Vol.2009-MPS-75 No.13
2009/9/10

Graph Structured Program Evolutionによる動的な探索空間に対する探索アルゴリズムの獲得

白川 真一[†1,†2] 長尾 智晴[†1]

現在までに，進化計算に代表されるメタヒューリスティクス探索法が様々提案され，その有効性が幅広い領域で示されている．しかし，与えられた問題に対して有効な探索アルゴリズムを設計することは一般的に困難であり，多くの試行錯誤を必要とする．そのため，与えられた問題に対して有効な探索アルゴリズムを自動的に構築することは重要な課題であるといえる．本稿ではグラフ構造を表現形式とする自動プログラミング手法の Graph Structured Program Evolution（GRAPE）を用いて，動的な探索空間に有効な探索アルゴリズムの自動構築を行う．実験では動的な探索問題の例として，EvoSTOC 2009 という国際ワークショップのコンペティションの Yellow Submarine Challenge を取り上げる．GRAPE によって自動構築された探索アルゴリズムは，典型的な探索法と比較して性能が高いことを示す．

Evolution of Search Algorithms for Dynamic Search Spaces Using Graph Structured Program Evolution

Shinichi Shirakawa[†1,†2] and Tomoharu Nagao[†1]

Numerous evolutionary computation (EC) techniques and related improvements showing effectiveness in various problem domains have been proposed in recent studies. However, it is difficult to design effective search algorithms for given target problems. It is therefore essential to construct effective search algorithms automatically. In this paper, we propose a method for evolving search algorithms for dynamic search spaces using Graph Structured Program Evolution (GRAPE), which has a graph structure and is one of the automatic programming techniques developed recently. In the experiments, we adopt "Yellow Submarine Challenge" which is a competition problem in EvoSTOC 2009 as an example of dynamic searching problems. Numerical experiments show that the constructed search algorithm is more effective for the dynamic search spaces than classical search algorithms.

1. はじめに

近年，コンピュータプログラムを自動生成する自動プログラミングに関する研究が活発に行われている．遺伝的アルゴリズム（GA）を用いてコンピュータプログラムを自動的に生成する遺伝的プログラミング（GP）[1),2)] では，一般的にプログラムの表現形式として木構造が用いられ，遺伝操作には部分木の交換による交叉やノードの突然変異などが用いられる．これまでにプログラムの表現形式として木構造ではなく，グラフ構造を用いてプログラムの自動生成を行う手法も研究されている．グラフ構造を扱う利点としては，木構造に比べて複雑な表現が可能であることやグラフ上での時系列情報の保持が可能であることが挙げられる．本報告で用いる Graph Structured Program Evolution（GRAPE）[3),4)] もグラフ構造を利用した自動プログラミング手法である．GRAPE ではグラフ構造をプログラムの表現形式としているため，分岐やループを含む複雑なプログラムの表現が可能である．

ところで，進化計算法に代表されるメタヒューリスティクス探索法はこれまでに様々な手法やその改良法が提案され，様々な問題領域に対して有効性を示している．一方，"すべてのブラックボックス探索において，あらゆる目的関数について他を常に上回るような探索アルゴリズムは存在しない"，という No Free Lunch（NFL）定理[5)] が示されている．このことから，ある特定の問題領域に対して効率的な探索アルゴリズムを構築することが重要であると考えられる．ある問題が与えられたときにその問題に対する有効な探索法を人手で構築することが一般的であるが，この探索アルゴリズムの構築過程を自動化することは非常に重要な課題であるといえる．これに対して，GA を用いて GA のパラメータの最適化を行うメタ GA が提案されている[6)]．一般的なメタ GA では，GA のパラメータとして，集団サイズや交叉率，突然変異率などの最適化を行う．また，GP の一種である Linear Genetic Programming（LGP）[7)] を用いて世代交代の手順を進化させる方法[8)] や，GP を用いて個体の交叉方法を進化させる方法[9)]，Particle Swarm Optimization （PSO）[10)] の更新方法を進化させる方法[11)] などが提案され，有効性が示されている．

さらに，何らかの学習メカニズムを使用して，低レベルのヒューリスティクスを組合わせて新たなヒューリスティクスを作り出す概念はハイパーヒューリスティクス（Hyper-heuristics）

†1 横浜国立大学 大学院環境情報学府
Graduate School of Environment and Information Sciences, Yokohama National University
†2 日本学術振興会 特別研究員 PD
Research Fellow of the Japan Society for the Promotion of Science

Vol.2009-MPS-75 No.13
2009/9/10

[2),12)] と呼ばれ，近年活発に研究が行われている．進化計算によるハイパーヒューリスティクスに関する研究の例としては，GA や GP によって bin-packing problem に対するヒューリスティクスを生成した例[13),14)] や SAT 問題に対するヒューリスティクスを自動生成した例[15)]，GP を用いて 2 目的の 0/1 ナップザック問題に対して，ヒューリスティクス集合を獲得した例[16)] などがある．先に述べたメタ GA や交叉法，世代交代の手順の進化，本稿で行う探索アルゴリズムの進化的獲得は，このハイパーヒューリスティクスに含まれる．

我々は先に，GRAPE を用いて探索空間を探索するエージェントの行動プログラムを自動構築することで，探索アルゴリズムの進化的獲得を行い，関数最適化問題のベンチマーク問題とテンプレートマッチング問題に対して有効な探索アルゴリズムが獲得されたことを確認した[17),18)]．先の実験で扱った問題は，探索中に探索空間が変化しない静的な探索問題であった．そこで本稿では，探索中に探索空間が変化する動的な探索問題に対して有効な探索アルゴリズムの獲得を行い，その有効性の検証を行う．

2. Graph Structured Program Evolution（GRAPE）

GRAPE[3),4)] では，複雑なプログラムの記述を可能にするためグラフ構造を採用している．GRAPE のプログラムは有向グラフとデータセットから構成される．データセットは有向グラフ中を流れ，各ノードにおいてそのノードに応じた処理が施される．各ノードではデータセットに対する処理やデータセットを用いた分岐が行われる．GRAPE の実行時には，まずデータセットに初期値として入力値や定数値などをセットする．その後，スタートノードからプログラムを開始して各ノードを遷移していくことで処理が行われる．GRAPE ではグラフ構造をプログラムの表現形式としているため，分岐やループを含む複雑なプログラムの表現が可能である．さらに，グラフ中を流れるデータセットに様々なデータ型を用意することで，複数のデータ型を 1 つのプログラム中で扱うことができる．各ノードではあらかじめ定められたデータ型を使って処理を行う．GRAPE では表現型のグラフ構造を遺伝子型にマッピングし，遺伝子型に対して遺伝操作を行う．GRAPE の遺伝子型は各ノードの種類，接続，引数を定義した一次元の整数列で表現される．遺伝子型から表現型に変換する際，ノードの種類によっては接続先や引数を指定する遺伝子が表現型に発現しない場合もある．GRAPE の総ノード数はあらかじめ指定するため，染色体の遺伝子長は固定長になる．総ノード数は固定であるが，接続の状態によって使用されるノード（active node）と使用されないノード（inactive node）があるため，表現上はノード数は可変となる．GRAPE はこれまで，階乗を求めるプログラムや任意の長さのリストをソートするプログラムの自動生成などに適用され，その有効性が示されている[3),4)]．

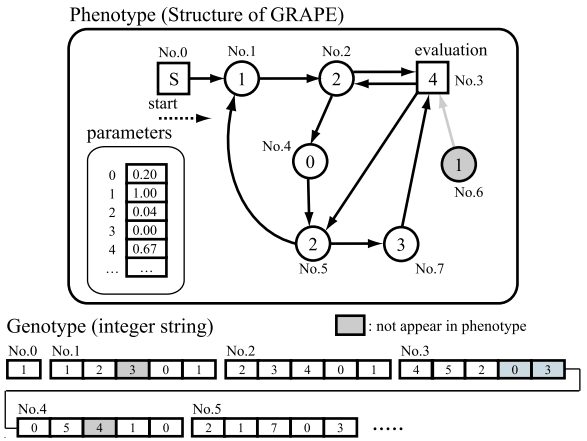


図 1 探索エージェントの行動プログラムを表す GRAPE の構造とその遺伝子型の例.
Fig. 1 Structure of GRAPE (phenotype) which represents action program of search agents and its genotype.

3. GRAPE による探索アルゴリズムの進化

本稿では，GRAPE を用いて探索空間を探索するエージェントの行動プログラムを自動構築することで，探索アルゴリズムの進化を行う．本モデルでは探索空間中をエージェントが移動することで，次の探索点を決定する．エージェントは GRAPE を用いて表現された行動規則にしたがって探索空間を移動する．GRAPE によって表現された行動規則を最適化することで，与えられた探索空間に有効な探索アルゴリズムが獲得できると考えられる．本稿での GRAPE の構造例を**図 1** に示す．GRAPE のデータセットには最適化対象である設計変数が保存される．GRAPE の各ノードを遷移していくことで自身の設計変数に対する処理が行われ，設計変数が更新される．その後，評価ノード（evaluation node）に達すると自身の設計変数が評価され，次のノードに移る．このように，エージェントの行動規則を用いて探索が行われていく．本稿の実験ではすべてのエージェントが同一の行動プログラムに基づいて設計変数を更新する Homogeneous モデルを採用している．各エージェントの行動プログラムは同一であるが，分岐条件などによって設計変数の更新方法が異なることがあるため，多様な探索アルゴリズムの表現が可能となる．

今回の実験で用いたノード関数は，設計変数をランダムに初期化するものや，他のエージェントの設計変数を代入するもの，2 つの設計変数から一様乱数や正規乱数によって新たな設計変数を生成するもの，確率に基づいて分岐を行うものなどである．用意したノードの中には，設計変数の更新を行う際に自身以外のもう一つの設計変数 q を用いることができるものがある．今回の実験で使用したノード関数を**表 1** に示す．基本的には文献 17)，18) で用いたものと同様であるが，動的な探索空間に対応するために，設計変数 q や評価ノードでの分岐条件などを変更している．設計変数 q としては，前世代で最も評価の高かった設計変数（q0），注目エージェントの前世代の設計変数（q1），ランダムに選択したエージェントの前世代の設計変数（q2）の中から決定できることとした．これらのノード関数は実数値 GA などで用いられるものを基に用意した．例えば，N18 は実数値交叉の BLX-α[19] である．また，評価ノード（EVAL）では，前世代の最も良い適応度より良い評価の場合，自身の前世代の適応度より良い評価の場合，その他の場合で 3 分岐することとしている．

GRAPE によって表現された探索アルゴリズムを用いて実際に探索を行い，その結果を基に，より良い探索アルゴリズムの進化を行う．GRAPE の各個体の評価は，与えられた探索空間の探索を行った結果を基に評価を行う．例えば探索を数回行った際の評価値の合計などである．今回の実験で用意したノード関数は比較的単純なものであるが，それらを組み合わせることで複雑な探索アルゴリズムの表現が可能となる．

4. 動的な探索空間に対する探索アルゴリズムの獲得実験

本節では動的な探索空間に対する探索アルゴリズムの獲得実験を行い，獲得した探索アルゴリズムと典型的な探索アルゴリズムの比較を行う．

4.1 Yellow Submarine Challenge

ここでは，EvoStar 2009 という国際会議の一部である 6th European Workshop on Evolutionary Algorithms in Stochastic and Dynamic Environments（EvoSTOC 2009）[⋆1] という国際ワークショップのコンペティションの題材となった Yellow Submarine Challenge を問題として取り上げる．Yellow Submarine Challenge は動的な環境における最適化問題であり，会議のホームページで探索空間（適応度関数）のコードが公開されている [⋆2]．この問題は時系列信号のファイルを使用して動的な探索空間を生成しているため，時系列信号の

表 1 実験に用いたノード関数一覧.
Table 1 Node functions used in the experiments.

Id	# Connections	Arg(s)	Definition
			For a selected nth parameter p_n.
N1	1	n	Initialize the nth parameter p_n with random number.
N2	1	q, n	$p_n = q_n$
N3	1	q, n	$p_n = p_n + u(p_n - \alpha\lvert p_n - q_n\rvert, p_n + \alpha\lvert p_n - q_n\rvert)$
N4	1	q, n	$p_n = u(min(p_n, q_n) - \alpha\lvert p_n - q_n\rvert, max(p_n, q_n) + \alpha\lvert p_n - q_n\rvert)$
N5	1	q, n	$p_n = N(p_n, (\alpha\lvert p_n - q_n\rvert)^2)$
N6	1	q, n	$p_n = N(\frac{(p_n+q_n)}{2}, (\alpha\lvert p_n - q_n\rvert)^2)$
N7	1	n	$p_n = 100 - p_n$
			For a random selected parameter p_r.
N8	1	–	Initialize the parameter p_r with uniform random number.
N9	1	q	$p_r = q_r$
N10	1	q	$p_r = p_r + u(p_r - \alpha\lvert p_r - q_r\rvert, p_r + \alpha\lvert p_r - q_r\rvert)$
N11	1	q	$p_r = u(min(p_r, q_r) - \alpha\lvert p_r - q_r\rvert, max(p_r, q_r) + \alpha\lvert p_r - q_r\rvert)$
N12	1	q	$p_r = N(p_r, (\alpha\lvert p_r - q_r\rvert)^2)$
N13	1	q	$p_r = N(\frac{(p_r+q_r)}{2}, (\alpha\lvert p_r - q_r\rvert)^2)$
N14	1	–	$p_r = 100 - p_r$
			For the all parameters p_i. $(i = x, y)$
N15	1	–	Initialize the all parameters p_i with random number.
N16	1	q	$p_i = q_i$
N17	1	q	$p_i = p_i + u(p_i - \alpha\lvert p_i - q_i\rvert, p_i + \alpha\lvert p_i - q_i\rvert)$
N18	1	q	$p_i = u(min(p_i, q_i) - \alpha\lvert p_i - q_i\rvert, max(p_i, q_i) + \alpha\lvert p_i - q_i\rvert)$
N19	1	q	$p_i = N(p_i, (\alpha\lvert p_i - q_i\rvert)^2)$
N20	1	q	$p_i = N(\frac{(p_i+q_i)}{2}, (\alpha\lvert p_i - q_i\rvert)^2)$
N21	1	–	$p_i = 100 - p_i$
			Other types.
N22	1	q	$p_i = (p_i - \alpha de_i) + R \cdot ((q_i + \alpha de_i) - (p_i - \alpha de_i))$ d: the distance between p_i and q_i, $e_i = \frac{(q_i - p_i)}{\lvert q_i - p_i\rvert}$, $R = u(0.0, 1.0)$
N23	1	q	$p_i = p_i + ze_i$ $z = N(0, (\alpha d)^2)$, d: the distance between p_i and q_i, $e_i = \frac{(q_i - p_i)}{\lvert q_i - p_i\rvert}$
N24	1	q	$p_i = \frac{(p_i+q_i)}{2} + ze_i$ $z = N(0, (\alpha d)^2)$, d: the distance between p_i and q_i, $e_i = \frac{(q_i - p_i)}{\lvert q_i - p_i\rvert}$
N25	2	–	Decide the next node with rondom number. If $u(0.0, 1.0) < 0.5$, then connection 1 is chosen. Else connection 2 is chosen.
N26	2	–	Decide the next node with rondom number. If $u(0.0, 1.0) < 0.1$, then connection 1 is chosen. Else connection 2 is chosen.
			Evaluation node.
EVAL	3	–	Evaluate the current parameters p. And if the previous best fitness is updated, then connection 1 is chosen. Else if the previous fitness is updated, then connection 2 is chosen. Else connection 3 is chosen.

p_i: ith parameter of agent p.
q_i: ith parameter of partner agent q.
(q is the previous best parameters of all agents $q0$ or the agent's previous parameters $q1$
or the previous parameters of the random selected agent $q2$)
$u(x, y)$: uniformly distributed random number among $[x, y]$.
$N(\mu, \sigma^2)$: normally distributed random number.
Parameter $\alpha = 0.5$.

⋆1 http://evostar.na.icar.cnr.it/EvoWorkshops/EvoSTOC/EvoSTOC.html
⋆2 http://evostar.na.icar.cnr.it/EvoWorkshops/EvoSTOC/EvoSTOCCompetition/EvoSTOCCompetition.html

ファイルを変更すると異なる探索空間が作成できる [*1].

次に Yellow Submarine Challenge の具体的なルールを述べる．問題は動的な探索空間における最大化問題であり，2 次元の設計変数（x, y）によって適応度が決定する．最大世代数が 10000 世代であり，各世代ごとに探索空間の形状が変化する（変化しない場合もある）[*2]．各設計変数は実数値で定義され，その定義域は $[0, 100]$ である．各世代では 20 点だけ個体評価を行うことができる．探索アルゴリズムの評価は各世代の最大適応度の合計で行う．

コンペティションのファイナリストの探索アルゴリズムは，会議のホームページで公開されている探索空間とは異なる探索空間にも適用され，評価が行われた．今回の実験では会議のホームページで公開されているものと，ファイナリストに対して提供された探索空間の 2 つを用いる．ここで便宜上，会議のホームページで公開されているものを “探索空間 1”，ファイナリストに対して提供されたものを “探索空間 2” とする．これらの探索空間は対称性を有しており，探索空間の変化が緩やかな期間と変化が激しい期間が混在している．特に，探索空間の変化が激しい期間に適応度が大きい領域が現れるという特徴をもつ．他に特徴的な点としては，最初の 1000 世代程度は探索空間の形状がほとんど変化しないことが挙げられる．

4.2 実験の設定

今回の実験で使用したパラメータ値を**表 2** に示す．実験ではノードの遷移回数に制限を設け，最大遷移回数（Execution step limits）内に評価ノードに到達しない個体は最も悪い評価値が与えられることとした．今回の実験では最大遷移回数を 50 とした．GRAPE の遺伝操作としては一様交叉と文字列に対する突然変異を用い，世代交代モデルには Minimal Generation Gap（MGG）[20] を使用した．GRAPE の各個体を評価する際には，実際に動的な探索空間に対して探索を行い，その結果を基に評価を行い，探索アルゴリズムを構築する．探索アルゴリズムの構築の際には，探索空間 1 に対して 5 回の探索を行い，その評価値の合計で GRAPE の個体の評価を行う．探索の際の初期世代の設計変数は一様乱数によって初期化される．Yellow Submarine Challenge のルール通り，エージェント数は 20，世代数は 10000 である．その後，GRAPE によって構築された探索アルゴリズムを用いて，探索空間 1，2 に対して探索を行い，その性能も評価する．

表 2 実験に用いた GRAPE のパラメータ値.
Table 2 Parameters of GRAPE used in the experiments.

Number of generations	10000
Population size	100
Children size (for MGG)	20
Crossover rate	0.6
Uniform crossover rate (P_c)	0.1
Mutation rate (P_m)	0.02
Maximum number of nodes	50
Execution step limits	50

4.3 比較手法

GRAPE によって構築された探索アルゴリズムの性能を評価するために，いくつかの典型的な探索アルゴリズムとの比較を行う．ここでは，比較に用いた各探索アルゴリズムの説明を行う．

4.3.1 Genetic Algorithm（GA）

本実験で用いた GA の各世代での手順は次の通りである．

(1) 前世代で評価の高い 2 個体をエリートとして次世代の個体として保存する．
(2) トーナメント選択によって前世代の個体集団から 2 個体を選択し，BLX-α[19] によって 2 個体を生成する [*3].
(3) 生成された 2 個体に対して突然変異を施し，次世代の個体とする（突然変異は各個体の設計変数に対して突然変異率に基づいて発生し，対象となった設計変数が一様乱数によって初期化される）.
(4) 次世代の個体が 20 個体になるまで（2）と（3）を繰り返す.

実験ではトーナメントサイズを 2，BLX-α のパラメータ α を 0.5，突然変異率を 0.05 として実験を行う．これらのパラメータは予備実験から設定した．また，初期世代の個体は一様乱数によって生成することとした.

4.3.2 Particle Swarm Optimization（PSO）

PSO[10] は動物の社会的振る舞いを模倣した多点探索手法であり，解候補である個体は現在の位置情報である実数値を保持し，それを更新式に従って移動させることで最適解を探索

⋆1 今回のコンペティションでは The Beatles の Yellow Submarine から時系列信号ファイルが生成されたのではないかと推察される.
⋆2 探索空間の形状の変化の様子が `http://www.youtube.com/watch?v=4S3fGBvDfmU` で公開されている.
⋆3 BLX-α は 2 親によって定まる各設計変数の区間を α 倍拡張してできる超直方体内に子個体を一様乱数によって生成する交叉法である.

する．各個体 i の現在の位置ベクトルを x_i，速度ベクトルを v_i とする．今回は動的な探索空間に対応するために，静的な探索問題に対する PSO の更新式を一部変更している．次にその更新式を示す．

$$v_i = \omega v_i + c_1 r_1(\hat{x}_r - x_i) + c_2 r_2(\hat{x}_b - x_i) \quad (1)$$

$$x_i = x_i + v_i \quad (2)$$

ここで，$\hat{x}_r$ は前世代からランダムに選択した他個体の位置ベクトル，$\hat{x}_b$ は前世代の最良個体の位置ベクトルである．ω は速度に対する重み係数，c_1 と c_2 は各項に対する重み係数である．r_1 と r_2 は $[0.0, 1.0]$ の範囲の一様乱数を表している．各パラメータ値は予備実験から，$\omega = 0.72$，$c_1 = 0.72$，$c_2 = 1.49$ とした．また，速度ベクトル v_i の範囲は $[-25, 25]$ として，v_i の初期値もこの範囲内の一様乱数によって決定することとした．

4.3.3 ランダム探索（RAND）

ランダム探索では，各世代で定義域内に一様乱数によって個体を生成する．

4.4 実験の結果と考察

図 2 に GRAPE によって構築された探索アルゴリズムの構造を示す．この探索アルゴリズムでは，次のように設計変数を更新する．

(1) 前世代の最も良い適応度を更新したエージェント
- 設計変数を更新しない．

(2) 前世代の自身の適応度を更新したエージェント
- 確率 0.1 で次の手順（a）を，それ以外の場合は次の手順（b）によって設計変数を更新する．

(a) 1 番目の設計変数を反転させる（$p_x = 100 - p_x$）．その後，ランダムに選択した 1 つの設計変数を前世代で最も良い評価のエージェントの設計変数の周辺に正規乱数で生成した値に更新する．

(b) ランダムに選択した 1 つの設計変数を前世代の中からランダムに選択したエージェントの設計変数の周辺に正規乱数で生成した値に更新する．その後，前世代で最も良い評価のエージェントの設計変数と BLX-α を行うことで値を更新する．

(3) 前世代の自身の適応度を更新しなかったエージェント
- ランダムに選択した 1 つの設計変数を一様乱数で初期化した後，確率 0.5 で次の手順（a）を，それ以外の場合は次の手順（b）によって設計変数を更新する．

(a) 一様乱数によって全ての設計変数を初期化する．次に，全ての設計変数を前世代の中からランダムに選択したエージェントの設計変数の周辺に正規乱数で生成した値に更新する．その後，2 番目の設計変数を一様乱数によって初期化する．最後に，現在の設計変数と前世代で最も良い評価のエージェントの設計変数とを結ぶ直線上に一様乱数で生成した値に更新する．

(b) (2) の場合と同様の手順で更新を行う．

この探索アルゴリズムによって生成される探索点は，前世代で最も良い評価を受けた探索点の周辺に生成されものが多いが，一部は離れた場所に生成されるため，多様性を保つことができていると考えられる．これによって，探索空間の変化が緩やかな場合には適応度の高い領域に個体を生成し続けることができる．また，探索空間の形状が大きく変化した場合にも，前世代で適応度の高かった場所から離れた場所に生成された探索点によって，変化に追従することができていると考えられる．

次に，GRAPE によって構築された探索アルゴリズムと比較手法を探索空間 1，2 に適用する．ここでは，各探索アルゴリズムについて 1000 回の独立な試行を行った．1 回の試行で得られる各アルゴリズムに対する評価値は，各世代における最大適応度の合計である．GRAPE によって構築された探索アルゴリズムと GA，PSO，ランダム探索の探索性能をそれぞれ**表 3** に示す．PSO は探索の序盤で集団が収束してしまうと，探索が進まないため，試行によっては得点がランダム探索より低い場合が見受けられた．表 3 から，どちらの動的な探索空間に対しても，GRAPE によって構築された探索アルゴリズムは他の探索アルゴリズムより高い性能を示していることが確認できる．このことからも，GRAPE によって Yellow Sumbarine Challenge に対して有効な探索アルゴリズムの獲得が行えたといえる．さらに，探索空間 1 を用いて探索アルゴリズムを構築したにも係わらず，探索空間 2 に対しても効果的な探索アルゴリズムとなっていることから，Yellow Submarine Challenge に対して有効な探索アルゴリズムが獲得できたと考えられる．

図 3 は，GRAPE によって構築された探索アルゴリズムを用いて探索を行った際の適応度の推移をそれぞれ示したものである．このときの各世代の適応度の合計は，探索空間 1 が 148.28，探索空間 2 が 101.41 である．他に比べて高い適応度が獲得できている期間は，探索空間の形状が大きく変化している期間である．GRAPE によって構築された探索アルゴリズムは探索空間の形状が大きく変化する際にも，変化に追従できていると考えることができる．

Vol.2009-MPS-75 No.13
2009/9/10

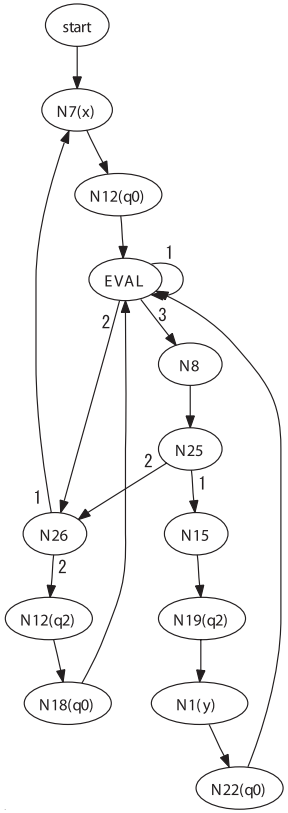


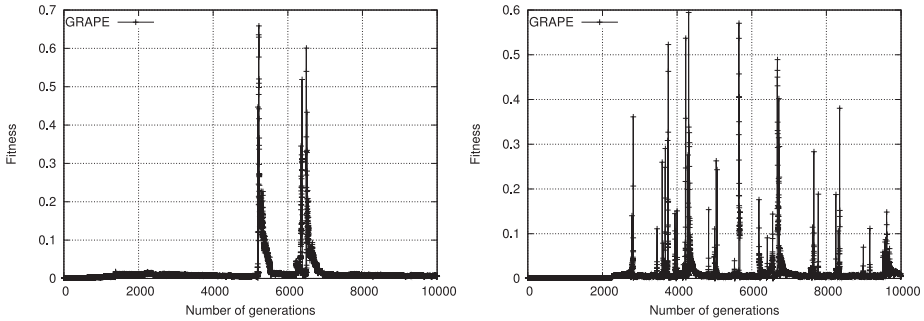
図 2 Yellow Submarine Challenge 問題に対して獲得した探索アルゴリズムの構造.
Fig. 2 Obtained search algorithm structure for "Yellow Submarine Challenge" problem using GRAPE.

また，今回の実験での探索過程の様子を表した動画を著者のホームページ上に公開する[⋆1]ので参照していただきたい．これらの動画からも GRAPE によって構築された探索アルゴリズムが，動的な探索空間に適応できている様子が見て取れる．

⋆1 http://www.nlab.sogo1.ynu.ac.jp/~shirakawa/ysc/mps.html

表 3 GRAPE によって獲得された探索アルゴリズムと比較手法の性能（1000 回の試行の平均）.
Table 3 The average performances by applying the evolved search algorithm and comparative algorithms (average over 1000 trials).

	Average	Stddev	Best	Worst
Search Space 1				
GRAPE	143.31	2.45	148.28	133.32
GA	134.88	3.22	141.33	124.79
PSO	115.89	16.47	124.34	0.004
RAND	72.81	0.83	75.56	69.70
Search Space 2				
GRAPE	96.44	2.05	101.41	88.15
GA	91.19	1.58	95.37	85.69
PSO	83.00	13.16	88.42	0.019
RAND	54.68	0.86	57.82	52.05



探索空間 1　　探索空間 2

図 3 GRAPE によって構築された探索アルゴリズムの探索空間 1，2 に対する適応度の推移.
Fig. 3 Transitions of fitness value for search space 1 and 2 using the search algorithm constructed by GRAPE.

5. ま と め

本稿では GRAPE を用いて探索空間を探索するエージェントの行動プログラムを自動構築することで，動的な探索空間に対する探索アルゴリズムの進化を行った．実験では，国際ワークショップのコンペティションで用いられた Yellow Submarine Challenge を対象に探

索アルゴリズムの構築を行った．GRAPE によって構築された探索アルゴリズムは，Yellow Submarine Challenge の動的な探索空間に対して良好な結果を示すものであった．さらに，典型的な探索アルゴリズムと比較してもその性能の高さを示すことができた．最後に，本稿で述べた GRAPE によって構築された探索アルゴリズムは，実際の Yellow Submarine Challenge のコンペティションで優勝することができた．

参 考 文 献

1) Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press (1992).
2) Poli, R., Langdon, W. B. and McPhee, N. F.: *A field guide to genetic programming*, Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk` (2008). (With contributions by J. R. Koza).
3) Shirakawa, S., Ogino, S. and Nagao, T.: Graph Structured Program Evolution, *Proceedings of Genetic and Evolutionary Computation Conference (GECCO'07)*, Vol.2, pp.1686–1693 (2007).
4) 白川真一，長尾智晴：Graph Structured Program Evolution によるプログラムの自動生成，電気学会論文誌 C， Vol.128, No.3, pp.370–380 (2008).
5) Wolpert, D.H. and Macready, W.G.: No Free Lunch Theorems for Optimization, *IEEE Transactions on Evolutionary Computation*, Vol.1, No.1, pp.67–82 (1997).
6) 伊庭斉志：遺伝的アルゴリズムの基礎，オーム社 (1994).
7) Brameier, M. and Banzhaf, W.: *Linear Genetic Programming*, Genetic and Evolutionary Computation, No.XVI, Springer (2007).
8) Oltean, M.: Evolving Evolutionay Algorithm Using Linear Genetic Programming, *Evolutionary Computation*, Vol.13, No.3, pp.387–410 (2005).
9) Dioşan, L. and Oltean, M.: Evolving Crossover Operators for Function Optimization, *Proceedings of the Ninth European Conference on Genetic Programming (EuroGP '06)*, LNCS, Vol.3905, Budapest, Hungary, Springer-Verlag, pp.97–108 (2006).
10) Kennedy, J.F. and Eberhart, R.C.: Particle Swarm Optimization, *Proceedings of the IEEE International Conference on Neural Networks*, Vol.IV, Perth, Australia, pp.1942–1948 (1995).
11) Dioşan, L. and Oltean, M.: Evolving the structure of the Particle Swarm Optimization algorithms, *Proceedings of the European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP '06)*, LNCS, Vol.3906, Budapest, Hungary, Springer-Verlag, pp.25–36 (2006).
12) Soubeiga, E.: Development and application of hyperheuristics to personnel scheduling, PhD Thesis, University of Nottingham (2003).
13) Ross, P., Marín-Blázquez, J.G., Schulenburg, S. and Hart, E.: Learning a procedure that can solve hard bin-packing problems: a new GA-based approach to hyperheuristics, *Proceedings of the Genetic and Evolutionary Computation Conference 2003 (GECCO '03)*, LNCS, Vol.2724, Chicago, Illinois, USA, pp.1295–1306 (2003).
14) Burke, E.K., Hyde, M.R. and Kendall, G.: Evolving Bin Packing Heuristics with Genetic Programming, *Parallel Problem Solving from Nature - PPSN IX*, LNCS, Vol.4193, Reykjavik, Iceland, Springer-Verlag, pp.860–869 (2006).
15) Fukunaga, A.S.: Automated Discovery of Local Search Heuristics for Satisfiability Testing, *Evolutionary Computation*, Vol.16, No.1, pp.31–61 (2008).
16) Kumar, R., Joshi, A.H., Banka, K.K. and Rockett, P.I.: Evolution of hyperheuristics for the biobjective 0/1 knapsack problem by multiobjective genetic programming, *Proceedings of the Genetic and Evolutionary Computation Conference 2008 (GECCO '08)*, Atlanta, GA, USA, ACM Press, pp.1227–1234 (2008).
17) 白川真一，長尾智晴：Graph Structured Program Evolution による探索アルゴリズムの進化，第 18 回インテリジェント・システム・シンポジウム（FAN 2008），pp. 209–214 (2008).
18) Shirakawa, S. and Nagao, T.: Evolution of Search Algorithms Using Graph Structured Program Evolution, *Proceedings of the 12th European Conference on Genetic Programming, EuroGP 2009*, LNCS, Vol.5481, Tuebingen, Springer, pp.109–120 (2009).
19) Eshelman, L.J. and Schaffer, J.D.: Real Coded Genetic Algorithms and Interval-Schemata, *Foundations of Genetic Algorithms 2*, San Mateo, CA, Morgan Kaufmann Publishers, pp.187–202 (1993).
20) 佐藤　浩，小野　功，小林重信：遺伝的アルゴリズムにおける世代交代モデルの提案と評価，人工知能学会誌， Vol.12, No.5, pp.734–744 (1997).