論 文

遺伝的プログラミングによる実数値GAの性能差を強調する探索空間の生成

Evolving Search Spaces to Emphasize the Performance Difference of Real-Coded Genetic Algorithms Using Genetic Programming

白川 真一*1 横浜国立大学 大学院環境情報研究院
Shinichi Shirakawa Research Institute of Environment and Information Sciences, Yokohama National University
shinichi.shirakawa@gmail.com, http://www.shirastar.com/

矢田 紀子 （同 上）
Noriko Yata yata@nlab.sogo1.ynu.ac.jp

長尾 智晴 （同 上）
Tomoharu Nagao nagao@ynu.ac.jp, http://www.nlab.sogo1.ynu.ac.jp/

keywords: genetic programming, real-coded genetic algorithm, function optimization, search space, multiobjective evolutionary algorithm

Summary

When we evaluate the search performance of an evolutionary computation (EC) technique, we usually apply it to typical benchmark functions and evaluate its performance in comparison to other techniques. In experiments on limited benchmark functions, it can be difficult to understand the features of each technique. In this paper, the search spaces that emphasize the performance difference of EC techniques are evolved by Cartesian genetic programming (CGP). We focus on a real-coded genetic algorithm (RCGA), which is a type of genetic algorithm that has a real-valued vector as a chromosome. The performance difference of two RCGAs is assumed to be a objective function of CGP, and the search space that increases the performance difference is evolved. In particular, we generate search spaces using the performance difference of real-coded crossovers or generation alternation models. As a result of our experiments, the search spaces that exhibit the largest performance difference of two RCGAs are generated for all the combinations. In addition, we extend the objective functions to two of the performance differences and the number of active nodes in CGP and attempt to generate multiple search spaces with an evolution using a multiobjective evolutionary algorithm. We then observe which types of elements expand the performance difference.

1. はじめに

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，種々の問題領域に対して有効性を示している．特に，実数値関数の最小値または最大値を求める関数最適化問題に対しては，様々なアプローチが試みられている．進化計算法による関数最適化問題に対する代表的な手法としては，実数値GA（Real-coded Genetic Algorithm; RCGA）[小林 09], Covariance Matrix Adaptation Evolution Strategy（CMA-ES）[Hansen 01]，Particle Swarm Optimization（PSO）[Kennedy 95]，Differencial Evolution（DE）[Storn 97] などが挙げられる．このような進化計算法の探索性能を評価する際には，その手法を典型的なベンチマーク関数に適用し，他の手法との比較実験によってその手法の性能を評価することが多い．しかし，複数のベンチマーク関数に対する手法の差異が微少であった場合，その手法にどのような特徴があるのか理解することは難しい．また，実験に用いていないベンチマーク関数や他の実問題に対しての性能は明らかではない．一方で，理論的に進化計算法の挙動を解析する研究も行われているが，複雑な進化計算法を理論的に解析，理解することは多くの仮定を必要とするなど困難な場合も多く存在する．

このような状況に対して，2 つの進化計算法の性能差を大きくするような探索空間を進化的に生成する研究が行われている．Oltean は設定を変えたシンプルな 2 つの (1+1)ES の性能差が大きくなるような 1 次元の実数値の探索空間の生成と，ランダムサーチが突然変異だけを用いる進化計算法よりも性能が高くなるようなバイナリの探索空間の生成に成功している [Oltean 04]．また，PSO や DE などの性能差を大きくするような関数最適化問題を遺伝的プログラミング（Genetic Programming; GP）[Koza 92, Poli 08] によって生成し，それを進化計算法の性能解

*1 現職：株式会社富士通研究所

析に応用しようとする研究が行われている [Langdon 07]. この研究では，PSO や DE の性能差を GP の評価関数とし，“$+, -, \times, \div$” の演算の組み合わせによって表現される探索空間を生成する．それによって，ある進化計算法が他の進化計算法に探索性能が勝る（または劣る）探索空間の生成に成功している．

探索問題を生成するという観点では，Hemert が行った組み合わせ最適化問題を進化的に生成する研究がある [Hemert 06]．この研究では，3 種類の組み合わせ最適化問題（二項制約充足問題，充足可能性問題，巡回セールスマン問題）の生成を行い，難しい問題を生成することに成功している．

本論文では実数値 GA に着目し，いくつかの実数値 GA 間の性能差を利用して探索空間の生成を行う．現在までに実数値 GA に対しては様々な交叉や世代交代モデルが提案されており，それぞれに特色がある．また，“機能分担仮説” や “統計量の遺伝” などの設計指針 [Kita 99a, 喜多 99b, 喜多 99c] によって，世代交代モデルと交叉の役割が明確である．本論文の実験で行う探索空間の生成には，ネットワーク構造をプログラムの表現形式とする GP の一つである Cartesian Genetic Programming（CGP）[Miller 00, Miller 06] を用いる．さらに，探索空間の生成時の目的関数を性能差と CGP のノード数の 2 つに拡張し，多目的進化アルゴリズムによって複数の探索空間を一度に生成することを試みる．本研究では 2 つの進化計算法間の性能差を強調する探索空間を CGP によって生成することで，アルゴリズムの違いとその探索への影響を理解する手助けになることを期待している．

本論文の構成は次の通りである．2 章では，実数値 GA の性能差を強調する探索空間の生成法について述べる．3 章では実数値交叉，4 章では世代交代モデルの性能差を利用して探索空間の生成実験を行う．さらに，5 章で多目的進化アルゴリズムによって複数の探索空間を一度に生成することを試みる．最後に本論文をまとめ，今後の課題について述べる．

2. 実数値 GA の性能差を強調する探索空間の生成法

2·1 CGP による探索空間の生成

本論文では，実数値 GA の性能差を強調する探索空間を GP によって生成する．本論文での探索空間の生成法の概要図を図 1 に示す．GP の各個体（プログラム）は探索空間を表す関数（数式）に対応している．実験では設定を変えた 2 つの実数値 GA の性能差を各個体の評価関数とし，その性能差が大きくなるような探索空間の生成を行う．このようにして探索空間を GP によって生成することで，ある実数値 GA がどのような探索空間に対して性能を発揮するのかを調べることができる．実験では特に，探索空間の景観の把握が容易な 2 次元の探索空間の生成を行う．つまり，GP の各個体は 2 つの入力値 (x, y) から，ある評価値 f を返す関数を表すことになる．

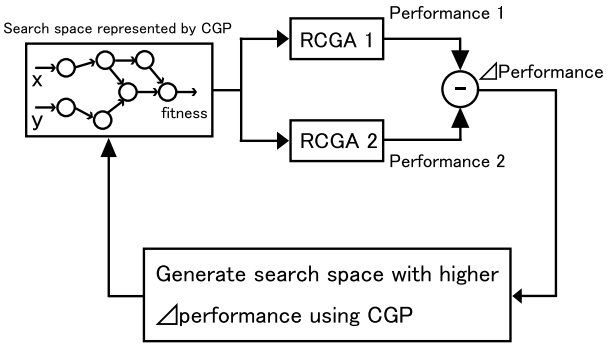


図 1 実数値 GA の性能差を強調する探索空間の生成法の概要図

本論文では GP のモデルとして，ネットワーク構造をプログラムの表現形式とする Cartesian Genetic Programming（CGP）[Miller 00, Miller 06] を採用する．CGP の各プログラムはフィードフォワード型のネットワーク構造で表現され，入力ノード，演算ノード，出力ノードから構成される．入力ノードはプログラムへの入力値が対応し，出力ノードからプログラムの出力値を得る．フィードフォワード型のネットワーク構造を採用する利点としては，ノードの再利用が可能となることが挙げられる．CGP では表現型のネットワーク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．ここで，各ノードにはあらかじめ番号が振られているものとし，入力ノード，演算ノード，出力ノードの順に番号が割り当てられる．入力元として選択できるノードを，自分のノード番号より小さい番号に制限することで，フィードフォワード構造を実現している．CGP の遺伝子型は各ノードの種類，接続元を記述した一次元の文字列で表現される．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元の遺伝子が表現型に発現しない場合もある．CGP の全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）が存在するため，表現上はノード数は可変となる．CGP の遺伝操作には通常，突然変異だけが用いられる．突然変異は突然変異率 P_{m} によって遺伝子単位で発生する．CGP の世代交代モデルには (1+4)ES を使用し，世代交代の際に最良個体が複数存在した場合は，子個体を優先して選択する．CGP では接続状態によって使用されないノードが存在するため，遺伝操作によって評価値の変化が起こらない場合が多く存在する（これは neutrality と呼ばれている）．最良個体が複数存在した場合は子個体を優先して選択することで，評価値が上昇しなかった場合でも探索点の移動が起こりうる．これによって，個体数の少ない単純な世代交代モデルでも効率的な探索が行えると考えられ，実験的にもその有効性が示されている．さらに，CGP の個体表現と小さな確率の突然変異を

表 1 探索空間の生成実験に使用する CGP の演算ノード

記号	入力数	定義
$+$	2	$x_1 + x_2$
$-$	2	$x_1 - x_2$
$\times$	2	$x_1 \times x_2$
$\div$	2	$x_1 \div x_2$ (if $x_2 = 0.0$ return x_1)
$\sqrt{\lvert\ \rvert}$	1	$\sqrt{\lvert x_1 \rvert}$
sin	1	$\sin(x_1)$
cos	1	$\cos(x_1)$
0.0	0	Constant value 0.0
1.0	0	Constant value 1.0
10.0	0	Constant value 10.0
-1.0	0	Constant value -1.0
π	0	Constant value π

用いることで，active node に変更がない遺伝操作が多く発生する．先に述べたように，このときにも探索点の移動が起こるが，その際には評価値計算を省略することができる．

表 1 に本論文で用いる CGP の演算ノードの一覧を示す．CGP の演算ノードには四則演算を中心に比較的単純なものを用いたが，これらの演算を組み合わせることで様々な探索空間を表現することが期待される．

CGP の各個体の評価の際には，その個体が表す探索空間に対して，設定の異なる 2 つの実数値 GA をそれぞれ複数回適用する．その探索結果の違いを基に CGP の各個体の評価を行う．本論文で用いた CGP の各個体の評価関数 F_{cgp} は次の式で定義される．

$$F_{\text{cgp}} = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T}\ln\left(\frac{g_1(t,n)}{g_2(t,n)}\right),$$

$$g_i(t,n) = \begin{cases} f_{\min} & \text{if } f_i(t,n) < f_{\min} \\ f_{\max} & \text{if } f_i(t,n) > f_{\max} \\ f_i(t,n) & \text{otherwize} \end{cases} \quad (1)$$

ここで，N は各実数値 GA の総試行回数，T は最大世代数，$f_i(t,n)$ は探索の n 回目の試行の t 世代目における最良評価値である．$g_i(t,n)$ の値は $[f_{\min}, f_{\max}]$ の範囲に制限されており，g_1 と g_2 は異なる設定の実数値 GA1，2 の探索からそれぞれ得られる値である．実験では，$N = 10$，$T = 500$，$f_{\min} = 10^{-8}$，$f_{\max} = 10.0$ とした．この評価関数 F_{cgp} は 2 つの実数値 GA の各世代における最良評価値の対数値の差を累積したものである．つまり，より早く良い評価値を得るほうが性能が高いことを表している．本論文の実験では，CGP が表現する探索空間に対してより小さい値を得ることができる変数を実数値 GA は探索することとする．そのため，$g_i(t,n)$ の値が小さいほど実数値 GA の性能が高いとみなしている．また，F_{cgp} の値が大きいほど性能差が大きい探索空間であることを表す（F_{cgp} の値が大きい程，その探索空間に対して実数値 GA1 より実数値 GA2 の性能が高いことを表している）．

このような評価関数 F_{cgp} を用いて，2 つの実数値 GA の性能の差が大きくなるように CGP によって探索空間の生成を行う．本論文では，探索アルゴリズムの性能は，より早く良い評価値を得るほうが高いとしているが，目的によっては最終的な最良評価値や解を発見するまでの評価回数が指標となる場合もある．このような場合には，最良評価値や評価回数を CGP の評価関数に採用することも考えられるが，最良評価値や評価回数を CGP の評価に用いると，CGP が表現する探索空間に対して 2 つの探索アルゴリズムが同じ最良評価値を得る場合や，解が発見できない場合に，それらの探索空間に対する性能差が評価できなくなってしまう．本論文では CGP が表現する探索空間に対する性能差をきめ細かく設定するという観点から，探索過程の評価値も式 (1) に導入している．

2·2 本論文で用いる実数値 GA

ここでは，本論文で使用する実数値 GA を構成する実数値交叉と世代交代モデルについて述べる．

実数値交叉としては，BLX-α [Eshelman 93]，単峰性正規分布交叉（Unimodal Normal Distribution Crossover; UNDX）[Ono 97, 小野 99]，シンプレクス交叉（Simplex Crossover; SPX）[Tsutsui 99, 樋口 01] の 3 種類を使用する．BLX-α は，2 親によって定まる各変数の区間を 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である．実験では BLX-α のパラメータは文献 [樋口 01] で示された統計量の遺伝を満たす $\alpha = 0.366$ とした．UNDX は変数間依存性へ対処することを目標に設計された交叉である．UNDX では 3 つの親個体によって決まる正規乱数を用いて子個体を生成する．子個体は 2 つの親個体を結ぶ線分の周辺に正規分布に従って生成され，3 番目の親は正規分布の標準偏差を決めるために用いられる．実験では UNDX のパラメータは推奨値の $\alpha = 0.5, \beta = 0.35$ とした．SPX は n を次元数としたときに，$n+1$ 個の親個体を頂点とする n 次元単体（simplex）に相似な単体内に一様分布に従って子個体を生成する交叉である．SPX も変数間依存性に対処することができる交叉である．実験では SPX のパラメータである拡張率 ε を推奨値である $\varepsilon = \sqrt{n+2}$ とした．

世代交代モデルには，Simple GA（SGA）で用いられている世代交代手順と Minimal Generation Gap（MGG）[佐藤 97] を使用した．SGA は最もシンプルかつ広く知られている GA のアルゴリズムであり，世代ごとに全個体が入れ替わる．本論文では便宜上 SGA で用いられている世代交代手順を SGA と呼ぶことにする．実験で用いる SGA の各世代での手順は次の通りである．

(1) 個体集団から交叉に必要な親個体をトーナメント選択（トーナメントサイズ：2）によって選択する．
(2) 交叉によって子個体を 2 個生成し，次世代の個体とする．
(3) 次世代の個体数分だけ個体が生成されるまで手順

表 2 実験に用いた CGP の各パラメータ値

Parameter	Value
Number of generations	15000
Mutation rate P_{m}	0.02
Number of arithmetic operation nodes	200
Number of input nodes	2
Number of output nodes	1

(1) , (2) を繰り返す.

SGA ではエリート保存戦略が併用されることが多い．エリート保存戦略は，集団中で最良個体をそのまま次世代に残す方法である．本論文ではエリート数を 1 とし，エリート保存戦略を併用した SGA を “SGA with Elite”，エリート保存戦略を併用しない SGA を単に “SGA” と呼ぶことにする．MGG は多様性維持に優れたモデルであり，実数値 GA で広く使用されている．実験で用いる MGG の各世代での手順は次の通りである．

(1) 個体集団から交叉に必要な親個体をランダムに選択する．
(2) 交叉によって子個体を n_{c} 個生成する．
(3) 親個体 2 個体（交叉が UNDX の場合は主親 2 個体，SPX の場合はランダムに 2 個体を選択する）と子個体の中から，最良個体と評価値のランキングに基づくルーレット選択で選択された 1 個体を個体集団に戻す．

実数値 GA による探索範囲は $[-10, 10]$ とし，初期個体はこの範囲に一様乱数によって生成する．また，交叉によって範囲外に子個体が生成された場合は，境界上に値を修正する．実数値 GA の探索過程で f_{min} より小さい評価値が得られた場合には，その世代で探索を打ち切る．式 (1) で CGP の評価値を算出するために f_{max} を設定しているが，実数値 GA の探索の際には評価値に上限を設けない．さらに，各実数値 GA の探索を行う際には，同一のシードを用いて乱数を初期化する．これによって，同じ探索空間に対しては毎回同一の評価を与えることができるようにしている．このため，同一の設定の実数値 GA を用いて式 (1) の値を算出した場合には必ず 0.0 となる．

3. 実数値交叉の性能差を強調する探索空間の生成実験

ここでは，実数値 GA の交叉に着目し，交叉方法を変えた 2 つの実数値 GA を用いて探索空間の生成を行う．

3·1 実験の設定

実験で使用した CGP の各パラメータ値を表 2 に示す．実数値交叉としては，BLX-α，UNDX，SPX の 3 種類を使用し，6 通りの組み合わせに対して探索空間の生成を行った．実数値 GA の世代交代モデルには MGG を使用した．実数値 GA のパラメータは各交叉で共通のものを用いることとし，個体数を 30，子個体生成数 $n_{\mathrm{c}} = 20$，最大世代数 $T = 500$ とした．CGP の各個体が表現する探索空間は 2 次元であるため，このパラメータで十分な探索が行えることを予備実験によって確認している．実験は各交叉の組み合わせににつき 10 回の CGP による探索空間の生成を行った．

表 3 10 回の試行で生成された実数値交叉の性能差を強調する探索空間の平均評価値（括弧内の数値は，左が標準偏差，右が評価値が 5000 以上となった試行回数）

		Lose		
		BLX-α	UNDX	SPX
Win	BLX-α	–	4778	5463
		–	(1533, 5)	(1473, 6)
	UNDX	7521	–	5911
		(1918, 8)	–	(1284, 9)
	SPX	7411	2854	–
		(2024, 8)	(1244, 1)	–

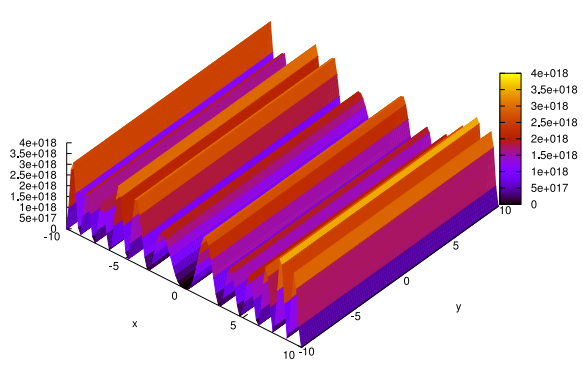
3·2 実験結果と考察

各交叉の組み合わせにつき 10 回の試行を行って得られた探索空間の評価値の平均を表 3 に示す．表中の数値は，各行の交叉を用いた実数値 GA が各列の交叉を用いた実数値 GA よりも性能が高くなる探索空間を生成した際の評価値を表している．また，括弧内の値は標準偏差と評価値が 5000 以上となった実験回数である．この表から，全ての組み合わせに対して，2 つの交叉の性能差が大きくなる探索空間の生成が行えていることが確認できる．

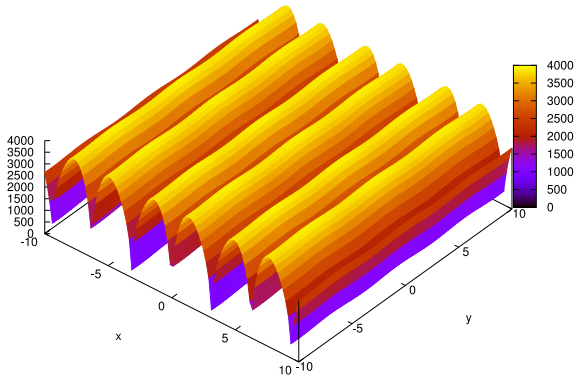
図 2 は各交叉の組み合わせに対して 10 回の試行で生成された探索空間のうち，最も評価値が大きかった探索空間の景観を示したものである．図中の探索空間の高さはその座標での評価値を表し，色についても同様に評価値を表しており，黄色は高い値を青色は低い値を表している．表 4 は図 2 の探索空間に対して各交叉を用いた実数値 GA で 30 回の探索を行った際の成功回数を示したものである．ここでは 1.0×10^{-8} より小さい値を得ることができた試行を成功としてカウントした．

図 2(a) と (b) は，BLX-α が UNDX と SPX にそれぞれ勝る探索空間の例である．どちらも多峰性の探索空間であり，x の値だけによって評価値がほぼ定められ，変数間依存性のない探索空間である．図 2(a) の探索空間では $x = 0$ の付近で最小値をとり，(b) の探索空間では x が π の倍数の付近で最小値をとる．BLX-α は変数間依存性がある探索空間で性能が悪化するとされているため，BLX-α が性能を発揮できる探索空間として図 2(a) や (b) のような変数間依存性がない探索空間が生成されたと考えられる．表 4 を見ると，BLX-α は全ての試行で探索に成功しているが，UNDX や SPX の成功回数は BLX-α に比べて少ない．

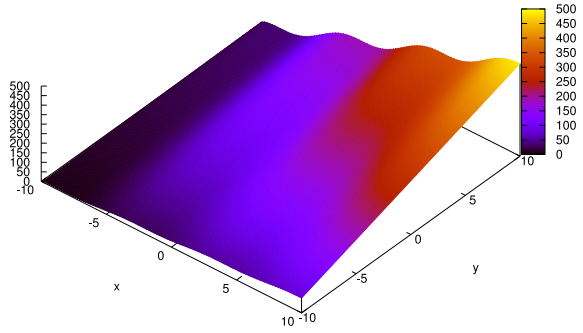
図 2(c) は UNDX が BLX-α に勝る探索空間の例である．この探索空間では評価値を大きく改善するためには



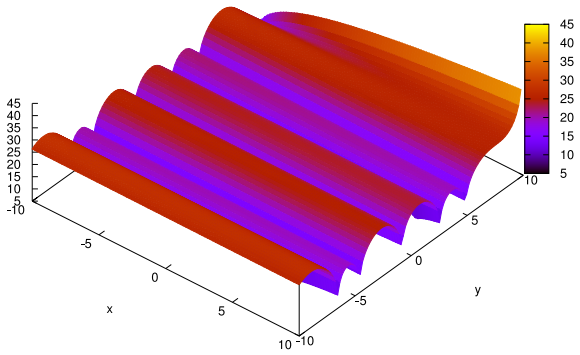
(a) BLX-α beats UNDX（評価値：7116）



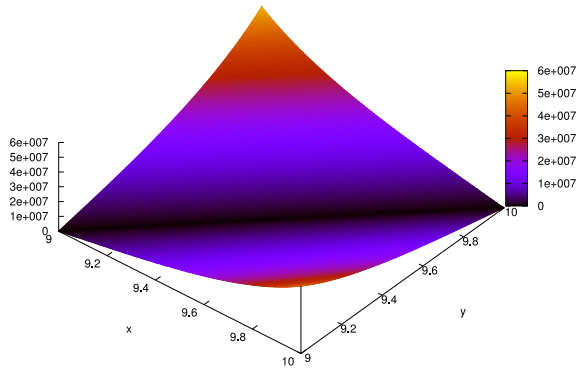
(b) BLX-α beats SPX（評価値：7669）



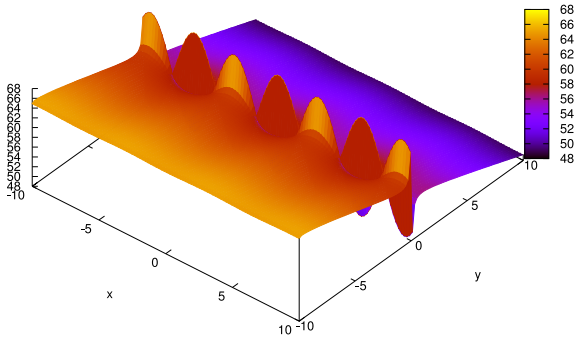
(c) UNDX beats BLX-α（評価値：9540）



(d) UNDX beats SPX（評価値：7854）



(e) SPX beats BLX-α（評価値：10132）



(f) SPX beats UNDX（評価値：5389）

図 2 各交叉間の性能差を強調する探索空間の例（色はその座標での評価値を表す）

x と y の両方の変数を変化させねばならず，変数間依存性がある探索空間であるといえる．また，$x=-10$ の付近で最小値をとるが，$x=-10$ では大きな値をとる．このため，境界外に個体を生成し $x=-10$ の値を得ても最小値を獲得することはできない．この探索空間で最小値を得るためには，$x=-10$ の近辺に個体を生成する必要がある．そのため，変数間依存性に対処でき，正規乱数によって個体を生成する UNDX の性能が高くなる探索空間であると考えられる．実際に探索の過程を解析したところ，BLX-α を用いた場合では $x=-10$ の境界上に多くの個体が生成されてしまい，最小値を得ることができていなかった．表 4 からも UNDX の成功回数が最も多いのが確認できる．

図 2(d) は UNDX が SPX に勝る探索空間の例である．この探索空間は多峰性であり，$(x,y)=(-10,10)$ の付近で最小値をとるが，$y=10$ では大きな値をとる．そのため，最小値を得るためには，$(x,y)=(-10,10)$ の近辺に個体を生成する必要がある．この探索空間は図 2(c) と同様に，正規乱数によって個体を生成する UNDX の性能が高くなる探索空間であると考えられる．

図 2(e) は SPX が BLX-α に勝る探索空間の例である．この探索空間では $x=y$ の直線から離れた座標では評価値が大きくなりすぎるため，図 2(e) では変数の範囲が $[9,10]$ の景観を示している．この探索空間は変数間依存性があり，$x=y$ で最小値となる．表 4 を見ると，SPX の成功回数は BLX-α より多いが，UNDX の成功回数が最も多い．これは，探索空間の生成時には UNDX の性能は考慮していないためではないかと考えられる．

表 4 図 2 の探索空間に各交叉を用いた実数値 GA を適用した際の成功回数（試行回数：30）

	BLX-α	UNDX	SPX
図 2 (a)	30	11	9
図 2 (b)	30	17	20
図 2 (c)	16	27	15
図 2 (d)	21	30	26
図 2 (e)	14	28	24
図 2 (f)	25	19	24

図 2(f) は SPX が UNDX に勝る探索空間の例である．この探索空間では $y = 0$ 付近に最小値をとる領域が存在するが，一方その近傍には大きな値の領域が存在している．また探索空間全体としては y の値が大きくなるほど小さな値をとる．表 4 を見ると，SPX の成功回数の方が UNDX よりも多いが，他の探索空間と比べるとその差は大きくない．

ところで，図 2 に示した探索空間には評価値が $f_{\max} = 10$ よりも大きな領域が多く存在する．この領域に 2 つの実数値 GA の最良個体が存在する間は式 (1) による優劣は付かない．しかし，早い段階でどちらかの実数値 GA が評価値が 10 よりも小さい評価値を発見すれば差を付けることができる．今回は $f_{\max} = 10$ として実験を行った結果，性能差が強調される探索空間として，このような探索空間が得られたが，$f_{\max}$ の値を変えて実験を行うことも考えられる．

次に，生成された探索空間を表す関数について述べる．図 2(b) に示した探索空間を表す関数は次の通りである．

$$\begin{aligned} f(x,y) = & \left(\sin\sqrt{|\sin(x)|} - 10\right) \\ & (9+\sin(\sin(10)))\sin(\sin(\sin(10)))\cos(y) \\ & +10\sqrt{|\sin(x)|}\,(10+\cos(x)) + 10 \\ & -\cos(10\,(10+\cos(x))) \end{aligned} \tag{2}$$

また，図 2(f) に示した探索空間を表す関数は次の通りである．

$$f(x,y) = 20\pi - 8 + \frac{\cos(x)}{y} + \sqrt{|10+y|} - y \tag{3}$$

このように，CGP によって生成された関数は複雑なものもあれば，構成される要素の数が比較的少ないものまで様々である．本章の実験では式 (1) を最大化するような関数の生成を行ったため，一見理解が困難な関数が生成される場合もあった．そのため，5 章では関数を構成するノード数を探索空間の生成時の目的関数に追加し，多目的最適化を行うことで，様々な探索空間を一度の探索で得ることを試みる．

3·3 既存のベンチマーク関数との比較

ここでは，既存のベンチマーク関数に対して各交叉を適用した際の性能差を算出することで，生成された探索空間の妥当性を検証する．関数最適化の代表的なベンチマーク関数として Sphere-d 関数，Rosenbrock 関数，Rastrigin-d 関数の 3 つの関数を採用する．

表 5 2 次元の Sphere-d 関数に対する各交叉間の性能差

		Lose		
		BLX-α	UNDX	SPX
Win	BLX-α	–	59	-42
	UNDX	-59	–	-100
	SPX	42	100	–

表 6 2 次元の Rosenbrock 関数に対する各交叉間の性能差

		Lose		
		BLX-α	UNDX	SPX
Win	BLX-α	–	-1812	-2319
	UNDX	1812	–	-53
	SPX	2319	53	–

Sphere-d 関数は次式で表される単峰性の関数であり，$(d, ..., d)$ で最小値 0 をとる．

$$f(x_1, ..., x_n) = \sum_{i=1}^{n} (x_i - d)^2 \tag{4}$$

Rosenbrock 関数は次式で表される変数間に依存関係のある単峰性の関数であり，$(1.0, ..., 1.0)$ で最小値 0 をとる．

$$f(x_1, ..., x_n) = \sum_{i=1}^{n-1} \{100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\} \tag{5}$$

Rastrigin-d 関数は次式で表される多峰性の関数であり，$(d, ..., d)$ で最小値 0 をとる．

$$\begin{aligned} & f(x_1, ..., x_n) \\ & = 10n + \sum_{i=1}^{n} \{(x_i - d)^2 - 10\cos(2\pi(x_i - d))\} \end{aligned} \tag{6}$$

全ての関数について探索範囲は $-10 \leq x_i \leq 10$ とし，$d = 1.0$ とした．次元数 $n = 2$ としたこれらの関数に対して各交叉間の性能差を式 (1) で算出した値を表 5，表 6，表 7 にそれぞれ示す．実数値 GA の設定などは先の実験と同様である．Sphere-d 関数に関しては各交叉間の性能差は小さく，ほとんど性能差はない．一方，Rosenbrock 関数に対しては，UNDX と SPX の性能差は微小であるが，BLX-α と UNDX，SPX の性能差の値は 1800 から 2300 と比較的大きい．これは，Rosenbrock 関数が変数間に依存関係のある関数であり，UNDX や SPX が変数間の依存関係に対処可能な交叉であるからと考えられる．Rastrigin-d 関数に対しては，Rosenbrock 関数に比べると性能差は小さいが，BLX-α が最も良い性能を示している．3·2 節で生成した探索空間はこれら代表的なベンチマークよりも性能差が大きく，各交叉間の性能差を強調するものになっている．

表 7 2 次元の Rastrigin-d 関数に対する各交叉間の性能差

		Lose		
		BLX-α	UNDX	SPX
Win	BLX-α	–	583	239
	UNDX	-583	–	-344
	SPX	239	344	–

表 8 次元数を変更して探索空間を生成した際の平均評価値

次元数	BLX-α beats UNDX	UNDX beats BLX-α
4	7333	8600
8	6301	7784
16	3284	9549

3·4 多次元の探索空間の生成

文献 [Langdon 07] では生成された探索空間を多次元に拡張することで，多次元の探索空間について検証を行っている．ここでは，CGP が表現する関数への入力数を多次元にすることで多次元の探索空間を直接生成することを試みる．次元数が増すことで問題が難しくなると考えられるため，実数値 GA のパラメータも次元数に併せて変更した．実数値 GA のパラメータは，次元数を n とした時に個体数を $15n$，子個体生成数 $n_{\mathrm{c}} = 10n$，最大世代数 $T = 500$ とした．次元数については，4，8，16 とした場合について検証を行った．実験は BLX-α と UNDX の 2 つの交叉の組み合わせに対して行い，得られた探索空間の評価値の平均を表 8 に示す．次元数 4 と 8 については 10 回の試行，次元数 16 については 5 回の試行の平均値である．

表 8 から，BLX-α が UNDX に勝る探索空間の性能差の値は特に 16 次元の場合に大きく減少していることがわかる．これは，CGP の入力次元数が増加すると変数間の依存関係が弱い探索空間の生成が困難になるからではないかと考えられる．一方，UNDX が BLX-α に勝る探索空間の性能差の値は入力次元数が増加しても減少していない．これは，変数間に依存関係のある関数では UNDX のほうが BLX-α よりも高い性能を示すため，CGP の入力次元数が増加しても性能差が強調される探索空間が生成し易かったからだと考えられる．これらの結果から，多次元の場合にも性能差が強調される探索空間の生成が可能であることが示された．

4. 世代交代モデルの性能差を強調する探索空間の生成実験

ここでは，実数値 GA の世代交代モデルに着目し，世代交代モデルを変えた 2 つの実数値 GA を用いて探索空間の生成を行う．

4·1 実 験 の 設 定

CGP の各パラメータ値は表 2 に示したものを用いた．世代交代モデルには SGA，SGA with Elite，MGG を使用し，実数値交叉には，BLX-α を採用した．実数値 GA のパラメータは各世代交代モデルで共通のものを用いることとし，個体数を 20，子個体生成数 $n_{\mathrm{c}} = 20$，最大世代数 $T = 500$ とした．各世代交代モデル間で 1 世代の個体評価回数を同一にするために，ここでは個体数を 20 とした．ここでも，CGP の各個体が表現する探索空間は 2 次元であるため，このパラメータで十分な探索が行えることを予備実験によって確認している．各世代交代モデルの組み合わせににつき，それぞれ 10 回の CGP による探索空間の生成を行った．

表 9 10 回の試行で生成された世代交代モデルの性能差を強調する探索空間の平均評価値（括弧内の数値は，左が標準偏差，右が評価値が 5000 以上となった試行回数）

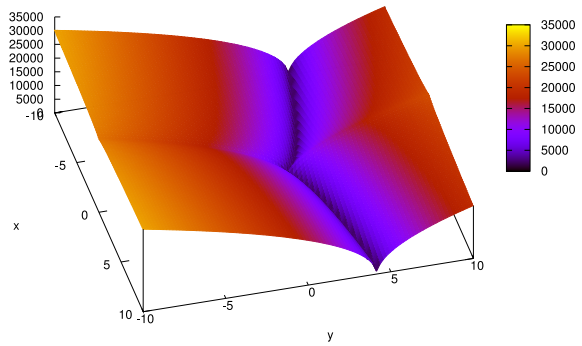
		Lose		
		SGA	MGG	SGA with Elite
Win	SGA	–	9103	4523
		–	(1018, 10)	(1209, 4)
	MGG	10045	–	9588
		(274, 10)	–	(670, 10)
	SGA with Elite	9676	9341	–
		(430, 10)	(448, 10)	–

4·2 実験結果と考察

10 回の試行を行って得られた探索空間の評価値の平均を表 9 に示す．表中の数値は，各行の世代交代モデルを用いた実数値 GA が各列の世代交代モデルを用いた実数値 GA よりも性能が高くなる探索空間を生成した際の評価値を表している．また，括弧内の値は標準偏差と評価値が 5000 以上となった実験回数である．この表から，どの組み合わせに対しても，性能差が大きくなる探索空間の生成が行えていることが確認できる．特に，SGA が MGG より性能が高くなる探索空間以外は評価値の平均が高く，今回の設定では生成し易かったと考えられる．

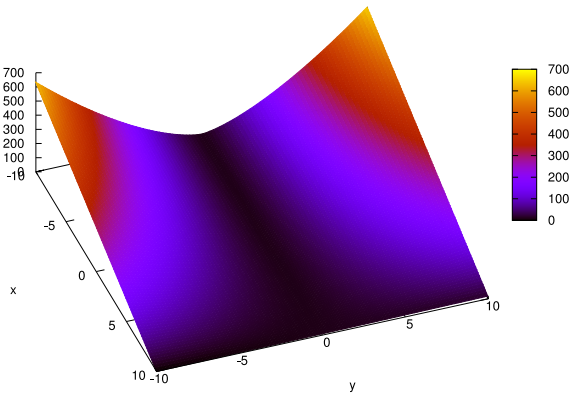
図 3 は SGA のほうが MGG より性能が高くなる探索空間，MGG のほうが SGA より性能が高くなる探索空間，SGA のほうが SGA with Elite より性能が高くなる探索空間，SGA with Elite のほうが SGA より性能が高くなる探索空間の例をそれぞれ示したものである．これらは 10 回の試行で生成された関数の中で，関数を構成する要素数が比較的少なかったものを著者が選択したものである．図中の探索空間の高さはその座標での評価値を表し，色についても同様に評価値を表しており，黄色は高い値を青色は低い値を表している．また，表 10 は図 3 の探索空間に対して各世代交代モデルを用いた実数値 GA で 30 回の探索を行った際の成功回数を示したものである．ここでも 1.0×10^{-8} より小さい値を得ることができた試行を成功としてカウントしている．

図 3 (a) は SGA のほうが MGG より性能が高くなる探索空間の例である．この探索空間は $\sqrt{|x|} - y + 1 = 0$ となる座標で最小値をとる．この探索空間に対して MGG



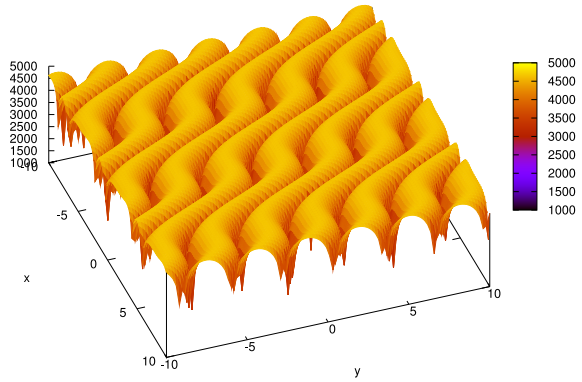
$$f(x,y) = 8000\sqrt{|\sqrt{|x|} - y + 1|}$$

(a) SGA beats MGG（評価値：8680）



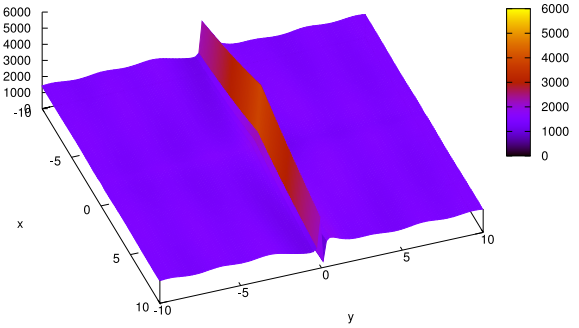
$$f(x,y) = \sqrt{|y^3|}(10 - x) + y + 10$$

(b) MGG beats SGA（評価値：9650）



$$f(x,y) = \frac{1}{\cos(\sqrt{10})+1}|\sin(x - \frac{y+\cos(x)}{\cos(\sqrt{10})})|^{\frac{1}{8}}$$

(c) SGA beats SGA with Elite（評価値：6430）



$$f(x,y) = (9+\pi)(\frac{\cos(\sqrt{|xy|})}{y} - \cos(\sin(y)) - 10)^2$$

(d) SGA with Elite beats SGA（評価値：9920）

図 3 各世代交代モデル間の性能差を強調する探索空間の例（色はその座標での評価値を表す）

で探索を行うと最適値付近に個体が集まるが，収束が足りずに最適値を得る前に探索が終了してしまう．一方，SGA では探索の早い段階から個体が収束し，局所的な探索が進むため最適値を発見することに成功している．また，この探索空間に対しては，MGG でも世代数を十分大きくすると最適解を得ることができることを確認している．

図 3 (b) は MGG のほうが SGA より性能が高くなる探索空間の例である．この探索空間は $(10, -10)$ で最小値をとり，$y = 0$ では x の値によらず評価値 10 をとる．この探索空間に対して SGA で探索を行うと個体が $y = 0$ のある座標に収束してしまい，探索が進まない．一方，MGG による探索では集団の多様性が保たれているため，個体が一点に集中することなく最適値を発見することができている．

図 3 (c) は SGA のほうが SGA with Elite より性能が高くなる探索空間の例である．この探索空間は多峰性であり，複数の座標で最小値をとる．この探索空間に対しては SGA の方が SGA with Elite よりも性能が高いが，表 10 から SGA による探索でも成功回数は 6 回と少ない．SGA と SGA with Elite の探索過程を観察すると，どちらも探索に失敗する時には局所解に集団が収束してしまっている．SGA with Elite ではエリート個体が必ず次世代に保存されるため，探索序盤に発見した局所解の影響が大きいと考えられる．一方，SGA ではエリート個体が保存されないため，SGA with Elite と比べると局所解から抜け出せる可能性がある．この差が図 3 (c) の探索空間における成功回数の違いに影響していると考えられる．

図 3 (d) は SGA with Elite のほうが SGA より性能が高くなる探索空間の例である．この探索空間は全体として多峰性の景観をもっており，$y = 0$ の付近に非常に急な山と谷があり，大きな値と小さな値をとる領域が存在する．エリート保存戦略を用いないでこの探索空間に対して探索を行うと，一度得られた良い個体が保存されず，局所解に集団が収束してしまうと考えられる．

5. 多目的進化アルゴリズムによる探索空間の生成実験

ここまでの実験では式 (1) だけを目的関数として探索空間の生成を行った．そのため，複雑な関数が生成される傾向が見受けられた．複雑な関数になるほど性能差が

表 10 図 3 の探索空間に各世代交代モデルを用いた実数値 GA を適用した際の成功回数（試行回数：30）

	SGA	MGG	SGA with Elite
図 3 (a)	29	0	30
図 3 (b)	2	30	1
図 3 (c)	6	0	1
図 3 (d)	2	30	26

表 11 多目的進化アルゴリズムによって生成された非劣解の関数群（BLX-α が UNDX より性能が高くなる探索空間）

生成された関数	ノード数	性能差
$\sqrt{\lvert y\rvert}$	3	1570
$\sqrt{\lvert 2y\rvert}$	4	1593
$\sqrt{\lvert xy\rvert}$	5	1704
$\sqrt{\lvert y(y+\cos(y))\rvert}$	6	1775
$\sqrt{\lvert y(y-\sqrt{\lvert x\rvert})\rvert}$	7	2122
$100\sqrt{\lvert\sin(y)\rvert}-\cos(y)$	9	4272
$100\sqrt{\lvert\cos(y)\rvert}-\cos(y)$ $-\cos(\sqrt{\lvert\cos(y)\rvert})$	10	4750
$100\sqrt{\lvert\cos(y)\rvert}-\cos(y)-\cos(x)$	11	4940
$100\sqrt{\lvert\sin(y)\rvert}-\cos(x\sin(x))$	12	5782

拡大する探索空間が表現できるようになるが，探索空間の特徴を理解することが難しくなると考えられる．そこで本章では，探索空間の生成時の目的関数に CGP の各個体中で実際に使用されているノード数を加え，2 目的問題に問題を拡張する．目的関数の一つ目には式 (1) を，二つ目には CGP の各個体中で実際に使用されているノード数を用いる．つまり，式 (1) を最大化，CGP の各個体中で実際に使用されているノード数を最小化することを目的とする．これによって，単一の探索空間だけではなく複数の探索空間を一度に生成し，どのような要素が性能差を大きくするのかを観察する．[Langdon 07] などの従来研究では単一の探索空間を生成しており，複数の探索空間を多目的進化アルゴリズムによって生成し，そこから知見を得ようとする試みは，本論文の大きな特徴の一つであるといえる．最適化には多目的進化アルゴリズムの Strength Pareto Evolutionary Algorithm 2（SPEA2）[Zitzler 01] を用いる．SPEA2 では目的関数空間における個体の支配関係と密集度を用いて各個体の評価値を決定する．また，アーカイブに優良個体を保持することで，エリートの保存を行う．SPEA2 のパラメータは，個体数を 5，アーカイブサイズを 20 とした．ここでは実数値交叉の性能差を利用して探索空間の生成を行う．実数値交叉には BLX-α と UNDX を使用し，BLX-α が UNDX より性能が高くなる探索空間の生成と，UNDX が BLX-α より性能が高くなる探索空間の 2 種類の探索空間の生成を行う．また，実数値 GA の世代交代モデルには MGG を用いる．CGP の各パラメータ値は表 2 に示した値を用い，実数値 GA の各パラメータ値は 3·1 節で説明した値を使用する．

表 11 と表 12 に多目的進化アルゴリズムによって得ら

表 12 多目的進化アルゴリズムによって生成された非劣解の関数群（UNDX が BLX-α より性能が高くなる探索空間）

生成された関数	ノード数	性能差
$10-y$	4	37
$\sqrt{\lvert y-x\rvert}$	5	2286
$\lvert y-x\rvert^{0.25}$	6	2640
$\lvert x(y-x)\rvert^{0.25}$	7	4268
$\sqrt{\lvert(y-x)(\cos(x^2)+x^2)\rvert}$	9	6092
$\lvert(y-x)(\cos(x^2)+x^2)\rvert^{0.25}$	10	7144

れた非劣解が表す関数をそれぞれ示す．表から確認できるように，ノード数が多くなるに従って性能差が大きくなり，各関数には共通の要素が含まれているのが確認できる．表 11 から，多峰性が BLX-α と UNDX の性能差を拡大させているのがわかる．図 4 は生成された BLX-α が UNDX より性能が高くなる探索空間のうちノード数が 7，9，12 のものの景観をそれぞれ示したものである．これらの探索空間は変数間依存性がほとんどなく，多峰性の景観となっている．図 5 は生成された UNDX が BLX-α より性能が高くなる探索空間のうちノード数が 6，7，10 のものの景観をそれぞれ示したものである．これらの探索空間は $x=y$ で最小値をとり，変数間依存性がある探索空間であるといえる．ノード数が多くなるに従い，$x=0$ の付近に値の小さな領域が形成されていくのが確認できる．また，表 12 から平方根などの要素によって性能差が拡大しているのが確認でき，鋭い評価値の変化をもつ探索空間が UNDX と BLX-α 性能差を拡大することがわかる．このように多目的最適化によって得られた複数の探索空間を観察することで，どのような要素が加えられると 2 つの実数値 GA の性能差が大きくなるのかを確認することができる．

6. ま と め

本論文では，交叉や世代交代モデルを変えた実数値 GA の性能差を強調する探索空間の生成を行った．まず 3 種類の交叉（BLX-α，UNDX，SPX）を用いて探索空間の生成を行い，全ての組み合わせに対して一方の交叉を用いた実数値 GA が他方の交叉を用いた実数値 GA を上回る探索空間が生成可能であることを確認した．また，世代交代モデルの差異を利用した探索空間の生成を行い，一方の世代交代モデルの性能が高くなる探索空間の生成に成功した．さらに，生成された探索空間を観察することで，性能差が生じる原因を考察した．その後，探索空間の生成時の目的関数に CGP のノード数を加え，多目的進化アルゴリズムによって複数の探索空間を一度に生成し，得られた複数の探索空間を観察することで，どのような要素が加えられると 2 つの実数値 GA の性能差が大きくなるのかを考察した．

本論文では交叉を変えた実数値 GA と世代交代モデルを変えた実数値 GA の性能差を強調する探索空間をそれ

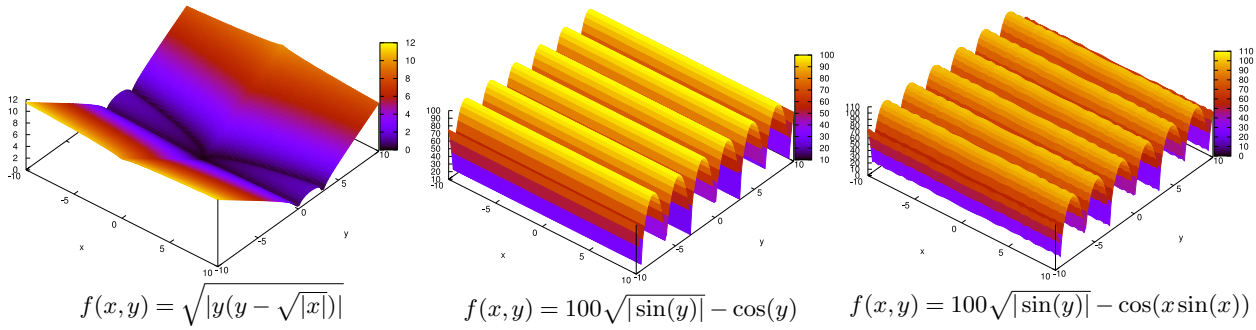


図 4 多目的進化アルゴリズムによって生成された BLX-α が UNDX より性能が高くなる探索空間の例（ノード数：7，9，12）

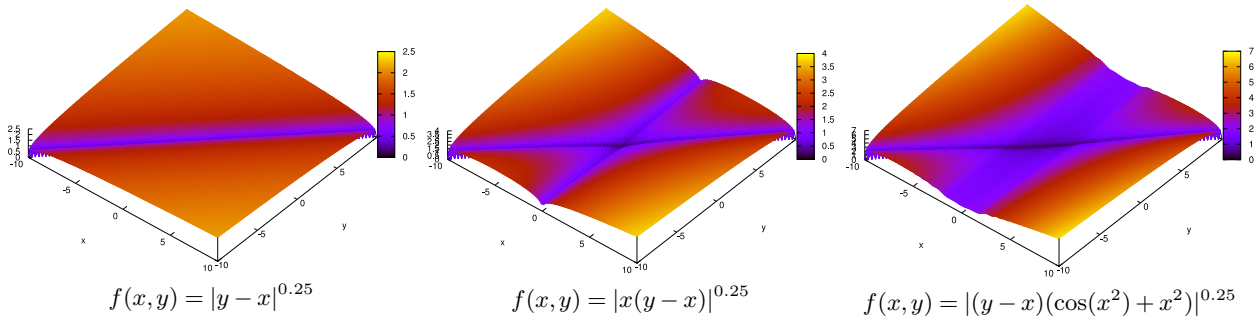


図 5 多目的進化アルゴリズムによって生成された UNDX が BLX-α より性能が高くなる探索空間の例（ノード数：6，7，10）

ぞれ独立に生成した．しかし，実際の GA の設計では，交叉や世代交代モデルなど様々な要素を組み合わせることで GA 全体の性能の向上を図る．そのため，交叉と世代交代モデルの両方を変えた実数値 GA によって探索空間の生成を行い，交叉と世代交代モデルの相互作用について検討を行うことも興味深い．さらに，交叉や世代交代モデルの違いだけでなく各パラメータの違いによる性能差を強調する探索空間の生成を行うことや，実数値 GA 以外の進化計算法や多目的進化アルゴリズムの性能差を利用して探索空間の生成を行うことが考えられる．最後に，本研究によって得られた成果が新たな探索アルゴリズムの開発の手助けになることや新たな問題領域の発見に貢献できることを期待している．

◇ 参 考 文 献 ◇

[Eshelman 93] Eshelman, L. J. and Schaffer, J. D.: Real Coded Genetic Algorithms and Interval-Schemata, in *Foundations of Genetic Algorithms 2*, pp. 187–202, Morgan Kaufmann Publishers (1993)

[Hansen 01] Hansen, N. and Ostermeier, A.: Completely Derandomized Self-Adaptation in Evolution Strategies, *Evolutionary Computation*, Vol. 9, No. 2, pp. 159–195 (2001)

[Hemert 06] Hemert, van J. I.: Evolving Combinatorial Problem Instances That Are Difficult to Solve, *Evolutionary Computation*, Vol. 14, No. 4, pp. 433–462 (2006)

[樋口 01] 樋口 隆英, 筒井 茂義, 山村 雅幸：実数値 GA におけるシンプレクス交叉の提案, 人工知能学会論文誌, Vol. 16, No. 1, pp. 147–155 (2001)

[Kennedy 95] Kennedy, J. and Eberhart, R.: Particle Swarm Optimization, in *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pp. 1942–1948 (1995)

[Kita 99a] Kita, H. and Yamamura, M.: A Function Specialization Hypothesis for Designing Genetic Algorithms, in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '99)*, pp. 579–584 (1999)

[喜多 99b] 喜多 一, 小野 功, 小林 重信：実数値 GA のための正規分布交叉に関する理論的考察, 計測自動制御学会論文集, Vol. 35, No. 11, pp. 1333–1339 (1999)

[喜多 99c] 喜多 一, 山村 雅幸：機能分担仮説に基づく GA の設計指針, 計測と制御, Vol. 38, No. 10, pp. 612–617 (1999)

[小林 09] 小林 重信：実数値 GA のフロンティア, 人工知能学会論文誌, Vol. 24, No. 1, pp. 147–162 (2009)

[Koza 92] Koza, J. R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press (1992)

[Langdon 07] Langdon, W. B. and Poli, R.: Evolving Problems to Learn About Particle Swarm Optimizers and Other Search Algorithms, *IEEE Transactions on Evolutionary Computation*, Vol. 11, No. 5, pp. 561–578 (2007)

[Miller 00] Miller, J. F. and Thomson, P.: Cartesian Genetic Programming, in *Genetic Programming: 3rd European Conference, EuroGP 2000*, Vol. 1802 of *LNCS*, pp. 121–132, Springer-Verlag (2000)

[Miller 06] Miller, J. F. and Smith, S. L.: Redundancy and Computational Efficiency in Cartesian Genetic Programming, *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 2, pp. 167–174 (2006)

[Oltean 04] Oltean, M.: Searching for a Practical Evidence of the No Free Lunch Theorems, in *Biologically Inspired Approaches to Advanced Information Technology: First International Workshop, BioADIT 2004*, Vol. 3141 of *LNCS*, pp. 472–483, Springer-Verlag (2004)

[Ono 97] Ono, I. and Kobayashi, S.: A Real-coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover, in *Proceedings of the 7th International Conference on Genetic Algorithms (ICGA '97)*, pp. 246–253, Morgan Kaufmann Publishers (1997)

[小野 99] 小野 功, 佐藤 浩, 小林 重信：単峰性正規分布交叉 UNDX を用いた実数値 GA による関数最適化, 人工知能学会誌, Vol. 14, No. 6, pp. 1146–1155 (1999)

[Poli 08] 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)

[佐藤 97] 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)

[Storn 97] Storn, R. and Price, K.: Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, Vol. 11, No. 4, pp. 341–359 (1997)

[Tsutsui 99] Tsutsui, S., Yamamura, M., and Higuchi, T.: Multi-parent Recombination with Simplex Crossover in Real Coded Genetic Algorithms, in *Proceedings of the Genetic and Evolutionary Computation Conference 1999 (GECCO '99)*, pp. 657–664, Morgan Kaufmann Publishers (1999)

[Zitzler 01] Zitzler, E., Laumanns, M., and Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm, Technical Report 103, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Zurich, Switzerland (2001)

〔担当委員：筒井 茂義〕

2010 年 8 月 7 日 受理

著 者 紹 介

白川 真一（一般会員）

2005 年 3 月横浜国立大学工学部電子情報工学科卒業．2007 年 3 月同大学大学院環境情報学府情報メディア環境学専攻博士課程前期修了．2009 年 3 月同博士課程後期修了．2009 年 3 月日本学術振興会特別研究員 PD．2010 年 4 月より株式会社富士通研究所勤務．博士（工学）．進化計算，画像処理等の研究に従事．進化計算シンポジウム 2009 最優秀発表賞，IEEE Computational Intelligence Society (CIS) Japan Chapter 2009 年 Young Researcher Award 受賞．

矢田 紀子

2003 年 3 月横浜国立大学工学部電子情報工学科卒業．2005 年 3 月同大学大学院環境情報学府情報メディア環境学専攻博士課程前期修了．2008 年 3 月同博士課程後期修了．2008 年 4 月より横浜国立大学大学院環境情報研究院特任教員（研究教員）．博士（工学）．視覚情報処理，画像処理，神経回路網，進化計算等の研究に従事．

長尾 智晴（一般会員）

1985 年東京工業大学大学院博士後期課程中退．同年同大学工学部附属像情報工学研究施設助手．同大学工学部助教授を経て，2001 年横浜国立大学大学院環境情報研究院教授，現在に至る．工学博士。画像処理，進化計算，神経回路網，分散人工知能等の研究に従事．