蟻の餌争奪ゲームによるマルチエージェントシステムの協調動作評価

久保正男† 嘉数侑昇††

一般に解析的なモデリングが困難な対象や，対象自体の持つ構造が変動するような問題に対し，自律的な振る舞いをするエージェント群を導入し問題解決を図ろうとするシステム，いわゆるマルチエージェントシステムはそれが持つ並列性，自律性により柔軟な問題解決能力をもつであろうことが期待されている．マルチエージェントシステムがその期待能力を発揮するためには対象問題に依存した各エージェント間の協調機能の発現が不可欠となる．従来のマルチエージェントシステムの研究においてはこの原点の欠如がみられる．本研究では提案するそれぞれの能力の異なる複数のエージェントとしての蟻群から構成される二つの蟻塚間における蟻争奪ゲームを対象例として，各蟻の自陣との状態に応じた協調機能をいかにして獲得させるか，また獲得した協調機能の定量的評価をいかに行うかについて確率的学習オートマトンをベースにした理論展開を図り行った種々の計算機実験を通して，問題の持つ性質の推論を行い，学習環境と獲得された協調動作との関係や微妙なエージェントの能力差によるシステム内での役割分担の発現に関する実験を行い，その結果，本学習マルチエージェントシステムが動的な環境変化に適応できることや各エージェントが自己の能力に応じたシステム寄与を獲得していることを確認している．さらに学習マルチエージェントシステム間の能力差や学習環境への考察を行っている．

Evaluation of Coordinated Motions of Multi-Agent Systems on Competition for Food between Ant Colonies

Masao Kubo† and Yukinori Kakazu††

In this study, a simple and powerful methodology for realizing a distributed and autonomous system is proposed. Usually, *Multi Agent Systems* have difficulty in generating actions that will be suitable taken in their totality, and which we should refer to here as *Coordinated motions*. The proposed methodology improves a colony's total activity through an individual learning process for each agent by Stochastic Learning Automaton (SLA). We propose a Game, similar to football, called The Competition for Food between Ant Colonies. By applying such system to an dynamic changeable environment, we will demonstrate its adaptability. This ability should be evaluated solely through its solution-reaching capability against a highly changeable environment. This game environment serves sufficiently as a complex and changeable environment. By the winning rate of the Competition, the suitability of the coordinated motions is demonstrated.

1. はじめに

本論文は，動的環境に適応し問題解決を行うマルチエージェントシステムの構築を目的とし，そのために学習による協調動作の獲得と，2人ゲーム環境での勝率による相対的システム評価を行う．

近年，マルチエージェントシステム[1)~4)]という計算機構が注目を集めている．マルチエージェントシステムは複数の分権的処理エージェントからなり，エージェント間の協調関係を自己組織化しながら問題解決を行う．この並列性，自己組織化能力により，問題構造の変化が既知でない問題に対するロバストで高速な解決手法[5)~7)]として期待されている．

一般に，エージェントが個別に処理可能な問題には限界があり，個別には処理困難な問題を解決するためには，個々のエージェントの処理能力を統合しなければならない．したがってマルチエージェントシステムの問題解決能力は適切な協調関係の自己組織化能力に深く依存している．

† 北海道大学工学部情報工学科
Division of Information Engineering, Hokkaido University
†† 北海道大学工学部精密工学科
Department of Precision Engineering, Hokkaido University

では分権的なエージェント群が問題に対し自己組織化するには自己の処理能力のシステムへの寄与を各エージェントが認識する必要があるが，問題の構造変化が既知でなく，また解法が複数存在する場合にはこれをアプリオリな情報として設計者が各エージェントに記述することは，マルチエージェントへの要求であるロバストな解決手法という観点から相応しくない[8)~10)]．

したがって，システムとして問題解決を行いながら，各エージェントはシステムへの寄与を獲得しなければならない．そこでここでは，学習による適切な協調関係の獲得を行う[11),12)]．各エージェントがおのおのの環境を観測し，処理行動を出力し，各評価関数に準じて学習することにより，経験的にシステムとして整合性のとれた自己組織化が実現され，さまざまな問題環境においてシステム全体のロバスト性をさらに拡大するものとして期待できる．

この際，設計者は各エージェントに評価関数やパラメータを与えなければならない．各エージェントはこれらに基づいて学習を行うゆえに，評価関数やパラメータ設定は自己組織化の適切さや生成速度に大きく影響を及ぼすと思われる．したがってこれらの前提要素を適当に決定するためには，システム全体の振る舞いに関する何らかの評価が必要となる．このために理想的な評価方法として，エージェント間の相互関係を解析しボトムアップ的にトータルな評価を行うことが考えられるがこれは今後の課題である．

ここでは2人ゲーム環境で二つのマルチエージェントシステムを対戦させ，勝率よりその評価関数やパラメータの適切さを推測するような手法によるシステム評価を行う．環境変化，すなわち相手の生成する次第に高度になる戦略に対して適切な自己組織化が実現できなかったり，組織化が遅れれば通算としての勝率に大きく現われることが予想される．よってゲーム環境が持つ上記の性質を利用した性能評価方法は，動的環境を踏まえたシステムの目的達成能力を相対的にではあるが比較でき，マルチエージェントシステムの現実的評価方法として妥当性をもつと思われる．

次章で本ゲーム環境である蟻の餌争奪ゲームを詳述する．次にエージェントの学習機能を実現する基本型である，強化学習法の一つ，確率的学習オートマトンの概略を紹介したのち，本エージェントを記述し，簡単な計算機実験を行う．最後に本論文で得られた評価基準とパラメータと協調動作能力に関して考察を行う．

2. 蟻の餌争奪ゲーム

ここで導入する蟻の餌争奪ゲームは，生物の集餌行動にヒントを得た，複数の駒の操作に基づく2人ゲームである．

2.1 蟻の餌争奪ゲーム

いま，有限の二次元連続平面上に複数の餌と蟻塚を二個配置する（図1）．各蟻塚にはそれぞれ異なった能力をもつ任意数の蟻からなる一つのコロニーが生息している．各蟻は住み家である蟻塚へ餌を運ぶ．本ゲームではコロニー間で蟻塚に溜まった餌の量を競う．問題設定を，以下に記述する．

$$\tilde{C}_i = \{A_1^i, A_2^i, \cdots A_{m_i}^i, \widetilde{Nest}^i\} \quad (1)$$

$$|\tilde{C}_i| = m_i \quad (2)$$

$$nest_l^i \in \widetilde{Nest}^i \quad (3)$$

$$nest_l^i = (nex_l^i, ney_l^i) \quad (4)$$

各コロニーは蟻と蟻塚からなる（式(1)）．ここで，$\tilde{C}_i$ はコロニー i を表し，$\widetilde{Nest}^i$ はコロニー i の蟻塚集合，nex_l^i, ney_l^i は蟻塚 $nest_l^i$ の x, y 座標である．

また蟻はゲームフィールド上での絶対座標，牽引と移動能力と移動方向で表す．

$$A_j^i(t) = \{Px_j^i(t), Py_j^i(t), Rn_j^i(t), Tr_j^i(t), Vc_j^i(t)\} \quad (5)$$

ここで $A_j^i(t)$ は時刻 t におけるコロニー i に属する j 番目の蟻，$Px_j^i(t)$, $Py_j^i(t)$ は $A_j^i(t)$ の x, y 座標値で

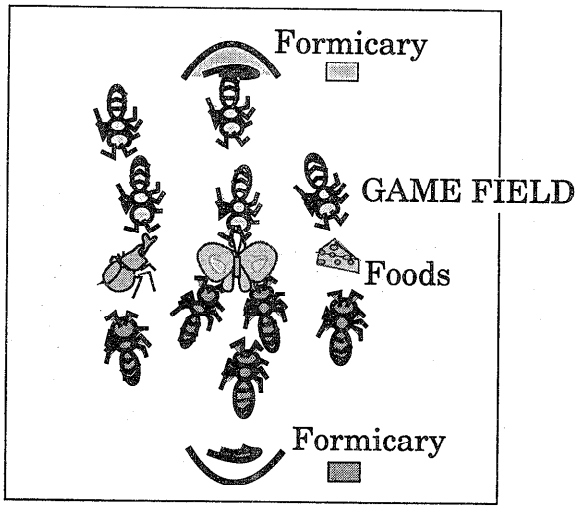


図1 蟻の餌争奪ゲーム
Fig. 1 Competition for foods between ant colonies.

あり，$Rn_j^i(t)$, $Tr_j^i(t)$, $Vc_j^i(t)$ は $A_j^i(t)$ の移動能力，餌牽引力および移動方向ベクトルである．餌は式(6)に示すように絶対座標と移動ベクトルで表現する．

$$\tilde{F}=\{F_1, F_2, \cdots, F_n\} \quad (6)$$

$$|\tilde{F}|=n \quad (7)$$

$$F_k(t)=\{Px_k^F(t), Py_k^F(t), Sp_k(t)\} \quad (8)$$

ここで $\tilde{F}$ をゲームフィールドに存在するすべての餌とし，$F_k(t)$ を k 番目の餌の時刻 t における状態を示す．$Px_j^i(t)$, $Py_j^i(t)$ は $F_k(t)$ の x 座標，y 座標，また $Sp_k(t)$ は時刻 t における移動方向ベクトルであり，式(9)にように規定する．

$$Sp_k(t)=\sum_{Pull(A_j^i(t), F_k(t))=true}^{\kappa} Tr_j^i(t)\cdot Vc_j^i(t) \quad (9)$$

(9)式は餌を引く蟻の牽引力の総和の差に比例して餌の移動速度が定まることを示しており，蟻は力を合わせることにより，早く蟻塚まで餌を引くことが可能である．ここで，κ は比例定数で，$Pull(A_j^i(t), F_k(t))$ は時刻 t においてコロニー i に含まれる蟻 j が餌 k を牽引していることを表す．

蟻が餌を蟻塚へと運ぶと(10)式のように餌は各コロニーの所有物となる．本ゲームはこの餌の数をコロニー間で競うゲームである．

$$sqrt((nex_i^i-Px_k^F(t))^2+(ney_i^i-Py_k^F(t))^2)=0.0 \quad (10)$$

2.2 蟻の餌争奪ゲームにおける協調動作と性質

以上の定義によって表現した蟻は自己の位置を与えられた能力の範囲で変更しながら餌を蟻塚まで運ぶ．**図2**に簡単な配置列を示すように，本ゲームにおける協調動作とは蟻の餌への配置の様相とみなすことができる．配置1は一度に複数の餌を運ぶことができるが，牽引速度が遅いため，敵の蟻に襲われ奪われる可能性がある．したがって餌場付近に敵蟻がいる場合には用いるべきではない．配置2は複数の蟻で少数の餌を牽引することにより素早く蟻塚まで運ぶことができる．しかし，餌場から遠ざかることになり，無防備とも考えられる．したがってこの配置は餌場に戻る必要のない場合に極めて有効と思われる．また移動力が高い蟻がいるときには配置3のように縦方向に並ぶ方法も効果的である．餌場に留まった蟻が残った餌を守っている間に仲間が餌を蟻塚へと運び込める．また組を作り行動を共にする手法も堅実な配置である（配置4）．このように本ゲームでは仲間との位置関係が非常に重要でありしたがって緊密な協調関係が要求されるゲームと見なすことができる．

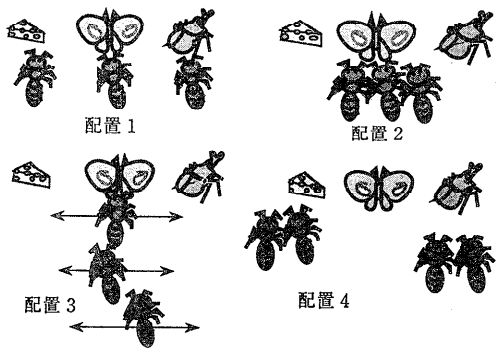


図 2 本ゲームにおける協調動作例
Fig. 2 Examples for coordinated motions in this game.

また，双方が学習を行う場合，相手を妨げ勝つ戦略を学習すると思われる．この性質によってマルチエージェントシステムにはより高度な協調関係の生成が要求される．理想的な学習下では，この対戦によって次第にお互いの協調動作生成能力を高めて行くと思われる．

したがって，本ゲームがシステム内での緊密な協調関係を要求すること，相手の協調動作や戦況に応じて適切な協調関係を迅速に生成する必要があること，また繰り返しゲームを行った場合では常に自己の協調関係をリファインする必要があることから，繰り返しゲームを行った際の勝率は相手の作りだす環境へのトータルな適応度と見なすことができ，総合的なマルチエージェントシステムの適応能力を測る上で妥当性を持つ．相手の能力の変化や蟻の死亡といった変化，相手の配置傾向の変化といった多様な環境へ適切に適応できるならば，そのコロニーは高勝率を残すことが予想される．

例えば適切な協調関係にあった仲間の蟻が突然死亡した状況を考えてみる．協調関係にある場合，仲間はおのおのの観測行動の結果を通じて相互の行動を踏まえた上で運動を行っているものと考えられる．今，蟻1の死を，能力が0であることによって表現する．これは“確かに蟻1は存在する，けれども蟻1は動かない”という無機物としての存在状況に対応する．これによって死の前に観測した状況とは異なる観測結果がもたらされ，仲間の蟻は蟻の死を暗に認識し，必要に応じて蟻の死が発生するまでに培った協調関係の再構成を行う．これはマルチエージェントシステムにおけるエージェントの故障に該当する．

上記のような個々のエージェントの行動が集団全体に影響を持つゲームは無数に考えられる．例えば将棋の各駒をエージェントと考え，勝率によって協調動作能力を評価することも可能である．しかし，将棋やチェスといった各駒の能力が極端に異なり，制約が多い大規模ゲームを，ここで目的とする協調動作の評価という観点からモデル化することは現状では困難である．ゆえに，相互の協調動作生成能力のみが勝率に影響を与える単純化したゲームのモデル化を図ることにする．

3. 設計方針

ここでは学習エージェント蟻をゲームフィールド上の状態を一意に復元可能な情報（全情報）ではなく，その部分集合となる情報を用い，強化学習手法によって構成する．

一般に計算システムは集中処理[13)~16)]と分権的処理[2),7),17)~20)]に分類できる．集中処理は観測された情報をある一つのプランナーに集め，各蟻にプランを割り付けるという計算機構である．これに対し，マルチエージェントシステムに代表される分権的な手法は，**図3**に示すように各蟻がプランナーとなり，おのおののプランを作成する．全く等しい情報を元に計算を双方の計算機構で行った場合，一般に分権的な手法はその並列性ゆえに処理速度に優れるが，生成されたプラン間の整合性に疑問が生じる．

そこでマルチエージェントシステムで整合性のとれたプランを生成するために，各エージェントが観測してからプランを生成するまでの時間，すなわち処理速度を高めることによって不整合なプランが生成された際に必要なリプランニング時間に割り当てることを提案しこれを設計方針とする．

また扱う情報は現実的な応用を考えた場合，すべての要因を用意することが困難な環境下で問題解決を余儀なくされることがある．そこでここでは部分情報を元に各蟻はプランニングを行う．具体的には，各蟻は，他の蟻に関する情報を一切用いず，餌に関するすべての情報と自己の情報をのみを元にプランニングを行う．つまり餌の観測結果をとおして，他の蟻の状況を推測しなければならない．

学習手法には $\alpha\beta$ 法を用いて相手の配置と各蟻の行動を探索によって決定するもの[21)~23)]やクラシファイアーシステム[24)]，ニューラルネットワーク[25)]が一般によく知られているが，探索による手法は探索木の処理コストがかかることから，観測してから行動出力までの時間を突き詰める本目的にはそぐわない．クラシファイアーシステムはクレジットアサイメントの際のパラメータのロバスト性や Genetic Algorithm[26)] のもつ収束性から，常に学習を要求される本システムには適さない．これに対し強化学習は簡単な線形的な学習スキーマによる学習手法であるが，処理速度が早く，オンライン学習性，パラメータのロバスト性に優れている．

以上より**図4**に示すような，各エージェントにおいて，全体として有効な環境の観測結果と行動対の獲得を実現する手法として強化学習手法の一つである確率的学習オートマトン(Stochastic Learning Automata; SLA)[27)]を用いる．他の強化学習手法として，Q-Learning[12)] が提案されているが，一般に SLA よりも学習対象の安定性が必要であり，またパラメータに対して敏感である．そこでパラメータのロバスト性について優れた SLA に基づいて，問題向きに改良して学習機能を実現する．

4. 確率的学習オートマトン

コロニーにとって有効な行動を個々の蟻が生成，学習することを目的として本問題向きに構築し直した確率的学習オートマトン (Stochastic Learning Automata; SLA) を導入する．一般によく知られて

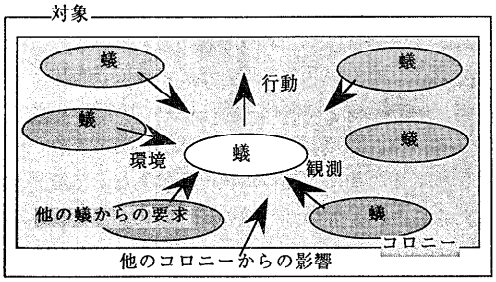


図3 本マルチエージェントシステムの概念
Fig. 3 Concepts of this multi-agent system.

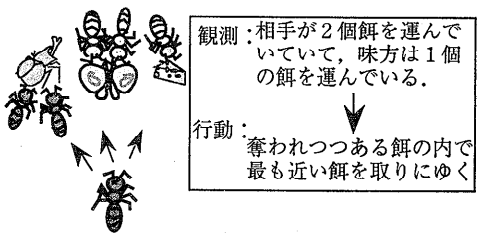


図4 環境の観測結果と行動対の例
Fig. 4 The observation and action of the agent.

いるように SLA は先験的な入力状態集合と出力集合，入力状態集合と出力集合のマッピングを確率的に保存する出力選択確率ベクトル集合から構成されている[12),28)-30)]．SLA ではある入力に対しそれに適切な出力を獲得した出力選択確率ベクトルに基づいて出力する．また SLA の学習とは出力選択確率ベクトルをあらかじめ設定した評価基準に基づいて更新することである．上記の機能によって入力状態とそれに対する適切な出力の対を獲得することが可能となる．個々の蟻は，例えば餌や他の蟻の位置といった自身を取り巻く環境の観測結果を自己の SLA への入力とし，次の移動方向を出力とすることで環境状況に見合った適切な行動を行うことが期待される．

図 5 に示すように，SLA は一般に，入力状態集合 $\tilde{\phi}$，出力集合 $\tilde{\alpha}$，強化信号 $\tilde{\beta}$，出力選択確率ベクトル集合 $\tilde{\boldsymbol{P}}$ と評価アルゴリズム $\mathcal{A}$ から構成されている．

$$SLA=\{\tilde{\phi},\tilde{\alpha},\tilde{\beta},\tilde{\boldsymbol{P}},\mathcal{A}\} \tag{11}$$

$$\tilde{\phi}=\{\phi_1,\phi_2,\cdots,\phi_m\} \tag{12}$$

$$\tilde{\boldsymbol{P}}=\{\tilde{P}_1,\tilde{P}_2,\cdots,\tilde{P}_m\} \tag{13}$$

$$\tilde{P}_i=\{P_{i1},P_{i2},\cdots,P_{in}\} \tag{14}$$

$$P_{ij}=Pr(j|i) \tag{15}$$

$$\tilde{\alpha}=\{\alpha_1,\alpha_2,\cdots,\alpha_n\} \tag{16}$$

$$\tilde{\beta}=\{1,0\} \tag{17}$$

ここで，m は入力状態数，n は出力数，P_{ij} は入力状態 ϕ_i のとき出力 α_j を行う条件付き確率を行う．したがって，任意の入力状態 ϕ_i において，出力選択確率ベクトル集合 $\tilde{\boldsymbol{P}}$ は条件(18)を満たす．

$$\sum_{j=1}^{n}P_{ij}=\sum_{j=1}^{n}Pr(j|i)=1.0 \tag{18}$$

環境の観測結果が入力状態集合の ϕ_i に含まれるならば，出力される行動は出力選択確率ベクトル $\tilde{P}_i$ に従って，確率的に $\tilde{\alpha}$ の中から選択される．

一方，学習は出力選択確率ベクトルの更新によって行われる．環境に出力した行動が，先験的な評価アルゴリズム $\mathcal{A}$ に従って，適切ならば，強化信号 1 が SLA に送られ，その入力状態集合と出力集合の結合が，出力選択確率ベクトルを高めることによって，強められる．逆に出力行動が不適切であったなら（強化信号＝0），結合は弱められる．例えば，入力状態 ϕ_i で，出力 α_j を行ったときに，強化信号 1 が SLA に与えられたならば，

$$P_{ij}\Leftarrow P_{ij}+reward\cdot(1.0-P_{ij}) \tag{19}$$

$$P_{ik\neq j}\Leftarrow reward\cdot P_{ik\neq j} \tag{20}$$

を用いて，線形に更新する．ここで，k は j 以外の出力を表し，$reward$ は更新量を表す．出力選択確率ベクトルの更新量を決定するアルゴリズムには，L_{R-P}，L_{R-I} スキーム等[27)]が提案されており，以下では L_{R-P} を簡略化した更新法を導入した．

図 5 に示したとおり，本システムでの SLA と一般型の違いはその強化信号の伝達経路である．一般型はデザイナー等より，強化信号が送られるが，本システムでは個々のエージェント内にある Evaluator よりそれが送信される．

5. 蟻とコロニー

本モデルの概略を**図 6** に示す．これまでの議論を踏まえ，以下では蟻の詳細な記述を行う．まず，各蟻の行動は次の ***p*1** から ***p*5** からなるものとする．

***p*1**：観測により情報を収集する行動．ここで，各蟻は自身の現況をもたらしたところの前回の SLA の出力（プラン）を覚えているものとする．エージェント $A_j^i(t)$ の観測情報は

$$Ob_j^i(t)=Px_j^i(t)\cup Py_j^i(t)\cup Vc_j^i(t)\cup\tilde{F} \tag{21}$$

自己の現在位置と移動ベクトルおよび餌のすべての状態である．

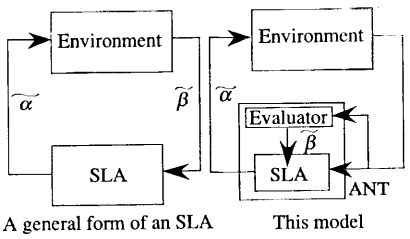


図 5 一般型と本 SLA の比較
Fig. 5 The general form of SLA.

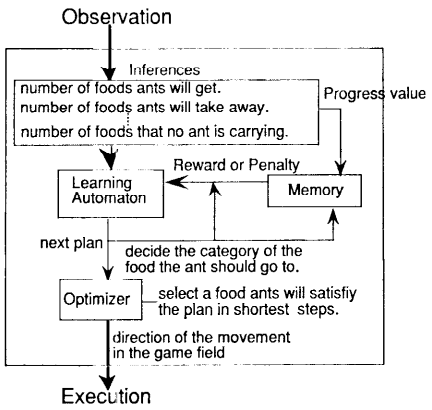


図 6 蟻の構成
Fig. 6 The frame of the agent "ANT".

p2：得られた情報から現在コロニーが置かれた状況を推論し，推論結果と対応する SLA の入力状態を選択する行動．入力状態集合は環境の観測結果に対応する．観測した情報から，各蟻は次の5項目について推論を行う．すなわち蟻塚へ運び込める餌の数 (inf^i_{j1})，および獲得に必要な時間 (inf^i_{j2})，敵に奪われる餌数 (inf^i_{j3}) および奪われる時間 (inf^i_{j4})，どの蟻も注目していないと思われる餌の数 (inf^i_{j5}) についてである．これらを定式化すると

$$\widetilde{inf}^i_j = \{inf^i_{j1}, inf^i_{j2}, \cdots, inf^i_{j5}\} \tag{22}$$

$$inf^i_{jk} | Ob^i_j(t) \Rightarrow R^i_{jk} \subset \mathcal{R} \tag{23}$$

となり，各 inf^i_{jk} から，推論結果・実数 R^i_{jk} を得る．

しかし，当然ながら協調動作を実現するためにはコロニーのおかれた状況を推論するだけでは不十分である．適切な協調動作は自律エージェントの実行義務[8),31),32)]が相互に両立している動作または状況と考えることができる．言い換えれば，適切な協調動作下においては，蟻の選択した行動はコロニーにとって必要十分な行動であり，蟻の行動は相互に保証されていると仮定できる．この結果，蟻の基本能力や現在の位置といった個々の性質と他の蟻からの要求が一致し，一意に個々の行動の必要十分性が得られているコロニーを，適切な協調動作下にあるコロニーと考えることができる．したがって適切な協調動作を実現するために必要な情報として，コロニー内での各蟻の役割の相異を表す情報が必要である．ここでは役割を表す情報として，後に詳述するが，各蟻が選択した SLA の出力である，プラン α^i_j に注目する．

以上より，協調動作の実現のために，コロニーの立場に関する情報として五つの推論結果を離散化したものと，コロニー内での各蟻の立場を意味する履歴に対応するプランを組み合わせ，432 個の入力状態を持つ入力状態集合 $\tilde{\phi}^i_j$ を用意し，SLA を構成した☆．

$$\tilde{\phi}^i_j = \{\phi^i_{j1}, \phi^i_{j2}, \cdots, \phi^i_{j432}\} \tag{24}$$

そこで推論結果 R^i_{jk} と前回の選択プラン $\alpha^i_j(t-1)$ が満たす状態 ϕ^i_{jl} を $\tilde{\phi}^i_j$ から選択する．このとき，いかなる推論結果と $\alpha^i_j(t-1)$ に対しても入力状態集合 $\tilde{\phi}^i_j$ から式(25)を満たす入力状態 ϕ^i_{jl} が一つだけ存在するものとする．

$$\phi^i_{jl} \supseteq \bigcup_{k=1}^{5} R^i_{jk} \cup \alpha^i_j(t-1) \tag{25}$$

p3：推論結果と対応する入力状態の出力選択確率ベクトルに従って確率的にプランを選択する行動．**p2** で得られた状態 ϕ^i_{jl} に対応する出力選択確率ベクトル $\tilde{P}^i_{jl}$ に従って確率的にプラン集合 $\tilde{\alpha}^i_j$ から行動として一つのプランを選択する

$$\tilde{\boldsymbol{P}}^i_j = \{\tilde{P}^i_{j1}, \tilde{P}^i_{j2}, \cdots, \tilde{P}^i_{j432}\} \tag{26}$$

$$\tilde{P}^i_{jl} = \{P^i_{jl1}, P^i_{jl2}, \cdots, P^i_{jl5}\} \tag{27}$$

$$\tilde{\alpha}^i_j = \left\{ \begin{matrix} NO\_MARK,\ SPEED\_UP,\ CARRY \\ COLLECT\_FOOD,\ DEADLOCK \end{matrix} \right\} \tag{28}$$

式(28)に示すとおり，ここでの SLA の出力はプランという言葉どおりマクロな移動計画である．右や左といったプリミティブな移動計画ではなく，各プランは移動目標とする餌の種類を主に表している．これは，4章で言及したように SLA は入力状態と出力集合とのマッピングを学習により獲得するが，このとき獲得すべき確率分布が問題解決に直結していることが望ましい．各蟻の行動として，右や左に移動するといった移動方向を直接，出力集合として構成すると各入力状態に固有の確率分布を得ることが期待できない．そこで餌を分類し，移動目標の種類を出力集合とした．

具体的には図7に示すように，*NO_MARK* はどの蟻にも引かれていない餌，*SPEED_UP* は自塚へと運ばれている餌，*COLLECT_FOOD* は *SPEED_UP* の逆の餌を意味する．これらの分類はしきい値処理によって行っており，いずれにも属さない餌を *DEADLOCK* とする．各蟻はこのプランに従って，条件を満たす餌の方向へ，各蟻固有の移動速度に従って移動する．実際，餌へ到達するためには数単位時間

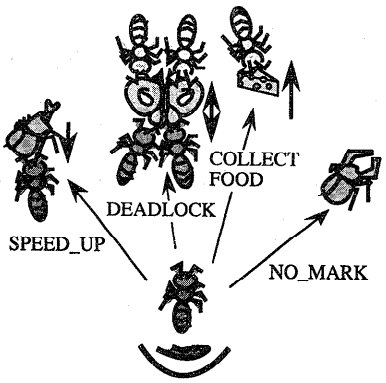


図7 プラン
Fig. 7 The plans.

☆ 履歴として保存するのは基本的には前回の選択プラン $\alpha^i_j(t-1)$ である．ただし，*CARRY* を選択した際にはこの時点での履歴プランを継続して履歴として保持する．したがって式(24)の432状態は推論結果をしきい値処理したものと *CARRY* を除く4プランと *CARRY* のオンオフからなる直積空間の分割である．

必要なことが多く，したがって数ステップにわたって同一のプランを選択する必要がある．このようにして餌に辿り着いたとき，はじめて餌を牽引する条件が満たされる．*CARRY* は牽引する条件が満たされているとき，餌を引くことを表す．

p4：プランの条件を最短時間で満たすことが期待できる餌の方向へ移動する行動．選択したプラン $\alpha_j^i(t)$ からゲームフィールド上での移動方向を導出する．選択したプランは二次元平面上での移動方向を一意に指し示すものではない．そこでプランを Optimizer に入力しプランを最短時間で達成可能な餌を特定し，移動ベクトルを出力する[33]．

p5：プランを変更したときコロニーが置かれた状況（戦況）の改善具合に応じて出力選択確率ベクトルを更新する．前章で述べたように学習は各 SLA の出力選択確率ベクトル集合を更新することによって行われるが，そのためには選択した行動を評価するアルゴリズムが必要である．一般的に，コロニー全体の行動を評価し，各蟻の行動にグローバルな立場から優劣を与えることは，相互関係が複雑になればなるほど問題性質上困難となるので，ここでは各蟻が個々の基準に基づいて自己の行動のコロニーへの寄与を評価する手法を考える．そのために，まずコロニーの状況を表す戦況値（progress）を導入する．戦況値は ***p2*** で用いた五つの推論結果を元に，各蟻がもつ基準に従って計算される整数値である．値が大きいほど，コロニーの状況が有利であることを意味し，小さいほど，不利な状況に対応する．式(29)に設定例を示す．

```
if nowAwayFood>startAwayFood then
  progress=−2
if nowAwayFood<startAwayFood then
  progress=+2
if nowAwayTime<startAwayTime then
  progress=−1
if nowAwayTime>startAwayTime then
  progress=+1
if nowGetFood>startGetFood then
  progress=+2
if nowGetFood<startGetFood then
  progress=−2
if nowGetTime<startGetTime then
  progress=+1
if nowGetTime>startGetTime then
  progress=−1                    (29)
```

ここで，各項は以下の意味をもつ．すなわち *now*, *start* は現在とプラン実行開始時，*Get*, *Away* は自陣が獲得する餌，敵に奪われる餌を示し，*Food*, *Time* は餌，時間を表し，例えば，*if nowGetFood>startGetFood then progress=+2* はプラン開始時よりも，自陣が獲得可能な餌が多ければ戦況値に2加えることを示す．

戦況値を評価基準として出力選択確率ベクトルを更新するわけであるが，一般的な手法[28]でとられるような SLA への入力を行うたびに更新することは本モデルの場合，有効な方法ではない．適切な協調動作下では短時間で戦況値が大きく変化することはなく対応する入力状態が同一であることが予想され，随時更新を実施すると過度な出力選択確率ベクトルの更新が行われる恐れがある．そこで図8にしめすように，評価および更新は選択したプランが前回のプランと異なるときに行う．十分な学習後では，SLA をリコグナイザーを見なすことができ，選択したプランがコロニーとして適切ならば，数ステップ継続して出力され，逆に不適切であるならばすぐに変更されるはずである．したがって，プラン変更時に評価を行うことが最も妥当である．

もし A_j^i において，戦況が改善されていたならば強化信号1が送られ，次式(30)のように出力選択確率ベクトルを更新する．

$$
\begin{aligned}
&if((P_{jkl}^i + reward/100.0) \geq 1.0)\\
&\quad P_{jk*}^i \Leftarrow P_{jk*}^i\\
&if((P_{jkl}^i + reward/100.0) < 1.0)\\
&\quad P_{jkl}^i \Leftarrow P_{jkl}^i + reward\cdot(1.0 - P_{jkl}^i)/100.0\\
&\quad P_{jkx*l}^i \Leftarrow P_{jkx*l}^i\cdot(1.0 - reward/100.0) \qquad (30)
\end{aligned}
$$

ここで，l は A_j^i が選択したプラン，k はプラン l をはじめて選択したときの入力状態 ϕ の番号，＊は任意を表す．またここでの *reward* は SLA 一般型とは

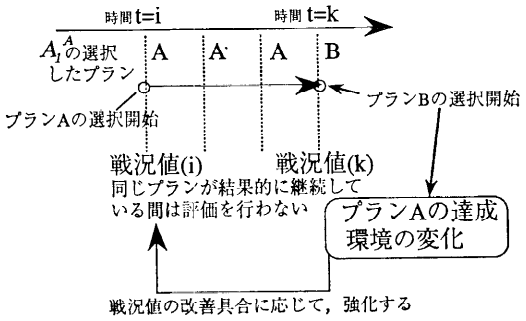


図 8 SLA の更新過程
Fig. 8 The learning timing of SLA.

異なり，100 倍している．

6. 計算機実験および考察

6.1 学習の収束性

初めに SLA の収束性[27]について検証する．SLA は定常な環境では一定の出力選択確率ベクトルに到達し安定する性質を持つ．この性質に着目し，提案したマルチエージェントシステムの基本的な枠組みの正当性について検証する．

実験1のゲーム設定は以下のとおりである．ゲームフィールド上にはそれぞれ4匹の蟻から構成されたコロニー二つと，ゲームフィールド中央に置かれた7個の餌がある．蟻の走る能力は蟻塚から最近の餌まで約6単位時間で移動できる程度に統一した．また基礎能力である走る能力および牽引する能力は，牽引する能力を走る能力の約8割に設定した．蟻塚は簡単化のためにゴールラインを引き代用した．餌は蟻塚方向へ長さをもち，中心がゴールラインを越えると餌の権利が決定する．ただしゴールラインの存在を明示してはいない．蟻の初期位置はこのゴールラインに沿って乱数でゲームごとに変更し，実行した．以下の実験でゲームを繰り返して行う際には実験1同様，初期位置はゲームごとに乱数で発生させた．また確率的学習オートマトンの入力状態や学習パラメータは学習が可能と思われる範囲で設定した．

SLA のリワード，ペナルティをそれぞれ 15.0, 0.1 として，蟻の初期位置を除いて，全く同じパラメータ設定下で 500 ゲーム対戦させた．各ゲームは，どちらかのコロニーが四つの餌を獲得することを勝利条件として時間制限を設けず実行した．

図9は横軸にゲーム回数，縦軸に一方のコロニーの1エージェントが1ゲームの中にペナルティを与えた回数である．ゲーム数が進むにつれペナルティが与えられる回数が減り，SLA を導入する上で適切な入力状態集合が用意できていることがわかる．また極端にペナルティ回数が異なるゲームが見受けられるが，耐久戦や速攻戦といった，進行の違いによるゲーム時間の違いが原因だと思われる．

次に学習能力とエージェント数の変化に対する適応性について検証する．そこで次のような3種のコロニーを用意した．コロニー1は4匹の蟻から構成されており，各蟻の SLA のリワード，ペナルティをそれぞれ 15.0, 0.1 として，学習する．コロニー2は3匹，コロニー3は2匹の蟻から構成されており，各蟻の SLA はコロニー1と等しい学習係数を用いて学習する．コロニー4は4匹の蟻から構成されているが，各蟻の SLA のリワード，ペナルティをそれぞれ 0.0, 0.0 として学習を行わず，ランダムなプラン選択を行う．コロニー4の各蟻は餌を引く条件を満たしているならば，40% の確率で餌を引く．したがってある時間内ですべての餌を蟻塚へ運ぶ性質を持つ．以後の実験では，コロニー4を学習を行わずランダム選択を行うコロニーとして扱うものとする．

ゲーム設定は以上を除いて実験1と等しい．コロニー1とコロニー4，コロニー2とコロニー4，コロニー3とコロニー4の対戦を 500 ゲームずつ行った結果を**図10**に示す．横軸にゲーム数，縦軸にコロニー1, コロニー2, コロニー3の累積勝率を百分率で表示した．黒線は 500 ゲーム付近ではコロニー1が同時の能力の相手には負けないことを表しており，またコロニー2，3の対戦より，自己の基礎能力を上回るコロニーに対して，学習によって適切な戦況に適した協調動作の生成が行われていることが推測できる．また個体数の増加に対して急激に勝率が上がることより，本

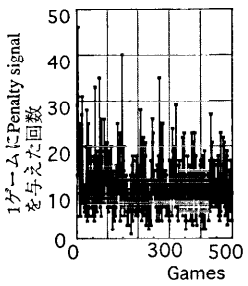


図 9 学習の収束
Fig. 9 The convergence of SLA.

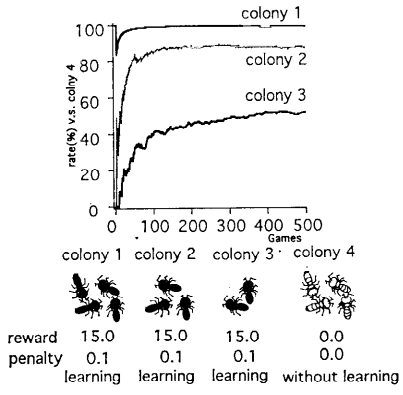


図 10 ランダム戦略コロニーとの戦い
Fig. 10 V.S. random-action-selecting colony.

ゲームが協調動作生成能力に大きく依存していることがわかる．

6.2 蟻の死亡

故障への頑健性を検証するために，ゲーム中にエージェントの数を減らし，勝率の変化を検証する．ゲーム設定は実験1と同様に設定し，400 ゲーム行う．コロニー1とコロニー4を200ゲーム対戦させた後，突然コロニー1を構成する蟻の1匹の二つの基本能力をどちらも 0.0 にし，コロニー2状態にし，さらに 200 ゲーム，コロニー4と対戦させた．この“蟻の死亡”によってコロニー1は獲得していた協調動作を大幅に変更しなければならないことが予想される．

つまり，蟻の死亡によって，それまでに獲得した協調動作が正確に再現できなくなり，前半では有効な協調動作“持久戦”が後半になると無力化する，といったことが発生するはずである．コロニー1の勝率の変化を表したものが**図11**のグラフである．縦軸に勝率，横軸にゲーム数をとった，200 ゲーム時点で急激な勝率の低下をみせるが，250 ゲーム程度で安定状態に移行しほぼコロニー2と等しい勝率になることより，きわめて柔軟に協調動作を変更していることがわかる．

6.3 コロニー内での役割分担

次に協調動作とエージェントの役割を考察するために次のような実験を行った．コロニー5，コロニー6はそれぞれ4匹の蟻から構成されている．コロニー5が含む蟻は，学習係数はリワード，ペナルティが 15.0，0.1 で学習を行い，一方，コロニー6の蟻は，それぞれリワード，ペナルティが 14.0，0.1 で出力確率ベクトルを更新する．蟻に与える基本能力はコロニー5，6共に等しい上限と下限を設定し，確率的に定めた．これにより，コロニーレベルで等しい基本能力をもちながら，能力に分散がある集団をつくった．

協調動作を生成する上で，集団内での各蟻の位置づけの違いによって，受け持つ動作がことなることが予想されることは前節でも議論したが，集団内での位置づけを明確に実現するために基本能力に差を持たせるわけである．蟻の能力の最大値は最小値の2倍程度のひらきを与えた．

ゲーム設定は実験1で用いたものをそのまま使用し，コロニー5とコロニー6が対戦させた．提案したゲームが繰り返し囚人のジレンマ問題の性質[34),35)]を持つことが予想できるので，双方の勝率がほぼ 50% になるまで対戦させた．このとき，双方のコロニーは相手に対し戦略的に安定状態にあることが予想され，コロニー内での上記で述べた役割に応じた協調動作を各蟻が獲得していることが推測される．そこで同一コロニーに含まれる二つの蟻の行動の差を，SLA の出力選択確率ベクトルの差から次式に従って評価した．

$$difference^{i}_{x\ and\ y}=\sum_{l}^{5} ABS(P^{i}_{xkl}-P^{i}_{ykl}) \quad (31)$$

ここで x, y はコロニー i の二匹の蟻を表す．

図12は横軸が SLA の各入力状態に対応し，縦軸が前式の値である．入力状態によって明らかな差が生まれており，蟻の能力の違いによって協調動作を生成する上で役割分担が行われていることが推測できる．

6.4 敵コロニーの進化

次に相手の基本能力が次第に向上してゆく場合への学習の効果について検証する．コロニー1を自己を相手に200ゲーム学習を行わせ，協調動作を獲得させた後，オンラインの学習を行うものと行わないものの2種を用意し，コロニー4と 2000 ゲーム対戦させた．一方，コロニー4は 20 ゲームごとに基本能力が1%

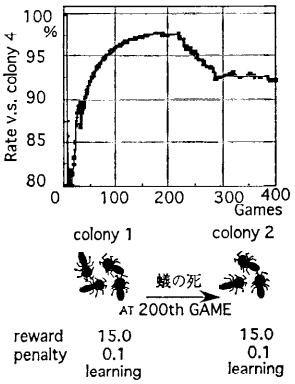
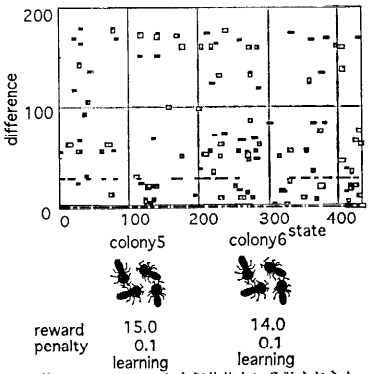


図11 蟻の死亡
Fig. 11 The death of ANT.



注 コロニー5, 6とも個体能力に分散を与えた

図12 コロニー内での役割の相異
Fig. 12 Emergent of roles.

ずつ向上する．つまり2000ゲーム時では，コロニー4の基本能力は0ゲーム時の2倍になっている．

$$Rn_j^{colony4}(aTrial)=Rn_j^{colony4}(0)\cdot\left(1+\frac{RoundDown}{100}\left(\frac{aTrial}{20}\right)\right)$$

$$Tr_j^{colony4}(aTrial)=Tr_j^{colony4}(0)\cdot\left(1+\frac{RoundDown}{100}\left(\frac{aTrial}{20}\right)\right) \quad (32)$$

ここで $aTrial$ は試行ゲーム回数で，j は任意である．

図 13 はコロニー4の能力の上がり具合をパーセントで横軸に，縦軸にコロニー1の勝率をとった．実線がオンラインの学習なし，破線がオンラインの学習を行ったコロニー1の勝率を表している．グラフを比較する上では，学習が有効に働いているとは思えない．このことからSLAの収束よりも頻繁に変化する環境では，学習の効果が得られにくいことが確認できた．

6.5 学習係数による獲得協調動作の相異

次に対戦相手によって獲得される協調動作が異なることを検証する．学習係数の異なる4蟻からなるコロニーを四つ用意し，Nor 1, Nor 2, Nor 3, Nor 4 と名付けた．**図 14** は自身と1000ゲーム学習を行いながら対戦させた後，学習を行わずコロニー4と100ゲーム対戦させたときの勝率を示したものである．**図 15** は Nor 4 と学習を行わず100ゲーム対戦した Nor 1, Nor 2, Nor 3 の勝率である．コロニー4との勝率をみるとほぼ100%であることから，各コロニーが同様な有効な協調動作を獲得していることが想像できるが，相互に対戦させた図 15 の結果より，獲得している協調動作が異なることが推測される．

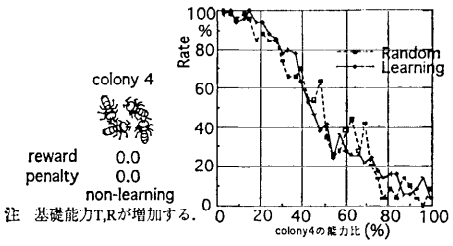


図 13 敵コロニーの進化
Fig. 13 Evolution of opponent.

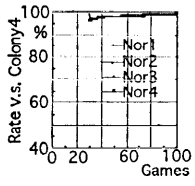


図 14 学習係数の相異1
Fig. 14 Influences of parameters (1).

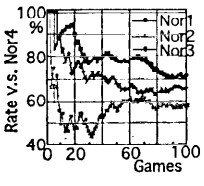


図 15 学習係数の相異2
Fig. 15 Influences of parameters (2).

6.6 評価関数による獲得協調動作の相異

最後に SLA の評価関数である戦況値の導出方法が異なる蟻によって構成されたコロニーの協調動作を検証する．

4匹の蟻から構成されたコロニーを四つ用意し，M 21, M 22, M 23, M 24 と名付けた．各コロニーは戦況値の導出方法が逆，すなわち属するコロニーの戦況が悪化することを良しとする“裏切り者”を程度の差こそあれ2匹含んでいる．学習パラメータはすべて等しい．**図 16** は1000ゲーム自身と対戦させ学習を行った後，コロニー4と学習を行わず100ゲーム対戦させた結果である．**図 17** は Nor 4 と100ゲーム学習を行いながら対戦させた Nor 4 の勝率である．Nor 4 が戸惑いながらも学習を行い，新たな協調動作を獲得していることがわかる．また戦況値の差によっても獲得している協調動作が異なることがわかった．

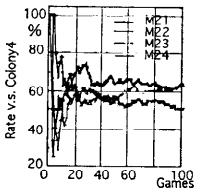


図 16 評価関数の相異1
Fig. 16 Influences of evaluation functions (1).

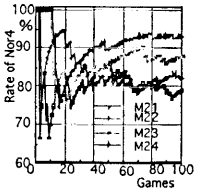


図 17 評価関数の相異2
Fig. 17 Influences of evaluation functions (2).

6.7 ま と め

以上をまとめると，本学習エージェントは観測情報

に暗に含まれる他のエージェントの状況を把握し，エージェントの故障時に適応できることがわかった．また微妙な能力の違いと乱数系列によって異なる振る舞いをするエージェントが発現する場合があることがわかった．また対戦相手によって獲得される協調動作が異なることが推測できた．また評価関数や学習パラメータの違いによって，定常化した際の勝率が異なり，トータルなパフォーマンスへの影響が窺えた．この際，例えば自身との対戦で勝率はほぼ 50% に落ち着くが，勝率が安定した後でも，協調動作の獲得は行われていると考えられる．ただし，ランダム戦略コロニーが生成する戦略を多様性の極致と考えると，学習は相手が生成する戦略への特殊化と見なすことができ，相手が固定的な戦略生成やローカルミニマに陥った際にはさらなる協調動作の獲得は困難になると思われる．このような勝率が安定なときにどのような協調動作が獲得されているのかということについては本手法では明確にすることができず，今後の課題である．

7. 結　　言

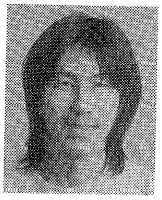
動的環境に適応し問題解決を行えるマルチエージェントシステムの構築を目的とし，エージェントによる学習による協調関係の獲得と，蟻の餌争奪ゲームを用いたシステム評価を行った．さらに計算機実験より，さまざまな問題変化への適応性等の有効性を確かめた．今後の課題として，観測結果と行動の関係，また集団内での役割の発現，学習環境に関して，理論展開を行う必要があると思われる．

参 考 文 献

1) Isida, T. and Kuwabara, K.: Distributed Artificial Intelligence (1): Cooperative Problem Solving, *Journal of Japanese Society for Artificial Intelligence*, Vol. 7, No. 6, pp. 13-22 (1992).
2) Kuwabara, K. and Isida, T.: Distributed Artificial Intelligence (2): Negotiation and Balancing, *Journal of Japanese Society for Artificial Intelligence*, Vol. 8, No. 1, pp. 17-251 (1993).
3) Numaoka, C.: Teamwork of Multiple Autonomous Robots, *Proceedings of ROBOMEC '92*, pp. 69-72.
4) Tan, M.: Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents, *Proceedings of the Tenth International Conference*, pp. 330-337, Morgan Kaufman Pub. (1993).
5) Rumelhart, D.E. and McClelland, J.L. and the PDP Research Group: *Parallel Distributed Processing*, The MIT Press, Cambridge, Massachusetts (1986).
6) Yano, H., Takemiya H., Nunokawa, H. and Noguchi, S.: Autonomous Decentralized Cooperative Computation Model: Kemari, *Trans. IPS Japan*, Vol. 33, No. 12, pp. 1476-1486 (1992).
7) Asama, H.: Distributed Autonomous Robotic System Configurated with Multiple Agents and Its Cooperative Behaviors, *JSPE*, Vol. 57, No. 12, pp. 2117-2122 (1991).
8) Sawaragi, T.: マルチエージェントによる組織的問題解決, *Computer Today*, No. 53, pp. 17-24, サイエンス社 (1993).
9) Dawkins, R.: *The Extended Phenotype*, Oxford University Press, Oxford (1989).
10) Dawkins, R.: *The Selfish Gene*, Oxford University Press, Oxford (1976).
11) Tesauro, G.: TD-Gammon, A Self-teaching Backgammon Program, Achieves Master-Level Play, *AAAI Fall Symposium Series Games*, pp. 19-23 (1993).
12) Sutton, R.S. and Chapman, D.: Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming, *Proceedings of the Seventh International Workshop on Machine Learning*, pp. 216-224 (1990).
13) Levinson, R.: Exploiting the Physics of State-Space Search, *AAAI Fall Symposium Series Games*, pp. 157-165 (1993).
14) 滝沢武信，ほか：将棋計算機対戦，bit, Vol. 14, No. 6, pp. 727-733 (1981).
15) 野下浩平：詰将棋を解くアルゴリズムについて，電子情報通信学会研究報告，COMP 91-56 (1991).
16) Newborn, M.: Cray Blitz: Still the World's Computer Chess Champion, *Abacus*, Vol. 4, No. 3, U.S.A. (1987).
17) Colorni, A., Dorigo, M. and Maniezzo, V.: An Investigation of Some Properties of an "Ant Algorithm", *Parallel Problem Solving from Nature*, 2, Männer, R. and Manderick, B. (eds.), pp. 509-520, Elsevier, North-Holland (1992).
18) Colorni, A., Dorigo, M. and Maniezzo, V.: Distributed Optimization by Ant Colonies, *Proceedings First European Conference on Artificial Life*, pp. 134-142, The MIT Press, England (1991).
19) Collins, R.J. and Jefferson, D.R.: AntFarm; Towards Simulated Evolution, *Artificial Life 2*, SFI Studies in the Sciences of Complexity, Vol. X, pp. 579-601, Addison-Wesley, MA

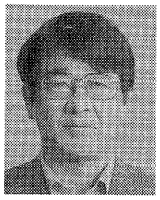
(1991).
20) Theraulaz, G. and Goss, S.: Task Differentiation in Polistes Wasp Colonies: A Model for Self-Organizing Groups of Robots, Meyer, J. A. and Wilson, S. W. (eds.), *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animates*, pp. 346-355, The MIT Press/Elsevier, England (1991).
21) Hieb, M., Hille, D. and Tecuci, G.: Designing a Computer Opponent for Wargames: Integrating Planning, Knowledge Acquisition and Learning in WARGLES, *AAAI Fall Symposium Series Games*, pp. 10-18 (1993).
22) Blair, J. R. S., Mutchler, D. and Liu, C.: Games with Imperfect Information, *AAAI Fall Symposium Series Games*, pp. 57-67 (1993).
23) Melax, S.: New Approaches to Moving Target Search, *AAAI Fall Symposium Series Games*, pp. 30-38 (1993).
24) Holland, J. H.: *Adaptation in Natural and Artificial Systems*, 2nd Ed., MIT Press, Cambridge, Massachusetts (1992).
25) Lin, L. J.: Reinforcement Learning with Hidden States, *Animal to Animat 2*, Meyer, Roitblat and Wilson (eds.), pp. 271-280, MIT Press, London (1992).
26) Goldberg, D. E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA (1989).
27) Naredra, K. and Thathachar, M. A. L.: *Learning Automata an Introduction*, Prentice Hall, A Division of Simon & Schuster, Englewood Cliffs, NJ (1989).
28) Bando, T., Mikami, S. and Kakazu, Y.: An Application to Machine Control by Stochastic Learning Automata—Swing-up Control of an Inverted Pendulum—, *JSPE Proceedings 92 in Hokkaido*, pp. 111-112.
29) 三上，嘉数：確率的ルール学習による動的タスク割り当問題へのアプローチ，日本機械学会論文集C，Vol. 58, No. 551, pp. 2276-2285 (1992).
30) Mikami, S. and Kakazu, Y.: A Stochastic Rule Based Learning Approach to Task Planning for Multiple Robot Environments, *JSME ROBOMEC 92 Proceedings* (1992).
31) Lesser, V. R. and Corkill, D. D.: Functionally-Accurate, Cooperative Distributed Systems, *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 11, No. 1, pp. 81-96 (1981).
32) Smith, R. G. and Davis, R.: Frameworks for Cooperation in Distributed Problem Solving, *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 11, No. 1, pp. 61-70 (1981).
33) Hobby Japan, 空戦マッハの戦い (1983).
34) 小野典彦：GA と人工生物の協調，計測自動制御学会誌，Vol. 32, No. 1, pp. 69-75 (1993).
35) Beasley, J. D.: *The Mathematics of Games*, Oxford University Press, Oxford (1989).

(平成5年12月 8 日受付)
(平成6年 4 月21日採録)



久保 正男（正会員）

1969 年生まれ．1991 年北海道大学工学部精密工学科卒業．1993 年同情報工学研究科修士課程修了．現在同博士課程在学中．知識表現，学習，記憶に興味をもち，現在，俳句，空手，アメリカンフットボールのモデル化に取り組んでいる．



嘉数 侑昇（正会員）

1973 年北海道大学大学院工学研究科博士課程修了．旭川工業高等専門学校，北海道大学工学部精密工学科助教授を経て1987年より同教授．組合せ問題，CAD/CAM, CG, ロボッティクス，AI, GA, AL, 強化学習などの研究に従事．日本ロボット学会，日本機械学会，ORSA，精密工学会，計測自動制御学会，人工知能学会などの会員．