A-007

新指数型進化的プログラミングの有効性に関する研究

A Study on an Efficiency of New Exponential Evolutionary Programming

向原　康平†　　井上　浩孝‡
Kohei　Mukaihara　　Hirotaka　Inoue

1．緒言

コンピュータシステムの開発において，初期のコンピュータ科学の頃から生物学は関わっており，今日の情報科学や生物科学などの多くの分野に影響を与えている．人工知能の開発で，自然界のシステムの適応過程と生物の進化のメカニズムを模倣することによるモデルの提案がされ，それが遺伝的アルゴリズム(Genetic Algorithm：GA) の原点である．親から子に形質を伝えつつ，環境に適応していく生物の進化は遺伝と進化と呼ばれている．この進化の考え方から進化戦略(Evolutionary Strategy：ES)，進化的プログラミング(Evolutionary Programming：EP) が提唱されている．これらの生物進化及び自然選択の情報処理機能を模倣した人工知能的技法を進化的アルゴリズム(Evolutionary Algorithm：EA)，進化的計算法(Evolutionary Computation：EC) と呼ばれている [1)2)]．本研究では従来の進化的プログラミングに加えて新たに作成した 2 種類の新指数型進化的プログラミングのそれぞれの比較検討を行うとともに、計算速度の向上を行った．

2．進化的プログラミングの概要

EP は変動する環境下で Darwin の進化の概念により適当な解を繰り返し生成する進化パラダイムの 1 つである．最も一般的な枠組みにおいて，EP はアルゴリズムの繰り返しにより，パラメータを最適にする最適化手法を考えられる．他の最適化アルゴリズムと同様に，最適点は探索アルゴリズムとは完全に独立しており，適応的地勢あるいは応答曲面により決定される．標準的な形において，基本的な進化的プログラミングは初期化，変動，進化，選択を利用する．

EP の基本的なアルゴリズムは以下の通りである．

① 初期化
　試行的な解の集団を生成する．
② 変動
　各個体にランダム変数を加え，子を生成する．
③ 進化，選択
　親と子を含めた個体集団の各個体を評価し，良質な解を確率的に選択する．

今回の研究では，良質な解を確率的に選択する際にトーナメント戦略を利用している．トーナメント戦略とは、集団から決められた数の個体数を無作為に選択し，その中で最も適合値の高い個体を選択する手法である．これを次世代に残したい個体数を満たすまで繰り返す．進化的プログラミングは最適化を行う際，局所的なアプローチは一切行わず，全域的なアプローチのみを行う．

† 呉工業高等専門学校専攻科　機械電気専攻
‡ 呉工業高等専門学校　電気情報工学科

3. 進化的プログラミング

進化的アルゴリズムは最適化を行う際，局所的なアプローチは一切行わず，全域的なアプローチのみを行う．最適点は探索アルゴリズムとは完全に独立しており，適応的地勢もしくは応答曲面によって決定される．進化的プログラミングは決まった構造をもたないが，基本的な進化的プログラミングは初期化，変動，進化，選択の４つの主要な成分を利用している．個体を表現する際，n 次元実数値ベクトルと n 次元戦略パラメータベクトルの対で表現する．

3.1　標準進化的プログラミング

標準進化的プログラミング(Classical Evolutionary Programming：CEP) とは Gauss 分布に従う突然変異と自己適応，実効値ベクトルで構成される進化的プログラミングである．

3.2　高速進化的プログラミング

し高速進化的プログラミング(Fast Evolutionary Programming：FEP) とは Cauchy 分布に従う突然変異を使用した進化的プログラミングである．

3.3　指数型進化的プログラミング

指数型進化的プログラミング(Exponential Evolutionary Programming：EEP) とは複合指数分布に従う突然変異を使用した進化的プログラミングである．

3.4　新指数型進化的プログラミング

新指数型進化的プログラミング(New Evolutionary programming：NEP) とは複合指数分布を突然変異だけでなく，戦略パラメータにも適用した進化的プログラミングである．

3.5　非戦略的指数型進化的プログラミング

非戦略的指数型進化的プログラミング(non-strategy Exponential Evolutionary Programming：nsEEP) とは戦略パラメータ使用せず，探索幅を進化世代とともに指数的に減少させる手法を用いたものである．また複合指数分布のパラメータを複数回変化させる．

4. 実験方法

従来の 3 種類の EP に加え，新たに今回作成した 2 種類の EP を単峰性関数と多峰性関数の最小化問題に適用し，収束特性と最終世代での適合度を比較検討した．使用した関数はベンチマーク問題としてよく知られているものである．本実験では個体数 100，トーナメントサイズ 10

とし， 世代数は nsEEP を 2000 とし， それ以外の手法は世代数を 5000 とした．本研究の適応度は 100 回の計算結果を平均した値とし， 試行回数を 100 回 とする．各 EP によって得られた最適化までの数値の変化を元にグラフを作成し， 比較検討を行う．なお横軸は世代数であり，縦軸は適応度である．また計算にかかる時間の比較， 改良を行った．

実験に用いた PC の性能を以下に示す．

メモリ:3.6GiB

プロセッサ:Intel(R)Core(TM)i5-2400CPU@3.10GiB

OS :Fedora release16

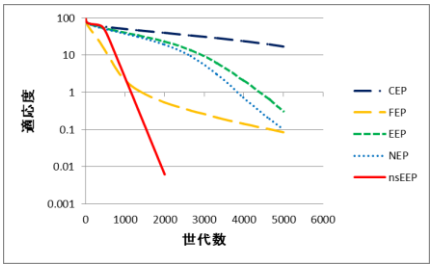

図 1．関数最小化問題の収束

5. 実験結果

今回の実験では 23 の関数最小化問題について最良解をそれぞれ求めた．その結果を表 1 に示す．最良解の個数は nsEEP が 19 個，CEP が 4 個，FEP，NEP が 3 個，EEP が 2 個という結果になった．図 1 に f=4 の各 EP の収束特性を示す．図 1 より FEP が若い世代で優れた収束特性を持つということがわかる．これは広範囲の探索を行っており， それにより適応度が高い場所をいち早く探しているということになる．これは突然変異に裾の広い Cauchy 分布を用いたことが関係していると考えられる．しかし，広範囲の探索を行っているがために， 解に近づくにつれ収束特性が悪くなってしまう．CEP は問題によっては素晴らしい収束特性を得るが，図 1 においては収束特性が良くないことがわかる．これは Gauss 分布が Cauchy 分布に比べ裾が狭く， 中心付近の値が大きいため， 局地的な探索を行っていることに起因する．CEP は局所解に陥りやすく， 収束特性が悪くなり解が算出できなくなる場合がある．EEP に用いた複合指数分布は Gauss 分布と Cauchy 分布の中間に位置するような分布であるので， 収束特性も CEP と FEP の間に位置するようになっている．また NEP により戦略パラメータが局所的な探索を行う Gauss 分布より複合指数分布の方が優れているということがわかった．nsEEP は探索範囲を戦略パラメータの分布によるものではなく強制的に狭くする手法をとり， 計算速度と収束特性の向上に成功した．これは，これまでのデータから戦略パラメータは世代が進むにつれ減少していることに着目したためである．これにより， 局所解に陥ることが少なくなり， 解の精度が向上した． さらに戦略パラメータの決定のプロセスが簡単になり， 計算量が減ったため計算速度が向上した．各 EP の計算時間を表 2 に示す．今回，nsEEP の世代数を 2000，他の EP の世代数を 5000 として実験した．世代数が少ないほうが計算回数を少なくできるので nsEEP の計算時間が短縮できたことが表 2 から分かる．また計算回数を減少させた中でも最も良い結果を出した nsEEP の有効性が確認できた．

表 1．各 EP の最良解

function	CEP	FEP	EEP	NEP	nsEEP	f min
f=1	0.000176857	0.1056	0.001093225	0.000947647	0.000204	0
f=2	0.03796	0.9258	0.09683	0.06439	0.01529	0
f=3	2.982	0.9258	3.009	2.392	0.3908	0
f=4	17.03	0.08464	0.3128	0.1023	0.006227	0
f=5	67.89	97.97	72.23	69.74	51.05	0
f=6	59.84	0.1809	48.61	12.31	0	0
f=7	0.05893	0.1229	0.05593	0.05454	0.01863	0
f=8	−7431.6	−12359.8	−7290.1	−7192.6	−10743.5	−12569.5
f=9	12.48	27.89	4.92	4.517	36.84	0
f=10	2.086	0.3543	0.2031	0.01468	0.005277	0
f=11	0.1543	0.01889	0.1516	0.03209	0.004629	0
f=12	1.059	0.000856	0.2227	0.107	0.0000009	0
f=13	0.09251	0.01922	0.01755	0.006868	0.00000003	0
f=14	3.831	1.552	4.229	3.743	0.7607	1
f=15	0.02051	0.01488	0.01983	0.0176949	0.0006405	0.0003075
f=16	−1.031628453	−1.031628452	−1.03162845	−1.03162843	−1.031628451	−1.0316285
f=17	0.3978873	0.3978873	0.3978873	0.3978873	0.3978873	0.398
f=18	3	3	3	3	3	3
f=19	−3.862	−3.862	−3.862	−3.862	−3.859	−3.86
f=20	−3.001	−3.001	−2.983	−2.981	−3.219	−3.32
f=21	−7.910	−7.502	−7.706	−7.604	−8.333	−10
f=22	−8.635	−8.542	−8.436	−8.422	−9.035	−10
f=23	−9.279	−9.022	−9.076	−9.076	−9.838	−10

表 2．各 EP の計算時間（秒）

function	CEP	FEP	EEP	NEP	nsEEP
f=1	429	357	347	270	51
f=2	436	365	355	277	55
f=3	489	416	406	329	74
f=4	437	363	355	278	53
f=5	439	366	355	278	53
f=6	428	357	347	270	51
f=7	550	482	468	390	53
f=8	499	426	415	337	81
f=9	475	412	391	314	75
f=10	481	405	386	308	73
f=11	512	410	419	342	78
f=12	510	440	429	351	87
f=13	497	431	416	338	78
f=14	468	400	389	312	78
f=15	430	361	349	272	78
f=16	423	356	344	266	78
f=17	425	357	346	270	78
f=18	425	358	345	267	78
f=19	436	369	356	279	78
f=20	437	372	358	280	78
f=21	435	363	355	277	78
f=22	437	365	357	280	78
f=23	441	368	360	282	78

6. 結言

本研究では各 EP を作成し比較検討を行い，新指数型進化的プログラミングである nsEEP の有効性を検討した．従来の EP に比べ収束特性がよくなり，計算速度も向上した．全体的に nsEEP の有効性が確認することが出来たが，問題によっては nsEEP よりも良い結果を出す EP もあった．しかし nsEEP に用いる指数分布のパラメータ設定によりそれらの EP の結果を超える可能性がある．今後は今回用いた問題以外の関数での nsEEP の試行も行い有効性を検討したい．

参考文献

1) X. Yao, Y. Liu, Fast Evolutionary Programming. Proc. of the 5th Annual Conference on Evolutionary Programming, MIT Press, pp. 451-460, 1996.

2) K. Kohmoto , H. Narihisa , K. Katayama , Evolutionary Programming Using Exponential Mutation , Proc. of the 6th World Multiconference on Systematics , Cybernetics and Informatics, vol. 11, Computer Science 2, July 14-18, USA, pp. 405-410, 2002.