染色体距離に基づく種分化を導入したLinear GP
Linear GP with the speciation based on chromosomal-distance

檜垣 紀志
Noriyuki Higaki
広島市立大学大学院情報科学研究科
Email: nhigaki @ ints.info.hiroshima-cu.ac.jp

原 章
Akira Hara
広島市立大学大学院情報科学研究科
Email: ahara @ hiroshima-cu.ac.jp

高濱 徹行
Tetsuyuki Takahama
広島市立大学大学院情報科学研究科
Email: takahama @ info.hiroshima-cu.ac.jp

***Abstract*—Linear Genetic Programming (LGP) has been proposed as an extended GP, in which the individual's genotype is represented by a linear array. However, LGP also shows premature convergence in early stage of search as well as GP. To solve the problem, we proposed the species-based LGP (SLGP). The speciation is performed based on the chromosomal-distance. By dividing the population to sub-population, SLGP could perform more global search. As the result of experiments, SLGP showed better performance than normal LGP.**

I. はじめに

遺伝的プログラミング (Genetic Programming：GP)[1] は進化的計算の内の一つであり，交叉や突然変異などの遺伝的操作によって個体を進化させることで，構造的表現を環境に適応させていく．GP では，個体の表現法として木構造が用いられているため，処理の過程でポインタ操作が必要となり時間計算量および空間計算量が増大するという問題が懸念されている．それを解決する一つの手法として Linear GP(LGP)[2][3] が提案されている．LGP では，個体に線形配列が用いられており，機械語に似た命令コード（以下コード）が複数連なることで複雑な関数を表現することが可能である．表現能力は GP と同等のまま，遺伝子型が線形配列で表現されているため，個体評価時のポインタ操作が必要なくなり処理速度は GP と比べより高速になっている．また，その性能の向上も示されている．しかしながら，GP と同様に，LGP でも適合度が早期収束する問題が起こる．これは解探索の序盤にて局所解に陥った際に，そこから抜け出せないことに起因していると考えられる．LGP では特有の個体構造を持つため，遺伝的操作の適用によって個体の表現が大きく変わることがある．このときに，将来的には適合度が向上する見込みのある個体であっても，一時的な個体の適合度低下により淘汰され，個体進化の停滞に繋がる可能性がある．よって大域的な探索が行えるようにする必要があると考えた．

Differential Evolution(DE) や Particle Swarm Optimization(PSO) など他の進化的計算手法では，効果的に複数解の発見を行う方法として種分化モデル [4], [5] が導入されている．個体の持つ位置情報から個体間の距離を求め，その距離を基に種分化することで個体群を複数の種に分けている．そして，解空間を分割し種毎に探索を行うことで，一度の探索で複数の最適解を発見する．一方 LGP では個体に位置情報がないため，上記手法と同様にして空間上の距離を求めることが困難である．よって Optimal Symbolic Alignment(OSA) Distance と呼ばれる編集距離を用いて，個体の持つ染色体間の距離を計測し，これを個体間の距離とする．

本論文では，染色体距離に基づく種分化モデルを LGP に導入することにより，母集団を複数のサブ集団へと分割し，解空間を各種が独立的に探索できるように改良した．これにより，早期的な適合度収束を回避し，より精度の高い解を求めることを目的としている．

II. Linear GP

GP の個体の遺伝子型は木構造によって表現されている．これにより，個体の評価および遺伝的操作時にポインタ操作が必要となり，また個体の成長に伴い計算量が増大する．この計算量を減少させるために，Linear GP（LGP）と呼ばれる手法が提案された．

A. LGP の個体表現

LGP での個体には線形配列が用いられ，命令コード（以下コード）が複数連なることで個体が表現される．各コードは，低級言語で使用されるレジスタ演算を模倣した簡単な演算式で表される．単一コードは 4 つの要素から構成され，第一要素が演算子，第二要素が第一オペランド，第三要素が第二オペランド，第四要素が目的レジスタ（演算結果を格納するレジスタ）を表している．遺伝子型の例を図 1 に，またその遺伝子型に対応する表現型を図 2 に示す．

ADD	a	1	a	MUL	a	a	b	SUB	b	2	a

図 1. LGP の個体の遺伝子型

```
1:  r[a] = r[a] + 1;
2:  r[b] = r[a] * r[a];
3:  r[a] = r[b] - 2;
```

図 2. LGP の個体の表現型

図 2 において r[a] が入力，および出力を表すレジスタとすると，このコードを先頭から逐次的に実行することで，個体の出力として関数 ($f(x) = (x+1)^2 - 2$) を得る．

B. LGP の遺伝的操作

LGP の遺伝的操作は，GP や GA と同様に交叉や突然変異等があるが，遺伝子の構造が配列であるため，GA に似た操作である．

- 交叉
  LGP の交叉には 1 点交叉または 2 点交叉などがあり，本論文では 2 点交叉を用いた．交叉は，まず親となる 2 個体を選択後，各々の親個体で 2 点をランダムに決定し，2 点間に存在するコードをお互いに交換することで子個体を 2 つ生成する．
- 突然変異
  LGP の突然変異は，作用する突然変異点によってその操作の名称が変わる．大きく分けると，マクロ突然変異とミクロ突然変異に分類することができる．マクロ突然変異はコード単位で作用し，単一のコードを挿入，個体から単一のコード削除，もしくは任意の単一コードを別のコードへと置換，の 3 種が存在する．また，ミクロ突然変異では，任意の単一コードの 1 要素のみを選択し，その要素を変化させる．

C. アルゴリズム

LGP のアルゴリズムは次の通りである．また，遺伝的操作において選択された個体は，排他的に交叉もしくは突然変異のどちらか一方が適用される．

1) 個体群の初期化
2) 各個体の評価
3) 個体の選択
4) 遺伝的操作 (交叉 or 突然変異)
5) 次世代の個体数分の個体が生成されていなければ 3. へ
6) エリート保存
7) 個体群の置き換え
8) 最大世代数に到達していなければ 2. へ

III. 種分化モデル

種分化モデルとは，解空間に存在する個体群を，個体間の距離を基に分割し種を形成する手法である．そして，種に分けられた身内の個体だけで独立的に探索を進めていく．これにより，種毎に異なる進化が可能になる．

この種分化のプロセスを LGP に応用することで，探索空間を種によって分割し幅広く解の探索を行えると考えた．そして，早期的に局所解へ陥ることによる適合度停滞を避け，高精度の解を得ることを目指す．本論文で用いた種分化のアルゴリズムを次に示す．

1) 個体群を適合度順にソートする
2) 種に未所属で，最も良い個体を種の支配者とする
   a) 種に未所属な個体と支配者との距離 d を計測する
   b) 距離 d が小さい順にソートする
   c) 距離の小さい個体から順に，あらかじめ設定した種の個体数となるまで種に属させる
3) 種に未所属な個体が存在すれば 2. へ戻り，次の種を形成する

これにより，全ての個体がいずれかの種に属するようになるまで繰り返し種分化を行う．実際に種分化されたときの様子を図 3 に示す．ここでは A,B,C の個体がそれぞれの種の支配者となっており，それぞれの適合度が $F_A < F_B < F_C$ で A が最良であるとする．また，1 つの種に属することのできる個体数を 4 と設定した場合で考える．A,B,C の個体から順に種を形成していくこととなり，A と B の両種に属することのできる個体が存在するが，その時点で A の種には 3 個体しか含まれていないため，実際には B との距離が近いが結果として A の種に属することになる．同様に B,C の両種に含まれる個体が存在した場合でも，B の種に空きがあれば B に所属するようになる．

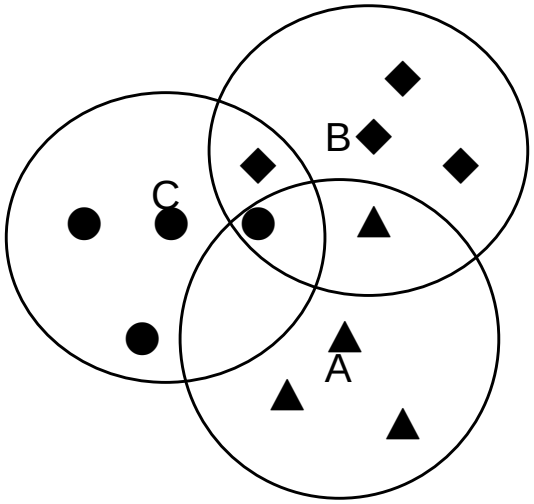

図 3. 種分化の様子

IV. 種分化を導入した LGP の提案

LGP などの進化的計算手法において，探索早期に局所解へと陥り，適合度が停滞するという問題がある．本論文では，LGP に種分化を組み込むことで，探索空間を分割して探索を行う．これにより種が独立的に進化を行うことで，1 つの種が局所解に陥った場合でも，他の種は別の空間を探索できる．これにより，適合度の早期収束に直面することを避け，結果として解の精度を高めることを目的としている．

A. LGP での種分化

本論文の LGP への種分化適用はまだ初歩的な段階であるため，文献 [4], [5] で使用されている一般的な種分化法とは種の形成方法が異なる．実際に異なる部分として以下の 2 つがある．

- 縄張り半径を廃止し，距離の近いものから種に属させる
- あらかじめ種の数を指定し，固定の種数で分化する

まず，1 つ目の「縄張り半径の廃止」について説明する．従来の種分化には縄張り半径 R が存在する．支配者との距離 d がこの R の値以下であった場合に種を形成するものとなっている．関数最小化問題で利用される種分化は個体が実数空間上にあり縄張り半径の決定が比較的容易であったが，LGP では個体は記号で表現されているため，この値の決定が非常に難しいと考えられる．よって，本論文では，単純に支配者との距離が近い個体から順に種を形成していくアルゴリズムに変更している．次に 2 つ目の「種数の固定」については，種に属する個体数に偏りが発生すると，それぞれの種で個体の進化する速度に大きく影響をきたすため、今回は種の数を固定し各種に含まれる個体数を同じにした．

B. 染色体距離の計測方法

GA や PSO，DE などの手法では個体は実数空間上に存在するため，個体の距離をユークリッド距離にて容易に計算することが可能である．しかしながら，LGP の個体は実数空間には存在しないため，個体の持つ染色体（コード）を利用して記号的な距離を計測し，これを個体間の距離とみなす．染色体の距離を求めるために，本論文では Optimal Symbol Alignment (OSA) Distance[6] を用いた．この手法は記号列においてある記号の現れる位置やその出現頻度を基に距離を算出している．現実世界における染色体間の距離や類似度を求める場合で，通常の編集距離よりも，高い精度で分類することが可能であり，またその計算量も小さくなっている．LGP では個体が進化する際にコードが長くなる傾向があるため，その影響による計算量の増大を軽減することができると考えられる．

個体 A,B 間の染色体距離 $d(A,B)$ は次の式により求めることができる．

$$d(A,B) = \sum_{x \in X_A \cup X_b B} d(x,A,B) \tag{1}$$

$$d(x,A,B) = \begin{cases} |A_x| & \text{if } x \in X_A - X_B \\ |B_x| & \text{if } x \in X_B - X_A \\ f(x,A,B) & \text{if } x \in X_A \cap X_B \end{cases} \tag{2}$$

$$f(x,A,B) = (|B_x| - |A_x|) + \frac{1}{n_{AB}} \sum_{i_h \in A_x} |i_h - \mathrm{pj}(i_h,A,B)| \tag{3}$$

数式に必要な情報を以下に説明する

x	：コードの要素
A_x	：個体 A に存在する要素 x
X_A	：個体 A に存在する要素 x の集合
$\|A_x\|$	：個体 A に存在する要素 x の個数
n_A	：個体 A の長さ
n_{AB}	：$\max\{n_A, n_B\}$
i_h	：要素の現れる位置
$\mathrm{pj}(i_h,A,B)$	：プロジェクション関数

(2) 式に示したように，ある要素が一方の個体にしか出現しない場合はその数がそのまま距離に加えられる．ある要素が個体 A,B に共通して出現する場合は (3) 式にて計算する．このとき，$|A_x| \leq |B_x|$ であると仮定する．そして (3) 式の後半部分，プロジェクション関数にて共通要素の位置がマッチングするように合わせることで位置合わせのコストを求める．この位置がどれくらい離れているかで距離が増減する．

プロジェクション関数の計算法を次に示す．個体 A に出現した要素 A_c の位置を $i_1 < i_2 < \ldots < i_{|A_c|}$ としたとき，位置合わせのコスト $|i_1 - j_1| + \ldots + |i_{|A_c|} - j_{|A_c|}|$ が最小となるように，個体 B でも出現する要素 B_c の位置 $j_1 < j_2 < \ldots < j_{|A_c|}$ を $|A_c|$ 個選択することである．そして，位置合わせができず，余った要素は 3 式の前半部分 $|B_c| - |A_c|$ のように，そのまま距離として加えられる．

例としてコード列 A : 13122 と B : 12211 の計算を示す．共通文字は 1 と 2 であり，1 の位置はそれぞれ $A : \{1,3\}, B : \{1,4,5\}$ となる．プロジェクション関数を適用すると位置合わせコストは $|1-1| + |4-3| = 1$ が最小であり，B 中の文字 1 が 1 つ余るため，$f(1,A,B) = 1 + 1/5 = 1.2$ となる．文字 1 と同様にして文字 2 では，$f(2,A,B) = 0 + 4/5 = 0.8$ となる．文字 3 は A にしか存在しないため，$d(3,A,B) = 1$ となり，全体として A と B の距離はこれらの総和 $1.2 + 0.8 + 1 = 3$ となる．

LGP のコードは 4 要素から構成され，それぞれの要素が異なる (演算子，レジスタ) 意味を持っているため，上記の数式により個体のコードを一括して計算することが難しい．そのため，本論文ではコードの各要素毎に OSA Distance を適用し，その総和を個体の距離とみした．

C. 種分化 LGP(SLGP) アルゴリズム

SLGP のアルゴリズムは次の通りである．SLGP でのエリート保存は，種毎に 1 個体のみ行うものとして，種の最悪個体をその種の最良個体と置き換える．このとき，現世代における種内の全ての個体が，前世代の種のエリート個体より適合度が良い場合は置き換えを行わない．

1) 個体群の初期化
2) 各個体の評価
3) 種分化
   a) 適合度順に個体をソート
   b) 支配者を決定し，距離の近い個体から順に種に属させる

   以下 4)-5) を種毎に行う
4) 個体の選択
5) 遺伝的操作 (交叉 or 突然変異)
6) 次世代の個体数分の個体が生成されていなければ 3. へ
7) 種毎にエリート保存
8) 個体群の置き換え
9) 最大世代数に到達していなければ 2. へ

V. 実験

A. パラメータ設定

従来手法 (LGP) と提案手法 (SLGP) の性能を比較するために，関数近似問題の実験を行う．本実験で扱った問題は $sinepoly$ であり，そのパラメータを表 I に示す．また，種分化パラメータおよび，LGP の基本となるパラメータをそれぞれ表 II,III に示している．

表 I
$sinepoly$ のパラメータ

目的関数	$sin(x) \times x + 5$
定義域	[-5,5]
値域	[0,7)
入出力レジスタ	r[0]
計算用レジスタ数	r[1],$\cdots$,r[4]
総レジスタ数	5
訓練データ数	101
演算子種類	$\{+,-,\times,/,x^y\}$
定数	$\{1,2,...,9\}$
適合度関数	SSE

B. 実験結果

LGP と SLGP を $sinpoly$ 問題に適用し，関数 $sin(x) \times x + 5$ の同定を行った．SLGP では種の数を 5,10,20 の 3 通りで実験を行い性能比較を行う．また，SLGP の種数 20 の場合においては，トーナメントサイズが 8 のままだと選択圧が大

表 II
種分化パラメータ

種分化間隔	1
種の数	5 / 10 / 20
種の個体数	200 / 100 / 50
距離関数	OSA Distance

表 III
LGP の基本パラメータ

最大世代数および個体数	1000
コード数下限, 上限	[10, 200]
初期コード数	10 ～ 30
トーナメントサイズ	8
交叉率	0.7
突然変異率	0.3
マクロ突然変異率	0.4
挿入：削除：置換率	0.6：0.3：0.1
ミクロ突然変異率	0.6

き過ぎるため，種に含まれる個体数の 10%，つまり 5 と設定している．これにより，高頻度で同じ個体が選択されることを避ける．LGP と SLGP の適合度は共に 30 回試行の平均値によって求め，そのときの標準偏差も求めた．その結果を表 IV に，LGP と SLGP の適合度の推移を図 4 に示す．

表 IV
適合度と標準偏差 (30 回試行の平均値)

	LGP	SLGP		
種の数	-	5	10	20
fitness	11.8276	8.37307	5.595	4.82896
std.	25.0012	17.9792	10.6659	8.05474

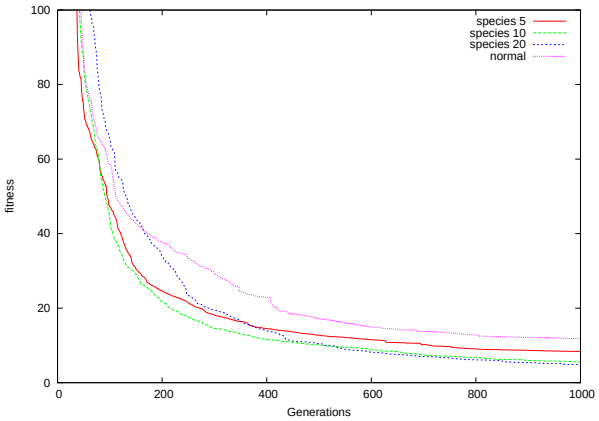

図 4.　適合度の推移

表 IV より，LGP では標準偏差の値が大きく，各試行によって求められる解にバラつきがあるのが伺える．それが，SLGP では 3 種類ともの実験で LGP を上回る性能を示すことができた．また，SLGP では種の数を増やすにつれ，適合度および標準偏差の値の向上が見込めた．これは，種の数が増えることにより，探索する空間が分割され個体が分散することで，局所解に陥る可能性が減少したからだと考えられる．図 4 より，LGP では適合度が速い段階 (100 世代辺り) で収束しだしているが，SLGP では個体が分散されるため，早期収束による適合度の停滞が避けられている．種の数が 20 では収束速度が若干遅くなっているが，これは種に所属する個体数が他の種数の場合と比べ 50 個体と少なく，個体が短期間で進化するために十分な個体数が得られていないと考えられる．しかし，最終的な世代では，種の数が 50 の場合に最も解の精度を高めることに成功したと言える．

本論文では，種の数を固定してからの種分化を行ったが，Li[4] や岩松 [5] の手法のように，本来は支配者からの縄張り半径に基づいて種を形成することが望ましい．今回の種分化では，単に距離の近い個体から固定数を種に属させることにしており，後半で形成される種では個体が種をまたがっている可能性が高い．これにより，距離の近い個体同士での種の形成が難しくなり，種毎による探索の集中化が困難になると考えられる．よって，解の探索をさらに発展させるためにも，種数や所属個体数の動的変化は必須である．種の個体数を動的に変動させることで，個体進化の序盤では種を多めに形成し，終盤では種を少なめに形成することで探索の集中化と分散化を図ることが可能になる．しかしながら，一般的な種分化ではなく，このように距離の近いものから種を形成する種分化でも，従来手法よりも解の精度を高められたことから，種分化は LGP における解探索に有効であることが実証できたと考える．

VI. おわりに

本論文では，GP を拡張した LGP に種分化の概念を導入することで，解空間を分割し幅広く探索することで局所解への収束を避け，高精度の解を得ることのできる SLGP を提案した．その結果，*sinepoly* 問題の関数同定では，LGP よりもより精度の高い解を得ることができた．今後の展望としては，LGP における縄張り半径の決定法，および種の数を動的に割り当てる手法の考案が挙げられる．現状では，適合度と個体の距離を基に 2 次元的な関数を描き，その峰を判定することで種を形成する手法について検討している．

参考文献

[1] John R.koza, “Genetic Programming: On the Programming of Computers Means of Natural selectio”, MIT Press, 1992.

[2] Marlus Brameier and Wolfgang Banzhaf, “Linear Genetic Programming”, Springer Science+Business Media, 2007.

[3] C.L.Alonso, J.Puente, J.L.Montaña, “Straight Line Programs:A new Linear Genetic Programming Approach”, *20th IEEE International Conference on Tool with Artificial Intelligence*, 2008.

[4] Xialdong Li, “Efficient Differential Evolution using speciation for multimodal function optimization”, *Proceeding of the 2005 conference on Genetic and evolutionary computation*, pp.873-880, 2005.

[5] 岩松 雅夫,「種分化するパーティクルスウォームによる多峰性関数の大域最適化」, **電子情報通信学会論文誌** *D*, vol.J87-D2, no.2, pp.771-774, 2004.

[6] Javier Herranz, Jordi Nin, Marc Solé, “Optimal Symbol Alignment Distance:A New Distance for Sequence of Symbols”, *IEEE Transactions on Knowledge and Data Engineering*, vol. pp, Issue:99, 2010.

問い合わせ先
〒 731-3194
広島市安佐南区大塚東 3-4-1
広島市立大学大学院 情報科学研究科 知能工学専攻
知能システム研究室
檜垣　紀志
E-mail: nhigaki@ints.info.hiroshima-cu.ac.jp