免疫型進化手法を用いた遺伝的ネットワ - クプログラミング によるエ - ジェント学習

正 員 伊藤 宏隆*   非会員 間瀬 友裕**
非会員 岩堀 祐之***

Agent Learning Using Immune Evolved Genetic Network Programming

Hiroataka Itoh*, Member, Tomohiro Mase**, Non-member, Yuji Iwahori***, Non-member

Genetic Network Programming(GNP) is extension of Genetic Algorithm(GA). The GNP is suitable for an agent programming. The GNP can evolve an operation program of the agent. But, the GNP has the premature convergence problem as an evolution technique as well as GA. On the other hand, to avoid the initial convergence of the GA, Immune Algorithm(IA) which is introduced the immune suppression to the GA had developed.

Then if the GNP and IA are combined, a better algorithm for agent programming can be developed. Authors proposed Immune evolved Genetic Network Programming (IGNP). To compare the GNP and IGNP, the simulation was done. As a result, IGNP is more excellent without the premature convergence than the GNP.

In this paper, the authors explain the IGNP outline and a effectiveness of the IGNP is stated through the simulation result.

キーワード : IGNP, GNP, 免疫系, エージェント学習

Keywords: IGNP, GNP, Immune system, learning agent

1. はじめに

遺伝的アルゴリズム (Genetic Algorithm:GA)⁽¹⁾ を代表とする進化論的計算手法は主に、組合せ最適化問題の解法として用いられ、様々な応用において成果を挙げて来ている。

GA は適用される問題に応じて改良されている。ロボットやエージェントプログラミングの分野における GA の改良型アルゴリズムが遺伝的プログラミング (Genetic Programming:GP)⁽²⁾⁽³⁾ である。GP は、GA の遺伝子構造を木構造に拡張し、ロボットプログラム向きの構造となっ

ている。しかし、GP には、交叉などを繰り返すことで木のブロー (膨張) を起こし、幅広い空間の効果的な探索が困難であるという欠点がある。

そこで、遺伝子構造をネットワーク状にした Genetic Network Programming(GNP)⁽⁴⁾⁻⁽⁷⁾ が考案されている。GNP は、GP よりも優れていることが報告されている。しかし、GNP は、構造はネットワーク構造を用いているものの、進化手法は GA と同じであり、GA のもつ初期収束問題を GNP も持っており、場合によっては進化アルゴリズムとしての探索効率が低くなる恐れがある。初期収束問題とは、進化の初期段階において他より明らかに優れた個体が突発的に生まれた場合、解がその個体に収束し、局所解に陥ることで、探索効率が低下することである。

GA における初期収束問題を回避するため、免疫系のしくみを取り入れた進化型アルゴリズムとして、免疫型アルゴリズム (Immune Algorithm:IA)⁽⁸⁾ が提案されている。IA は、(1) 進化過程で得られた優良個体に類似した個体を抑制、(2) 個体群の中で、自分とよく似た個体が多く存在する個体を抑制、という 2 種類の抑制処理を行うことで、解の多様性を維持し、幅広く複数の準最適解候補を探索しながら進化を行うことができ、初期収束問題を回避すること

* 名古屋工業大学 情報メディア教育センター

〒 466-8555 名古屋市昭和区御器所町

Center for Information and Media Studies, Nagoya Institute of Technology

Gokiso-cho, Showa-ku, Nagoya 466-8555

** キヤノン株式会社 宇都宮光学機器事業所

〒 321-3292 栃木県宇都宮市清原工業団地 20-2

Utsunomiya Optical Products Plant, Canon

20-2, Kiyohara-Kogyodanchi, Utsunomiya, Tochigi 321-3292

*** 中部大学工学部情報工学科

〒 487-8501 愛知県春日井市松本町 1200

Department of Computer Science, Chubu University

1200 Matsumoto, Kasugai, Aichi 487-8501

ができる。

そこで、GNP と IA を組み合わせた免疫進化手法を用いた遺伝的ネットワークプログラミング (Immune evolved Genetic Network Programming: (IGNP) ^{(9) (10)} を提案する。

IGNP は、構造は GNP と同じネットワーク構造となっており、進化は IA を基に行う。

IGNP の性能を検証するため、タイルワールドを用いてシミュレーション実験を行った。まず、単一の学習パターンを用いてエージェントのプログラムを進化的に自動生成させる実験を行った。実験結果より、IGNP では初期収束を起こさず、効率の良い学習が行われていた。次に、エージェントの汎化能力を検証させるための 15 個の学習パターンを用いた学習を行っている。その結果から、IGNP は GNP よりも汎化性に優れたエージェントプログラムを作成できることを確認した。

本論文では、まず、IGNP の概要を説明する。次に、タイルワールドを用いたシミュレーション実験により、GNP と IGNP の比較を行う。そして、最後にまとめにより、IGNP の有効性を論述する。

2. IGNP

ロボットやエージェントのプログラムを自動的に生成、進化させることができるアルゴリズムに GNP がある。しかし、GNP は GA と同じ進化アルゴリズムを使用しており、初期収束問題によって解の成長が妨げられる可能性がある。

一方、GA に免疫的進化手法を採り入れた IA は、GA の持つ初期収束問題を回避することができる。

そこで、エージェントのプログラムとして優れた構造を持つ GNP と、解の広域探索・多様性維持に優れた IA を組み合わせることにより、両者の長所を併せ持つアルゴリズムが実現できると考えられ、GNP と IA を組み合わせたアルゴリズムである免疫型進化手法を用いた遺伝的ネットワークプログラミング (Immune evolved Genetic Network Programming:IGNP) を提案する。

2・1 IGNP の構造 IGNP は GNP と同じネットワーク構造であり、その構造に適した交叉・突然変移方法も GNP と同じ方法を用いる。それらの手法については文献 (5) を参照。

2・2 IGNP の進化手法 IGNP の進化の流れを図 1 に示す。IGNP の進化手法は免疫系の個体調節機構を取り入れている。

IGNP の進化は以下のように行われる。

(1) 初期集団の生成

初期個体群をランダムに生成する。

(2) 適応度の計算

各個体毎に適応度を計算する。適応度には得点を用いている。得点とは、エージェントがタスクをどのくらい達成できるかを点数化したものである。

(3) 抑制処理

次に、IGNP では、個体群の抑制処理が行われる。抑制

図 1 IGNP の進化の流れ

Fig. 1. The flow chart of IGNP.

処理には次の 2 種類のものがある。

- 抑制細胞による抑制
- 濃度による抑制

抑制処理に関しては 2・3 で詳しく説明する。

(4) 選択

個体の適応度と (3) で求める濃度から下式によって各個体の期待値を求め、期待値の低い個体を消滅させる。

$$\text{期待値} = \frac{\text{適応度}}{\text{濃度}} \dots\dots\dots(1)$$

(5) 遺伝的操作

処理 (3)(4) によって消滅させられた個体の代わりに、新しい個体を遺伝子をランダムに配置することにより作成する。新しく作成された個体と、これまでの処理で残った個体との間で交叉を行う。また、これまでに残った個体を用いて突然変異を行う。この処理で、(3)(4) において消滅させられた個体を補完し、次世代の個体群を作成する。

(6) 終了条件判定

決められた世代数を消化するまで (2)~(5) の処理を繰り返す。

2・3 抑制処理について IGNP では 2 種類の抑制操作を行っている。ここでは、その抑制処理について詳しく説明する。

2・3・1 抑制細胞による抑制 図 2 に抑制細胞による抑制を示す。抑制細胞による抑制をするため、抑制細胞のデータ領域を M 個分用意する。各世代で最も適応度の高

図 2 抑制細胞による抑制処理

Fig. 2. The suppression by suppressor cell.

図 3 濃度による抑制処理

Fig. 3. The suppression by density.

い個体データを抑制細胞にコピーして格納する(図 2 上)。個体群の各個体と各抑制細胞との類似度を調べ、閾値以下の個体を消滅させる。(図 2 下)。これにより、ある優秀な 1 個体が重点的に進化対象とされるのを防ぐ。ただし、適応度が上位 5% 以内の個体は消滅の対象外とする。

抑制細胞に格納できる個体データは M 個であり、初期状態では抑制細胞に格納されているデータは 0 個である。したがって、 M 世代までは、1 個ずつ最も適応度の高い個体が格納される。 M 世代目で抑制細胞は満席となるため、 $M+1$ 世代以降は、 M 個の抑制細胞と抑制細胞に追加される個体を比較し、追加個体と最も類似している個体を削除し、追加個体を抑制細胞に入れる。抑制細胞は次世代に引き継がれていくが、寿命があり、抑制細胞に登録されてから一定世代経過した個体は消滅させる。

2.3.2 濃度による抑制 濃度による抑制を図 3 に示す。個体群の各個体に対し、濃度を設定する。ある個体の濃度は、個体群に自分と似た個体が多い個体ほど高くなり、似た個体が少ないほど低くなるパラメータである。図 3 に示すように濃度がある程度高い、つまり、類似した個体が多く存在する個体を消滅させることで、個体群の多様性を維持させ、より広域的な解の探索が可能となる。ただし、有効な個体を消滅させないために、個体群中の適応度上位 5% の個体に関しては、消滅の対象外としている。

2.4 個体間の類似度計算法 IGNP では、2 種類の抑制処理を行う際に個体間の類似度を判定している。GA のような一列から成る遺伝子データであれば、2 個体間の

図 4 2 個体間の類似度判定

Fig. 4. The decision of similarity between two individual.

ハミング距離(遺伝子同士の異なる部分の数)を取れば、2 個体がどれだけ似ているか判別できる。しかし、1 個体がネットワーク形状をしている IGNP では、個体間のハミング距離で類似度を表すことはできない。そこで、次のような方法で 2 個体間の類似度を計算している。

- (1) 図 4 に示す BODY 1, BODY 2 の 2 個体の類似度を考える。は処理ノード、は判定ノード、ノード内の英字はそれぞれのノードの種類を表す。
- (2) BODY 1, BODY 2 のそれぞれについて、類似度判定用テーブルを作成する。TABLE 1 は BODY 1, TABLE 2 は BODY 2 から作成されたテーブルである。これらのテーブルはそれぞれ、どの種類のノードから、どの種類のノードに何本枝が伸びているか、を数字で表している。
- (3) 2 個体間の類似度は、それぞれの個体のテーブルの差を取ったものから計算する。TABLE 1 と TABLE 2 の差を取ったものが図 4 の DIFFERENCE TABLE である。この DIFFERENCE TABLE 内の数字の数こそが、どれだけ 2 個体間で違いがあるか、を表している。DIFFERENCE TABLE の各行を見れば、ノード種類毎に 2 個体間でどれだけの違いがあるかがわかる。

個体 a, b の類似度 $H_{a,b}$ を考えた場合、2 個体間の類似度は、DIFFERENCE TABLE を元に、次の計算式で数値化される。

$$H_{a,b} = \sum_k H_k \dots\dots\dots (2)$$

$$H_k = \frac{X_k}{(P_{ak} + P_{bk}) \times N_k} \dots\dots\dots (3)$$

但し、 k はノードの種類、 N_k はノード k から伸びている枝の数、 P_{ak}, P_{bk} はそれぞれネットワーク a, b に存在するノード k の数、 X_k は DIFFERENCE TABLE のノード k の行に存在する数字の和を示している。つまり、2 個体間で各ノード毎の類似度を出し、その和が 2 個体間の類似度となる。式 (3) の値は、2 個体で差がなければ 0、2 個体が全く似ていなければ 1 となる。つまり、式 (2) の値は $0 \leq H_{a,b} \leq 1$ となり、この値が 1 に近いほど 2 個体は類似せず、0 に近いほど 2 個体は類似しているということになる。抑制細胞による抑制処理の際は、各個体と各抑制細胞の間で式 (2) を計算し、ある閾値を下回る個体を消滅させる。

濃度による類似度判定の場合は、式 (4) を用いる。ある個体 a の濃度 c_a は、

$$c_a = \frac{1}{1 + \frac{\sum_{n=1}^N H_{a,n}}{N}} \quad (a \neq n) \dots\dots\dots (4)$$

である。但し N は全個体数を表す。

式 (4) より各個体の個体群全体に対する濃度を計算し、濃度と適応度から式 (1) により、各個体の期待値を求める。個体群中で、期待値が低い個体、つまり、適応度が低く、かつ、濃度の高い個体を消滅させる。

3. シミュレーション環境

提案した IGNP と GNP の性能を比較検証するため、シミュレーション実験を行った。ここでは、シミュレーションに用いた環境を説明する。

3.1 タイルワールド 図 5 のようなタイルワールドを考える。タイルワールド上には、タイルと穴が各 2 個ずつ配置されている。このフィールド上を動作するエージェントを考え、エージェントの動作を制御するプログラムを GNP、IGNP で学習させることにより得る。エージェントには方向があり、進行方向にしか進むことはできない。エージェントから見て右側に進みたい場合は「右 90° 回転」という行動を行ったあと、前進することで実現する。穴と障害物は配置された位置から動くことはない。エージェントの目的は、フィールド上のタイルを全て穴に落とすことである。

GNP、IGNP では、各個体がネットワーク形状をしており、それがエージェントプログラムを表す。各個体の得点は、各ネットワークで表されるエージェントプログラムによって、エージェントがどのような行動を取るかによって決定される。

3.2 エージェントの得点 適応度となる得点を説明する。エージェントは次の 3 つの状況で得点を獲得することができる。

図 5 タイルワールド

Fig. 5. Tile world.

(1) タイルを穴に落とす

穴に落としたタイルの枚数に応じた得点が得られる。

(2) タイルを穴に近づける

タイルを穴から遠ざけてしまった場合は得点がマイナスになる。

(3) 目的達成時の残り時間

これはタイルを全て穴に落とすことができた場合にのみ得られる。

よって、得点の計算式は式 (5) のようになる。

$$\begin{aligned} fitness = & C_{tile} \times DropTile \\ & + C_{dist} \times \sum_{t \in TN} (InitialDist_t - LastDist_t) \\ & + C_{step} \times RestStep \dots\dots\dots (5) \end{aligned}$$

但し、 $DropTile$ はエージェントが制限時間内に穴に落とすことができたタイルの枚数を表し、 $InitialDist_t$ は初期状態におけるタイル t と穴の最短距離、 $LastDist_t$ は最終状態におけるタイル t と穴の最短距離を表す。 TN はタイルの枚数であり、本実験では $|TN| = 21$ である。 $RestStep$ は制限時間内に全てのタイルを穴に落とすことができたときの、残り時間を表す。つまり、第 1 項はエージェントがどれだけタイルを落とすことができたか、第 2 項はどれだけタイルを穴に近づけることができたか、第 3 項はどれだけ早く目的を達成することができたかを評価する。 $C_{tile}, C_{dist}, C_{step}$ はそれぞれの項にかかる重み係数である。本実験では、式 (5) で得られた得点に対して、獲得できる最大得点が 100 点となるように正規化している。

エージェントの動作関数には表 1 に示す合計 9 種類の処理・判定ノードを使用した。

4. シミュレーション実験

4.1 実験環境 実験は、Pentium4、メモリ 1GB の Linux PC 上で C 言語を用いてプログラミングし、シミュレーションを行った。

4.2 1 初期配置パターンを用いた学習 図 5 のフィールド上を動作するエージェントのプログラムを、GNP、IGNP を用いて進化させ、自動生成を行った。実験パラメータを表 2 に示す。IGNP、GNP 共通のパラメータとして、個体数 200 個、世代数 1000、制限時間 100、IGNP 固有の

表 1 エージェントの処理・判定ノード

Table 1. The processing and judgement node of agent.

type	function	action
processing	Advance	Go forward
	TurnLeft	Turn to the left
	TurnRight	Turn to the right
judgement	L-Forward	What is there forward?
	L-Left	What is there toward the left?
	L-Right	What is there toward the right?
	T-Sensor	What does agent touch?
	S-Object	Which direction is the object which agent looks for?
	IfBumped	What did agent bump before step?

表 2 学習パラメータ

Table 2. The parameter of learning.

population size	200
generation number	1000
limit time	100
suppression cell number	20

パラメータとして抑制細胞数を 20 個としている。なお、シミュレーションは IGNP, GNP の各手法で各 10 回試行を行った。

GNP による学習結果を図 6 に、IGNP による学習結果を図 7 に示す。図 6 と図 7 は 10 回の各試行での最良個体の適応度の推移を表している。図 6 と図 7 において、目的達成ラインが 88 点付近にあるが、このラインを越えると、エージェントが目的タスクを達成できたこととなる。すなわち、タイルを 2 個穴に落とすことができたことを表している。図 6 の GNP の学習結果では、最終世代までに、目的達成ができていない試行が 4 回あり、これは初期収束によるものであると考えられる。また、300 世代までに目的達成ラインを越えた試行は 3 回であり、残りの 3 回の試行は 500 世代以降に目的達成ラインを越えている。

一方、図 7 の IGNP の学習結果では、最終世代までに 9 回の試行で目的達成ラインを越えており、そのうち 7 回までは 300 世代で達成できている。IGNP では、個体の抑制処理によって、学習効率が GNP よりも良くなっていることがわかる。

図 8 に GNP, IGNP の各試行における世代別平均適応度と最良適応度の世代別推移を表している。10 回の試行それぞれにおいて、世代別の最良適応度のグラフを作成し、そのグラフの 10 回分の平均を平均適応度のグラフで、10 個のグラフのうち最終的に最も高い適応度を獲得していたグラフを最良適応度のグラフで表している。図 8 において、GNP と IGNP の最良適応度のグラフを比較すると、IGNP のグラフの方が、最高適応度 100 点に速く到達できている。平均のグラフにおいても IGNP は GNP よりも高い平均適応度推移を示している。

初期収束の有無を示すため、図 9 に多様性の推移を示す。図 9 の多様性は個体群の個体の濃度を平均したものであり、

図 6 GNP による学習結果

Fig. 6. The result by GNP.

図 7 IGNP による学習結果

Fig. 7. The result of IGNP.

図 8 学習結果

Fig. 8. The result.

図 9 多様性の推移

Fig. 9. The transition of diversity.

図 10 初期配置パターン

Fig. 10. The pattern of initial positions.

0に近いほど多様性があり, 1に近いほど多様性が失われていることを示す。GNP は学習開始後から多様性が失われていき, IGNP は多様性維持機構により, 多様性が維持されていることが確認できる。

シミュレーションの実行時間は 10 回の試行の平均で, GNP が約 33 秒, IGNP は約 68 秒であった。IGNP は抑制処理部分で計算コストがかかるため, GNP の約 2 倍程の計算時間がかかっているものの, 十分実用的な実行時間であり, 図 6 ~ 8 に示すように明白な効果があるため, IGNP は GNP よりも有効である。

次に, エージェントプログラムの汎用性を調べるためのシミュレーション実験を行った。図 5 で学習して獲得されたエージェントプログラムを, エージェントとタイルの初期配置を変えた 3 種類の配置パターン (図 10) に適用して実行させてみた。その結果を表 3 に示す。IGNP, GNP の各々の手法で獲得されたエージェントプログラムでは, どちらも良い結果を得られなかった。1 種類の初期配置パターンを用いて獲得されたエージェントプログラムは学習に用いたパターン以外の初期配置では有効に動作できない。

4.3 複数の初期配置パターンを用いた学習 そこで, エージェントとタイルの初期配置の複数パターンを, 学習に用いることで, 汎化性に優れたエージェントプログラムが作成可能であるかどうか調べる。図 11 に示す 15 種類の初期配置パターンを学習に用いて, エージェントの動作

表 3 実行結果

Table 3. The result.

	A		B		C	
	max	average	max	average	max	average
IGNP	34.80	7.21	9.23	5.84	41.66	10.95
GNP	12.63	4.33	10.41	3.56	34.02	9.49

図 11 学習用初期配置パターン

Fig. 11. The patterns of initial position for learning.

プログラムを作成した。学習に用いるパターンの増加に伴い, 学習の最大世代数を 10000 世代として学習を行った。

学習結果を図 12 に示す。複数の初期配置パターンを学習に用いた場合においても, IGNP は GNP よりも有効な学習を行うことができている。そのことは, 最終世代における IGNP の平均値が GNP の最良個体の適応度をも上回っていることから分かる。

次に, GNP, IGNP における, 図 12 内の最良適応度で表される試行において, 最終世代で最も高い適応度を獲得した個体が学習に用いた図 11 の各初期配置パターンにおいて, どれだけの適応度を獲得しているかを表 4 に示す。太字で表現されている部分は, そのパターンにおいて目的を

図 12 学習結果

Fig.12. The result.

表 4 パターン毎の獲得得点

Table 4. The scores of each pattern.

	IGNP	GNP		IGNP	GNP
1	98.45	46.93	9	97.80	93.47
2	96.58	47.64	10	96.37	95.45
3	92.23	56.74	11	94.40	50.14
4	92.88	90.71	12	98.94	48.99
5	93.35	56.29	13	96.74	97.29
6	90.98	58.33	14	95.13	91.68
7	97.20	61.47	15	97.47	94.67
8	94.84	92.90			

表 5 実行結果

Table 5. result.

	A		B		C	
	max	average	max	average	max	average
IGNP	74.62	44.98	64.44	35.39	76.38	42.00
GNP	67.61	22.09	57.08	26.64	57.08	31.97

達成することができていることを表す。

表 4 より、目的達成パターン数を比較してみると、IGNP では 15 種類全てで目的達成可能なエージェントプログラムを作成できているのに対し、GNP では 7 種類に留まっている。やはり IGNP で学習したエージェントプログラムの方が、より多くの環境に適応することができていることがわかる。

次に、15 種類の初期配置パターンを学習に用いて得られたエージェントプログラムを、図 10 の 3 種類の初期配置パターンにおいて実行させた結果得られた適応度を表 5 に示す。なお、図 10 は学習に用いた 15 種類のパターンには含まれていない。表 3 と表 5 を比較し、実行結果が向上していることがわかる。GNP と IGNP の適応度を比較すると、IGNP で学習したエージェントプログラムの方が優秀な結果を得ることができていることが確認できる。

実行時間は IGNP が約 42 分、GNP が約 48 分であり、ほぼ同じと言える。これは、IGNP の抑制処理にかかる計

算コストよりも、両手法共通の適応度計算のための 15 個の学習フィールドでの実行時間の方が計算コストが非常に高くなっているためである。

5. ま と め

エージェントのプログラムを遺伝的に進化させることができるアルゴリズムである GNP と GA 型進化における問題点である初期収束を回避し、解空間の広域的な探索を行うことができる IA を組み合わせたアルゴリズムである IGNP を提案した。

IGNP の性能評価を行うため、タイルワールドを用いてシミュレーション実験を行った。まず、1 種類の初期配置パターンを学習に用い、エージェントのプログラムの自動生成を行い、GNP と IGNP の性能比較実験を行った。IGNP は多様性維持機構により初期収束せず効率のよい探索が可能であり、GNP よりも早期に目的達成個体を発見できることを確認した。

しかし、1 種類のエージェントとタイルの初期配置パターンのみを用いた学習によって得られたエージェントプログラムは、少しの環境の変化にも対応することができなかった。

そこで、複数の初期配置パターンを学習し、多くの環境に対応することができるエージェントプログラムの獲得実験を行った。実験で得られたエージェントプログラムは、学習に用いた各パターンおよび学習に用いたパターンと異なる配置パターンに対して効率の良い動作を行うことができている。さらに、GNP と IGNP のそれぞれで得られたエージェントプログラムの動作結果の比較から、IGNP によって得られたプログラムの方が GNP よりも汎化性に優れていることがわかった。

実験結果、実行時間より、IGNP は GNP よりもエージェントのプログラムの自動生成手法として有効である。

(平成 16 年 4 月 28 日受付、平成 16 年 11 月 29 日再受付)

文 献

- (1) 北野宏明編：遺伝的アルゴリズム 1-4，産業図書 (1993-2000)
- (2) J.R. Koza: "Evolution of Subsumption Using Genetic Programming", Computer Science Department, Stanford University (1992)
- (3) W. Banzhaf, P. Nordin, R.E. Keller, and F.D. Francone: Genetic Programming - An Introduction, Morgan Kaufmann publishers Inc.
- (4) H. Katagiri, K. Hirasawa, J. Hu, and j. Murata: "Genetic Network Programming and Its Application to the Multi-agent System", *T. IEE Japan*, Vol.122-C, No.12, pp.2149-2156 (2002-12) (in Japanese)
片桐広伸・平澤宏太郎・胡 敬炉・村田純一: "Genetic Network Programming とそのマルチエージェントシステムへの応用", 電学論 C, **122**, 12, pp.2149-2156 (2002-12)
- (5) K. Hirasawa, M. Okubo, H. Katagiri, J. Hu, and j. Murata: "Comparison between Genetic Network Programming and Genetic programming using evolution of ant's behaviors", *T. IEE Japan*, Vol.122-C, No.11, pp.1001-1009 (2001-6) (in Japanese)
平澤宏太郎・大久保雅文・片桐広伸・胡 敬炉・村田純一: 「蟻の行動進化における Genetic Network Programming と Genetic Programming の性能比較」, 電学論 C, **121**, 6, pp.1001-1009

- (2001-6)
- (6) S. Mabu, K. Hirasawa, J. Hu, and J. Muarata: "Online Learning of Genetic Network Programming and its Application to Prisoner's Dilemma Game", *T. IEE Japan*, Vol.123-C, No.3, pp.535-543 (2003-5)
  - (7) K. Hirasawa, M. Okubo, J. Hu, j. Murata, and Y. Matsuya: "Co-evolutin of Hetero Multiagent Systems using Genetic Network Programming", *T. IEE Japan*, Vol.123-C, No.3, pp.544-551 (2003-5) (in Japanese)
平澤宏太郎・大久保雅文・胡 敬炉・村田純一・松家裕子:「Genetic Network Programming によるヘテロマルチエージェントシステムの構成」, 電学論 C, **123**, 3, pp.544-551 (2003-3)
  - (8) K. Mori, M. Tsukiyama, and T. Fukuda: "Application of an immune algorithm to multi-optimaizatin problems", *T.IEE Japan*, Vol.117-C, No.5, pp.593-598 (1997-5) (in Japanese)
森 一之・築山 誠・福田豊生:「免疫アルゴリズムによる多峰性関数最適化」, 電学論 C, **117**, 5, pp.593-598 (1997-5)
  - (9) 間瀬友裕・伊藤宏隆・岩堀祐之:「免疫型進化手法を用いた遺伝的ネットワークプログラミングによるエージェント学習」, 平 15 電気関係学会東海支部連大講演論文集, p.233 (2003)
  - (10) 間瀬友裕・伊藤宏隆・岩堀祐之:「Immune evolved Gnetic Network Programming のエージェント学習における性能評価」, 第 16 回自律分散システムシンポジウム資料, pp.221-226 (2004)

伊 藤 宏 隆 (正員) 1971 年生。1999 年名古屋工業大学大学院工学研究科電気情報工学専攻博士後期課程修了。博士 (工学)。1999 年名古屋工業大学情報処理教育センター助手。2000 年名古屋工業大学情報メディア教育センター助手。現在に至る。進化的計算手法に関する研究に従事。計測自動制御学会会員。

間 瀬 友 裕 (非会員) 2004 年名古屋工業大学大学院工学研究科電気情報工学専攻博士前期課程修了。2004 年キャノン (株) 入社。

岩 堀 祐 之 (非会員) 1959 年生。1988 年東京工業大学大学院電気・電子工学専攻博士課程修了。工学博士。1988 年名古屋工業大学情報処理教育センター助手。1992 年名古屋工業大学情報処理教育センター助教授。2002 年名古屋工業大学情報メディア教育センター教授。2004 年中部大学工学部情報工学科教授。現在に至る。この間 (カナダ) プリティッシュコロンビア大学客員研究員。コンピュータビジョン・パターン認識・知能情報処理に関する研究に従事。電子情報通信学会, 情報処理学会, IEEE 各会員。