，食料 (Food：”F”) で構成される．また，取りうる行動の種類は 8 種類であり，現在位置から 8 近傍の各状態へ移動できる．まず，エージェントはフィールド内の通路にランダムに配置される．障害物の方向へ移動する場合は，通行不可として移動前の状態に留まる．エージェントが食料に達した場合，報酬 r=1000 を獲得し探索が終了する．ただし探索ステップ数が最大ステップ数 $maxstep$ を越えた場合も探索を終了するが報酬は与えられない．なお，本論文では図 6 に示すフィールド (maze6) を用いる．また，maze6 における最短平均経路 (optimum step) は 5.19 である [5].

T	T	T	T	T	T	T	T	T
T						T	F	T
T			T		T	T		T
T		T						T
T				T	T			T
T		T		T			T	T
T		T						T
T						T		T
T	T	T	T	T	T	T	T	T

図 6 フィールド (maze6)

6.2 評価基準とパラメータ設定

評価基準は，1) 食料までのステップ数 (Performance)，2) 分類子集団 [P] 内の分類子数 (Population size) を用いる．食料までのステップ数は小さく最短平均経路に近いほど，正確に一般化された分類子を学習できていることを示している．[P] 内の分類子数は少ない値であるほど，分類子集団中に不要な分類子が少ないことを意味する．学習回数は 1 試行につき, maze5 ならびに maze6 ともに 3,000 回行う．このとき，各評価基準は学習回数 50 回ごとの移動平均で示し，試行数 10 回の平均をとる．パラメータ設定は先行研究と同様に設定する [5]．具体的には，maze6 における XCS のパラメータ設定は，N=3000, ν=5，β=0.2，ϵ_0=1，θ_{nma}=8，θ_{GA}=100，χ=0.8，μ=0.01，$P_{\#}$=0.3, $\tau = 0.4$ とするが，$maxstep$=50 と設定する [3]．XCScGA は，sr=0.2 (20%)，P_C=0.2 と設定し，その他のパラメータは XCS と同様に設定する．

6.3 実験結果

図 7，8 に，Maze6 における XCS と XCScGA の食料までのステップ数ならびに [P] 内の分類子数を示す．

図より, Maze5 の実験結果と同様に, XCS と XCScGA のステップ数は, 同程度の学習回数で最短平均経路 (optimum step) に収束していることから，正しい分類子を適切に学習していることがわかる．XCS の分類子数は，学習回数が経過しても分類子数が増加している．一方で，XCScGA の分類子数は，学習回数 250~500 回の間は XCS よりも大きい値であるが, その後分類子数が減少している．また, 学習終了後の XCS の分類子数は 1288 であるのに対し，XCScGA

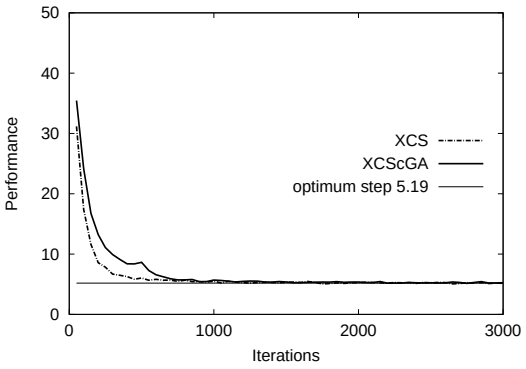

図 7 Maze6 における XCS と XCScGA の食料までのステップ数

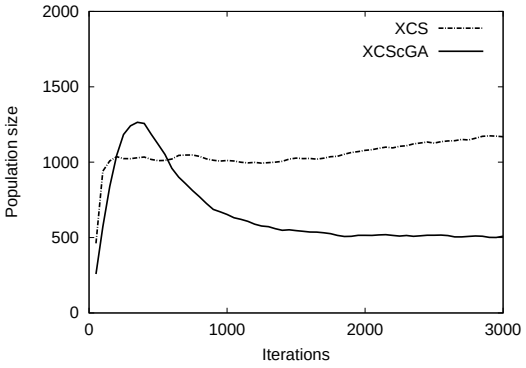

図 8 Maze6 における XCS と XCScGA の分類子数

はの分類子数は 508 であり，XCScGA は XCS の分類子数を約 54.7%削減していることがわかる．

7. おわりに

本論文では，現在主流の学習分類子システム (XCS) が膨大な分類子数を要するという問題に対し，Compact Genetic Algorithm を用いた確率モデル型分類子生成法を提案した．提案分類子生成法は，分類子がもつ部分解の存在確率をモデル化することで不要な分類子の生成を抑制し，XCS が必要な分類子数を削減する．提案手法の有効性を検証するために，提案法を導入した XCS(XCScGA) を教師あり学習問題 (multiplexer 問題) と強化学習問題 (block world 問題) に適用したところ，次の知見を得た．まず，1) 提案 LCS は従来 LCS よりも少ない学習回数で最適解を学習可能であり，2) 従来 LCS が学習した分類子数に対し，提案 LCS は最小でも 54%(最大で 76%) 削減した分類子数で学習可能であることを示した．

参考文献

[1] Bacardit, J. and Butz, M. V.: *Data Mining in Learning Classifier Systems: Comparing XCS with GAssist*, IlliGAL Report No. 2004030 (2004).
[2] Bernadó-mansilla, E. and M.Garrell-Guij, J.: Accuracy-based Learning Classifier Systems: Models, Analysis and Applications to Classification Tasks, *Evolutionary Computation*, Vol. 11, pp. 209–238 (2003).
[3] Butz, M. V.: *Rule-Based Evolutionary Online Learning Systems*, Springer (2006).
[4] Butz, M. V., Goldberg, D. E. and Tharakunnel, K.: Analysis and Improvement of Fitness Exploitation in XCS: Bounding Models, Tournament Selection, and Bilateral Accuracy, *Evolutionary Computation*, Vol. 11, No. 3, pp. 239–277 (2003).
[5] Butz, M. V., Goldberg, D. and Lanzi, P. L.: Gradient Descent Methods in Learning Classifier Systems: Improving XCS Performance in Multistep Problems, *Evolutionary Computation*, Vol. 9, No. 5, pp. 452–473 (2005).
[6] Butz, M. V., Kovacs, T., Lanzi, P. L. and Wilson, S. W.: Toward a Theory of Generalization and Learning in XCS, *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 1, pp. 28–46 (2004).
[7] Butz, M. V. and Wilson, S. W.: An Algorithmic Description of XCS, *Journal of Soft Computing*, Vol. 6, No. 3–4, pp. 144–153 (2002).
[8] Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning, *Addison Wesley* (1989).
[9] Harik, G. R., Lobo, F. G. and Goldberg, D. E.: The compact genetic algorithm, *IEEE Trans. Evolutionary Computation*, Vol. 3, No. 4, pp. 287–297 (1999).
[10] Holland, J. H.: Escaping Brittleness: The Possibilities of General Purpose Learning Algorithms Applied to Parallel Rule-based System, *Machine Learning*, Vol. 2, pp. 593–623 (1986).
[11] Hurst, J., Bull, L. and Melhuish, C.: TCS Learning Classifier System Controller on a Real Robot, *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, PPSN VII, Springer-Verlag, pp. 588–600 (2002).
[12] Larrañaga, P. and Lozano, J.: *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*, Kluwer Academic Publisher (2001).
[13] Llora, X., Sastry, K. and Goldberg, D.: The compact classifier system: scalability analysis and first results, *The 2005 IEEE Congress on Evolutionary Computation2005*, pp. 596–603 (2005).
[14] Sutton, R. S.: Learning to Predict by the Methods of Temporal Differences, *Machine Learning*, Vol. 3, No. 1, pp. 9–44 (1988).
[15] Urbanowicz, R. J., Granizo-Mackenzie, A. and Moore, J. H.: Instance-linked attribute tracking and feedback for michigan-style supervised learning classifier systems, *GECCO*, pp. 927–934 (2012).
[16] Urbanowicz, R. J., Andrew, A. S., Karagas, M. R. and Moore, J. H.: Role of genetic heterogeneity and epistasis in bladder cancer susceptibility and outcome: a learning classifier system approach, *Journal of the American Medical Informatics Association*, Vol. 20, No. 4, pp. 603–612 (2013).
[17] Whitley, D. and Kauth, J.: *GENITOR: A different genetic algorithm*, Colorado State University, Department of Computer Science (1988).
[18] Wilson, S. W.: Classifier fitness based on accuracy, *Evolutionary Computation*, Vol. 3, No. 2, pp. 149–175 (1995).

ⓒ 2013 Information Processing Society of Japan