2B3-1 幅と深さを考慮したGnenetic Network Programmingの進化手法

Genetic Network Programming considering the evolution of Breadth and Depth

早稲田大学　○江藤慎治，平澤宏太郎，古月 敬之
○ Shinji Eto, Kotaro HIRASAWA and Takayuki FURUZUKI, Waseda University

Abstract: Many methods of generating behavior sequences of agents by evolution have been reported. A new evolutionary computation method named Genetic Network Programming (GNP) has also been developed recently along with these trends. In this paper, a new method for evolving GNP considering Breadth and Depth is proposed. The performance of the proposed method is shown from simulations using garbage collector problem.

Keywords: Evolutionary computation, Genetic Network Programming

1 はじめに

従来より，人工知能やロボティクスの分野においてプラニングに関する様々な研究がなされている [1]．これらの研究は，ある環境下に存在するエージェントが与えられた目標を達成するために必要な行動の系列を生成することに関する．これは，エージェントを生物とみなすと，その生物の脳を設計することであるともいえる．

このような人工脳の実現，すなわちエージェントの行動系列の作成は，従来，設計者によって行われてきた．しかし，エージェントの目的や環境がより複雑になるにつれ，そのような行動系列の設計が困難を極めることは容易に想像できる．さらに，特定の設計者の行動基準に基づいて設計された行動では，エージェントの能力が設計者のスキルに大きく依存するといった問題や，環境変化が起こると与えられた行動系列では対処できないケースが多いといった問題が挙げられる．

このような事から，自律性と創造性に富んだエージェントの行動系列は，設計者によって与えられるよりはむしろ進化的手法により，設計者への依存度を低くし自動的に獲得されることが望ましい．

こうした背景から，エージェントの行動系列の自動設計に関する多くの研究がなされてきた [2]．そのようなエージェントの行動系列を実現する手法に，Genetic Algorithm (GA)，Evolution Strategy (ES)，Evolutionary Programming (EP)，Genetic Programming (GP)，Parallel Algorithm Discovery and Orchestration（PADO）などの進化型計算が多数提案されている [2, 3, 4, 5, 6]．

しかし，これまでの手法では，エージェントが現在置かれている環境の情報を全て用いて今何をするべきかという点を重視した行動系列の生成を試みているものが多い．これに対し，本論文では，現在必要と思われる環境に関する情報のみを必要に応じて用いるといった，現実の脳が行っている機能に近い手法で行動系列の生成を行う．脳は目や耳などの感覚器官から得られる情報の全てを利用して行動を決定するのではなく，必要に応じて情報の取捨選択を行い，それを利用して行動を決定する．そのような必要な情報のみを必要に応じて用いるという脳に近い機能をもち，部分観測マルコフプロセス環境下での行動系列の生成を行うことのできる Genetic Network Programming（GNP）[7, 8, 9] を用いてエージェントの行動系列の自動設計を行う．GNP は単純な機能を持った複数のノードをネットワーク状に接続し，ノード間の遷移規則により行動系列を表現するモデルである．

GNP はグラフ構造をしているため，様々な進化手法が考えられる．従って本論文では，GNP の進化手法を拡張することで GNP の性能を向上させることが目的である．従来の GNP では，スタートノードは 1 度使われると，そこに遷移することはなく 2 度と使用されることはない．しかし，提案手法では，スタートノードを複数（幅）設定し，一定のノードを実行する（深さ）とスタートノードの 1 つに遷移する．本論文では，この幅と深さをも進化で獲得する．シミュレーションより，提案手法によって GNP の性能が向上することを示す．

2 Genetic Network Programming (GNP)[7, 8]

2.1 GNP の構成要素

GNP の基本構成を図 1 に示す．GNP は個体を有向グラフで表現し，各ノードには判定ノード J_m と処理ノード P_n が設定されている．J_m は判定ノードのライブラリに記載されている m 番目の判定を表し，P_n は処理ノードのライブラリに記載されている n 番目の処理を表す．ライブラリには設計者が用意したエージェントの最小な判定・処理単位が記載されており，内容に重複はないものとする．また，S はスタートノードを示し，GNP はこのノードから開始される．

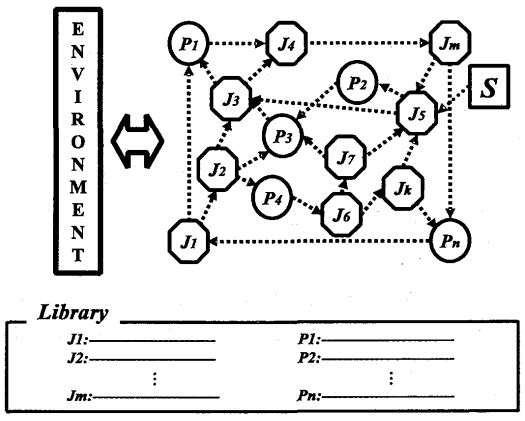

Fig. 1: The basic structure of GNP individual

2.2 GNP の進化

各 GNP 個体のノード数は同一であり，GNP 内の各ノードはユニークな番号を持っている，同一番号のノードの種類，機能は同一とする．また，GNP 内の種類，機能毎のノード数は同一とし，ノード間の初期接続はランダムとする．

本論文での，GNP の進化は，簡単のためにノード遺伝子の接続に関する情報のみを進化させることによって実現する．すなわち，ノード間の接続のみを進化的に変更することによって適切な構造を得ようというものである．

個体の進化のためのオペレータとして，交差は「2 親個体内の対応するノード間接続遺伝子を交差率 P_c で確率的に交換する」オペレータを，また，突然変異は「ノード間接続遺伝子を突然変異率 P_m で確率的に変更する」オペレーターを採用した．

ここで，個体とは GNP プログラムを構成するために必要なノード遺伝子の集合のことを意味する．すなわち，1 個体は複数のノード遺伝子からなる 1 つの有向グラフである．

3 多スタートノード GNP（GNPPS）

本章では GNP の拡張である，多スタートノード GNP（GNPPS）について説明する．

従来の GNP では，1 度スタートノードより GNP の実行が開始されると，2 度とスタートノードに戻ることなく，判定ノードと処理ノードを遷移する．しかしながら，従来手法ではいくつかのノードが使用されず未使用のままで残っているため GNP の機能を十分発揮できないことがある．従って，本論文では，多スタートノード GNP（GNP with plural start nodes, GNPPS）を提案する（図 2）．提案手法は，GNP の構造が複雑で，判定ノードからの接続が多い場合に効果を持つ．実際，問題が複雑であるほど，判定ノードからの接続が多くなる．加えて，GNP の遷移は，主に判定ノードによって決められるため，判定ノードがどのような判定結果を持つかは極めて重要である．そのため，GNP では GNP 内部の多くの判定結果が選択されるべきである．しかしながら，従来の GNP ではスタートノードが 1 個でノード遷移が限定されるため，その能力が乏しい．GNPPS では，一定の数の処理ノードを実行した後に，スタートノードの 1 つに遷移する．複数のスタートノードを設けることで，GNP は内部の多くのノードに遷移できるようになり，GNP の機能を十分に発揮できるようになる．GNP の進化において，スタートノードの数を幅，1 個のスタートノードから実行可能な処理ノードの数を深さとみなすことができる．

本論文では，スタートノードの数（S）と 1 個のスタートノードから実行可能な処理ノードの数（P）を進化で獲得する場合の，GNPPS の進化性能について検討する．

具体的には，以下について検討した．

- S と P を進化で獲得する GNPPS と従来の GNP を，様々なノード数の GNP について比較検討する．GNP のノード数は，各種類のノードにつき 1，3，5，10，20 個とする．また，S*P の値はシミュレーションの最大ステップ数である 250 以下とする．

スタートノードの数（S）と 1 個のスタートノードから実行可能な処理ノードの数（P）の進化については，以下の突然変異オペレーターを，通常の突然変異の代わりに使用する．

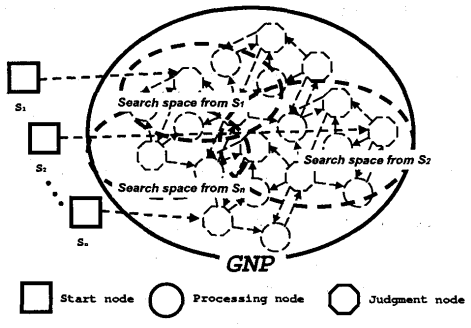

Fig. 2: The basic structure of GNPPS

- 突然変異オペレータ．S と P の値を突然変異確率 P_m でランダムに変更する．S*P の値は 250 以下となるようにする．さらにスタートノードを含む全てのノードの接続を突然変異確率 P_m でランダムに変更する．

交叉オペレータについては，複数のスタートノードの交換を含む通常の交叉を行う．

4 シミュレーション

4.1 ゴミ拾い問題

ゴミ拾い問題とは，環境内のゴミを拾い集めるという問題である．

自己充足的ゴミ拾い問題はチェスボードのような 2 次元格子上に，エージェントと複数のゴミをランダムに配置し，固定の場所にゴミ収集所を配置した環境である（図.3 参照）．エージェントは 1 つのセル上にちょうど収まる大きさで，1 ステップに前方に 1 セル動くか左か右に 90 度回転することができる．エージェントはゴミのあるセルに到達することでゴミを保持し，ゴミを保持した状態で収集所に到達することでゴミの収集を行う．エージェントの保持できるゴミの数は有限である．

Table 1: Parameter conditions for evolving GNP

Generation	1000
Number of GNP	100
Number of elite GNP	1
Number of crossover individuals	40
Number of mutation individuals	59
Crossover probability P_c	0.1
Mutation probability P_m	0.01
Number of nodes per each kind K	1, 3, 5, 10, 20

Table 2: Functions of a start node, processing nodes and judgment nodes

	function
0	start node
1	check the distance from the agent to the collection place (3)
1	check the distance from the agent to the check point (3)
1	check the direction of the agent to the collection place (8)
1	check the direction of the agent to the check point (8)
1	check the direction of the agent to the nearest trash (8)
1	check the direction of the agent to the second nearest trash (9)
2	move forward
2	turn right
2	turn left
2	stay

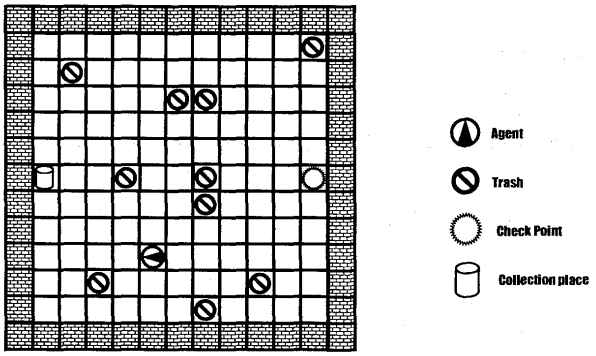

Fig. 3: Garbage Collctor Ploblem

4.2 シミュレーション条件

環境の広さは 11×11 マスで端が壁で囲まれている環境とする．エージェントは環境内を全て見渡すことができる．エージェントは1体，ゴミは10個，ゴミ収集所は1箇所とした．エージェントは250ステップ内で全てのゴミの収集を行うことを目指す．エージェントが一度に保持できるゴミの数は2個までとした．この条件で，エージェントの初期位置および向きとゴミの初期位置をランダムに設定した10環境を用意し，それぞれの環境の得点の合計に応じてGNPを進化させる．

今回のシミュレーションで用いたパラメーターを表1に示す．これらの条件下で，提案手法と従来手法が，どの程度問題に適応できるかについて検討を行った．

4.3 適合度関数

シミュレーションで用いた適合度関数を式（1）と式（2）に示す．GNPの適合度はゴミを集めた個数と全てのゴミを集められなかった場合エージェントに一番近いゴミとの距離によって行う．

$$Fitness = \sum_{n=1}^{10}(100N_n - D_n), \tag{1}$$

$$Fitness = N. \tag{2}$$

ここで n は各世代の環境の番号であり，環境は各世代毎にランダムに10個生成される．N_n は環境 n で収集したゴミの数で，D_n は環境 n で250ステップ後に残っているゴミとエージェント間の距離の最小値である．N は汎化能力テストの時に集めたゴミの数である．式（1）はGNPの進化の時に用いる適合度関数で、式（2）は得られた最良のGNPの汎化能力テストで用いる適合度関数である．

4.4 GNPのノードの設定

以下，今回のシミュレーションで用いる，GNPのノードについて説明する．

4.4.1 スタート／判定／処理ノード

GNP内のスタート／判定／処理ノードとして，表2で示されるノード群を用意した．0がスタートノード，1が判定ノード，2が処理ノードを示す．表中の（）内の数はそれぞれの判定ノードの判定結果の数を示す．．スタートノードは単純にGNPのスタート位置を示している．本論文のGNPPsではスタートノードが複数存在する．

判定ノードでは，それぞれの判定を行いその結果に従って次に実行するノードの選択を行う．「収集所までの距離」，「チェックポイントまでの距離」ではエージェントから収集所またはチェックポイントまでの距離を3段階（0-4, 5-7, 8-）に分け，距離に応じて3分岐を行う．「収集所の方向」，「チェックポイントの方向」ではエージェントから収集所またはチェックポイントの方向を8方向（前，右前，左前，右，左，右後，左後，後）に分け，方向に応じて8分岐する．「最も近いゴミの方向」，「2番目に近いゴミの方向」ではエージェントからゴミの方向を8方向（前，右前，左前，右，左，右後，左後，後）となしに分け，結果に応じて9分岐する．

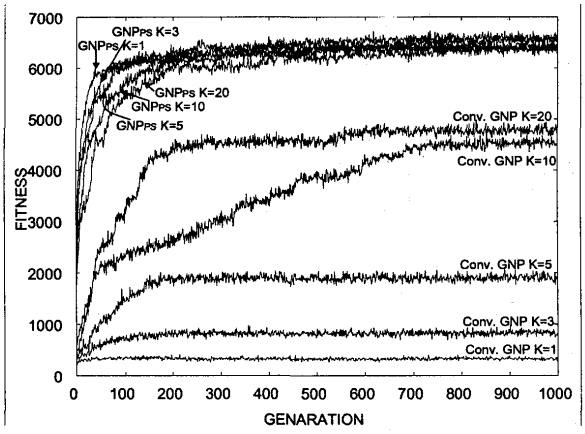

Fig. 4: Average of the best fitness values over ten independent trials

処理ノードは1つの接続先を持ち，エージェントの行動（MF,TR,TL,ST）を行う．

5 シミュレーション結果と考察

10種の異なる乱数系列を用いてシミュレーションを行い最良個体の適合度の平均値で評価を行った．

GNPPs K は提案手法の適合度で，K は種類毎のノードの数である．Conv. GNP K は従来手法の適合度を示す．図4より，提案手法は従来手法より良い結果を示していることがわかる．特に，提案手法ではどの値の K でも良い結果を示すのに対し，従来手法ではある程度 K の値が大きくなければ適合度が高くならないことがわかる．

表3に，10回の独立したシミュレーションで得られた10個の最良個体の汎化能力テストの結果を示す．ここで汎化能力とは学習した環境以外での問題解決能力であり，異なる1000環境における10個の最良個体の得点の最大値（MAX），平均値（AVE），最小値（MIN），標準偏差（STDEV）を示している．表3より，提案手法が従来手法に比べ高い汎化能力を持つことがわかる．汎化能力においても，従来手法が多くのノード数（K：大）を必要とするのに比べ，提案手法では少ないノード数（K：小）であっても問題を解くことができている．

図4と表3より，従来手法では解くことが難しい条件であっても，提案手法では問題を解くことが可能であることがわかる．ただし，提案手法では，スタートノードを複数持つため，スタートノードの数の分だけ接続が増加している．

6 結論

本論文では，多スタートノード GNP（GNPPs）を提案し，従来の GNP との比較を行った.. その結果，提案手法においてスタートノードの数およびスタートノードから実行可能な処理ノードの数を進化によって獲得する場合，提案手法が高い問題適合能力を持つことを示した．また，従来手法と比較し，提案手法が高い能力を有することも示した．ただし，進化で獲得したスタートノードの数およびスタートノードから実行可能な処理ノードの数の関係については今後の課題である．

Table 3: Generalization (Average Fitness of GNP)

	MAX	AVE	MIN	STDEV
GNPPs K=1	6.143	5.707	4.830	0.4152
GNPPs K=3	6.764	5.747	4.866	0.5311
GNPPs K=5	6.197	5.796	4.632	0.4519
GNPPs K=10	6.005	5.642	5.220	0.2482
GNPPs K=20	6.383	5.658	4.591	0.4845
Conv. GNP K=1	0.058	0.028	0.003	0.0186
Conv. GNP K=3	0.229	0.137	0.019	0.0658
Conv. GNP K=5	0.942	0.751	0.554	0.1205
Conv. GNP K=10	4.322	3.466	1.330	0.7940
Conv. GNP K=20	3.969	3.670	3.446	0.1518

References

[1] Brooks, R. A: "Robust layered control system for a mobile robot", IEEE Journal of Robotics and Automation, Vol.2, No.1, pp. 14-23, 1986.

[2] 馬場口登, 山田誠二: 人工知能の基礎, 昭晃堂, 1999.

[3] J. Holland: Adaptation in Naural and Artificial Systems - An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence -, Ann Arbor: University of Michigan Press, 1975.

[4] 伊庭斉志：遺伝的プログラミング入門，東京大学出版会，2001.

[5] J. R. Koza: "Genetic Programming", Cambridge, MA: MIT Press, 1992.

[6] A. Teller and M. Veloso: PADO : Leaning Tree-structured Algorithms for Orchestration into an Object Recognition System, Carnegie Mellon University, Technical Report Library, 1995.

[7] T. Eguchi, K. Hirasawa, J. Hu and N. Ota, "A study of Evolutionary Multiagent Models Based on Symbiosis", IEEE Trans. on System, Man and Cybernetics -Part B-, Vol.35, No.1, pp. 179-193, 2006.

[8] S. Mabu, K. Hirasawa and J. Hu, "A Graph-Based Evolutionary Algotithm : Genetic Network Programming and Its Extension Using Reinforcement Learning", Evolutionary Computation, MIT Press, to appear.

[9] 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，Trans. IEE of Japan, Vol.122-C, pp 2149-2156, No.12, 2002.