岡山理科大学紀要 第41号 A pp. 69-77(2005)

新指数型進化的プログラミングの有効性について

谷口 隆裕・片山 謙吾*・南原英生*・成久 洋之*

岡山理科大学大学院工学研究科情報工学専攻

*岡山理科大学工学部情報工学科

(2005年 9 月30日受付、2005年11月 7 日受理)

1 はじめに

進化的計算（Evolutionary Computation）[1] は自然の進化と適応の概念を導入した計算システムのことであり, 従来の伝統計算手法では解けない大規模で複雑な問題を解くための強靭なしかも効率的計算法と考えられ注目されている．これには進化的戦略（Evolutionary Strategies;ES）, 進化的プログラミング (Evolutionary Programming;EP), 遺伝的アルゴリズム (Genetic Algorithms;GA) と遺伝的プログラミング (Genetic Programming;GP) などがある.

ES は 1965 年に数値最適化手法として Rechenberg と Schwefel により提案されたものである.EP は Fogel により 1965 年有限状態機械の人工知能的アプローチとして提案され, 後に組み合わせや数値最適化手法に応用されるに至った．GA は 1975 年に Holland により適応的探索アルゴリズムとして提案されたものである.GP は 1987 年に Koza の LIPS プログラムとして提案されたものであるが, 木構造染色体に対する GA の応用とみなすことができる.

これらの計算手法はすべて個体集団 (population) を使用することと個体集団を構成する個体 (individual) の間の情報交換により, 解を探索しようとするものである.

本論文は進化的計算の中の EP について記述するものである. 進化的プログラミングは有限個の解集合からなる個体集団に対し突然変異 (mutation) と選択により, より質の高い解をもつ個体集団を発生させようとする集団探索法である．これまでに提案されている EP は主として Gaussian Mutation による CEP と Cauchy Mutation による FEP 手法がある．前者は Gauss 分布乱数を, 後者は Cauchy 分布乱数をそれぞれ使用し, 関数最適化問題に対しては一般的に FEP の方が有効であるとされている [2]. しかしながら, 問題によっては CEP の方がよりすぐれた収束特性を示すこともあり, さらに収束の状況によって結果は異なる. この観点から,Narihisa らは 2002 年に複合指数分布乱数を使用する指数型進化的プログラミング (EEP) を提案した [3]. これは指数分布のパラメータを制御することで分布の分散を変動させ収束の効率化を狙ったものである. これまでの研究では種々のパラメータ値に対する収束特性を検討してきたが [4][5][6][7], 今回はパラメータ値を進化世代 (evolutionary generation) に対応して変動させる新 EP 手法の有効性について検討するものである [8][9]. さらに, 従来の EP 手法も含めて, 対象問題毎にそれぞれ best-tuning なパラメータ値の存在性を併せて検討することで従来の EP で必要不可欠とされている戦略パラメータをなしとした nsEEP をも新 EEP 手法として提案し, その有効性を示すものである.

数値実験として, この分野でよく知られたベンチマーク問題に適応した結果, かなり良好な解が得られている.

2 標準進化的プログラミング（CEP）

Bäck と Schwefel による EP のアルゴリズムは次の通りである.

Step 1: μ 個の個体からなる初期個体集団の生成、$k = 1$ とする. 各個体は実数ベクトルの対 $(\boldsymbol{x_i}, \boldsymbol{\sigma_i})$,$\forall_i \in \{1, 2, \ldots, \mu\}$, ただし,$\boldsymbol{x_i}$ は変数ベクトル, $\boldsymbol{\sigma_i}$ はガウス分布の標準偏差ベクトルとする.

Step 2: 個体集団の各個体 $(\boldsymbol{x_i}, \boldsymbol{\sigma_i})$,$\forall_i \in \{1, 2, \ldots, \mu\}$ に対して適合度を計算する.

Step 3: 各個体 $(\boldsymbol{x_i}, \boldsymbol{\sigma_i})$,$i = 1, \ldots, \mu$ は単一の子孫 $(\boldsymbol{x_i'}, \boldsymbol{\sigma_i'})$ を生成する. $j = 1, 2, \ldots, n$ に対して

$$\sigma_i'(j) = \sigma_i(j) exp\{\acute{\tau} N(0,1) + \tau N_j(0,1)\} \tag{2.1}$$

$$x_i'(j) = x_i(j) + \sigma_i(j) N_j(0,1) \tag{2.2}$$

ただし,$x_i(i), x_i'(j), \sigma_i(j), \sigma_i'(j)$ はベクトル $\boldsymbol{x_i}, \boldsymbol{x_i'}, \boldsymbol{\sigma_i}, \boldsymbol{\sigma_i'}$ の j-成分を表わす.$N(0,1)$ は標準化された 1 次元正規乱数で, その平均は 0, 標準偏差は 1 となっていることを示す. $N_j(0,1)$ は各 j 毎に発生する正規乱数を示し, $\acute{\tau} = (\sqrt{2\sqrt{n}})^{-1}, \tau = (\sqrt{2n})^{-1}$ とする.

Step 4: 各子孫 $(\boldsymbol{x_i'}, \boldsymbol{\sigma_i'})$,$\forall_i \in \{1, 2, \ldots, \mu\}$ の適応度を計算する.

Step 5: q トーナメント選択を実施し,$(\boldsymbol{x_i}, \boldsymbol{\sigma_i})$,$(\boldsymbol{x_i'}, \boldsymbol{\sigma_i'})$ の中から μ 個選択し, 次世代の親とする.

Step6: 停止条件を満たせば停止, そうでなければ $k = k + 1$ として Step3 へ.

3 Cauchy 乱数を用いた進化的プログラミング (FEP)

CEP では Gauss 分布に従う正規乱数 N(0,1) を使用した突然変異を考えたのに対して,FEP では Cauchy 分布に従う乱数を使用する. 原点を平均とする 1 次元 Cauchy 分布の密度関数は

$$f(x,t) = \frac{1}{\pi} \cdot \frac{t}{x^2 + t^2}, -\infty < x < +\infty, t > 0 \tag{3.1}$$

であり,t はスケールパラメータである. これに対応する Cauchy 分布関数 $F(x,t)$ は次のようになる.

$$F(x,t) = \frac{1}{2} + \frac{1}{\pi} \cdot arctan\left(\frac{x}{t}\right) \tag{3.2}$$

したがって,[0,1] 区間の一様乱数を y とすると,

$$x = t \cdot tan\left\{\pi\left(y - \frac{1}{2}\right)\right\} \tag{3.3}$$

となり, この乱数を $C(0,t)$ で表わす. スケールパラメータ $t = 1$ とした乱数が $C(0,1)$ であり, 次のようになる.

$$C(0,1) = tan\left\{\pi\left(a - \frac{1}{2}\right)\right\}, 0 \leqq a \leqq 1 \tag{3.4}$$

FEP の処理アルゴリズムは CEP の Step 3 における式 (2.2) の代りに

$$x_i'(j) = x_i(j) + \sigma_i(j) C_j(0,1) \tag{3.5}$$

としたものである.

4 複合指数乱数を用いた進化的プログラミング (EEP)

EEP では複合指数分布に従う乱数 $E(0,\lambda)$ を用いた突然変異を考慮するものである. パラメータ λ の複合指数分布の 1 次元の確率密度関数 $f(x)$ は

$$f(x) = \frac{\lambda}{2} \exp\{-\lambda|x|\}, \ -\infty < x < +\infty, \ \lambda > 0 \tag{4.1}$$

として与えられる. したがって, この分布において平均 $\bar{x} = 0$, 分散 $var(x) = \frac{2}{\lambda^2}$ となる. このことから, この分布の分散は λ が小さければ大きく,λ が大きければ小さくなり,λ により制御可能であることを示している. $f(x)$ に対応した分布関数 $F(x)$ は

$$F(x) = \begin{cases} \frac{1}{2}\exp[\lambda x] & x \leqq 0 \\ 1 - \frac{1}{2}\exp[-\lambda x] & x > 0 \end{cases} \tag{4.2}$$

として与えられる．したがって,[0,1] 区間での一様乱数を y とすると,

$$x = \begin{cases} \frac{1}{\lambda}\ln(2y) & y \leqq \frac{1}{2} \\ -\frac{1}{\lambda}\ln\{2(1-y)\} & y > \frac{1}{2} \end{cases} \tag{4.3}$$

で発生させる乱数 x は $E(0,\lambda)$ で表わされる．このことから,$E(0,\lambda) = \frac{1}{\lambda}E(0,1)$ となり $E(0,1)$ を発生させることにより $E(0,\lambda)$ を計算できる．EEP では CEP の Step3 における式 (2.2) の代りに

$$x_i'(j) = x_i(j) + \sigma_i(j)E_j(0,\lambda) \tag{4.4}$$

としたものである．

この λ の値として, 次の 2 タイプを考える．

$$\lambda_L = \lambda_1 + (\frac{\lambda_2 - \lambda_1}{G})g, \tag{4.5}$$

$$\lambda_E = \lambda_1 exp[(ln(\frac{\lambda_2}{\lambda_1}))\frac{g}{G}], \tag{4.6}$$

$\lambda = \lambda_L$ とした EEP を *lin*EEP、$\lambda = \lambda_E$ とした EEP を *exp*EEP と呼ぶ．ただし,g は進化世代,λ_1, λ_2 は λ の初期値, 最終値,G は全進化世代数とする．さらに,*exp*EEP において $\sigma_i'(j)$ を使用しないものを *ns*EEP と呼ぶ．すなわち, 式 (2.1),(2.2) の代りに

$$x_i'(j) = x_i(j) + E_j(0,\lambda) \tag{4.7}$$

$$E_j(0,\lambda) = \frac{w_j}{\lambda_1}E_j(0,1)\exp[-ln(\frac{\lambda_2}{\lambda_1})\frac{g}{G}], \tag{4.8}$$

としたものである．ただし,w_j は与えられた問題の j 成分の各変数における許容範囲を示すものとする．したがって,*ns*EEP のアルゴリズムは次のようになる．

Step 1: μ 個の個体からなる初期個体集団の生成,$k = 1$ とする. 各個体は実数ベクトル $\boldsymbol{x_i}, \forall_i \in \{1, 2, \ldots, \mu\}$, ただし,$\boldsymbol{x_i}$ は変数ベクトルとする.

Step 2: 個体集団の各個体 $(\boldsymbol{x_i}), \forall_i \in \{1, 2, \ldots, \mu\}$ に対して適合度を計算する.

Step 3: 各個体 $(\boldsymbol{x_i}), i = 1, \ldots, \mu$ は単一の子孫 $\boldsymbol{x_i'}$ を生成する. $j = 1, 2, \ldots, n$ に対して

$$x_i'(j) = x_i(j) + E_j(0,\lambda)$$

ただし,$x_i(i), x_i'(j)$ はベクトル $\boldsymbol{x_i}, \boldsymbol{x_i'}$ の j-成分を表わす. $E_j(0,\lambda)$ は各 j 毎に発生する複合指数乱数を示す.

Step 4: 各子孫 $\boldsymbol{x_i'}, \forall_i \in \{1, 2, \ldots, \mu\}$ の適応度を計算する.

Step 5: q トーナメント選択を実施し,$\boldsymbol{x_i}, \boldsymbol{x_i'}$ の中から μ 個選択し, 次世代の親とする.

Step 6: 停止条件を満たせば停止, そうでなければ $k = k + 1$ として Step3 へ.

5 数値実験

5.1 今回の数値実験の目的

EP の計算においては戦略パラメータ σ の下限 ϵ の値,EEP においては λ_1, λ_2 の値によりその収束特性は異なる．本実験ではこれらのパラメータ値をある範囲で変動させて, 与えられた問題に適合したパラメータ値を使用することでどの程度に解精度を向上し得るか検討するものである．したがって, 各問題に対し $(\epsilon, \lambda_1, \lambda_2)$ の 3 つのパラメータの組合せで解かせるために並列コンピュータ Paragon XP/S15(296 ユニット) を使用した．

5.2 適用問題

EEP の収束特性を検討するために, 今回の実験で対象とした問題はこの分野でよく知られているベンチマーク問題で, 表 1 に与えている関数最適化問題である. この中の f_1〜f_6 は単一の局所解をもつ *uni-modal* 関数であり,f_7〜f_{12} は複数の局所解をもつ *multi-modal* 関数である. また, 各ベンチマーク問題における S は x_i の範囲,f_{min} は厳密解である.

5.3 パラメータ設定

数値実験におけるパラメータは次のように設定した.

個体集団の大きさ ; $\mu = 100$

トーナメントサイズ ; $q = 10$

計算世代数 ; G=5000

ϵ の値 ; $\{\epsilon_1 = 10^{-2}, \epsilon_2 = 10^{-3}, \epsilon_3 = 10^{-4}, \epsilon_4 = 10^{-5}, \epsilon_5 = 10^{-6}, \epsilon_6 = 10^{-8}\}$

*lin*EEP と *exp*EEP において
初期値 ; $\lambda_1 = \{\xi_1 = 1, \xi_2 = 0.1, \xi_3 = 0.05, \xi_4 = 0.01, \xi_5 = 10^{-3}, \xi_6 = 10^{-4}\}$
終期値 ; $\lambda_2 = \{\eta_1 = 10, \eta_2 = 20, \eta_3 = 50, \eta_4 = 100\}$

*ns*EEP において
初期値 ; $\lambda_1 = \{\xi_1 = 5 \times 10^{-5}, \xi_2 = 5 \times 10^{-4}, \xi_3 = 5 \times 10^{-3}, \xi_4 = 5 \times 10^{-2}, \xi_5 = 5 \times 10^{-1}, \xi_6 = 5.0,$
$\xi_7 = 5 \times 10^{1}, \xi_8 = 5 \times 10^{2}\}$
終期値 ; $\lambda_2 = \{\eta_1 = 5, \eta_2 = 5 \times 10^{1}, \eta_3 = 5 \times 10^{2}, \eta_4 = 5 \times 10^{3}, \eta_5 = 5 \times 10^{4}, \eta_6 = 5 \times 10^{10}, \eta_7 = 5 \times 10^{16},$
$\eta_8 = 5 \times 10^{22}\}$
戦略パラメータの初期値 ; $\sigma_i(j)$; [0,1] 区間の一様乱数

5.4 実験実施要領

本実験は複合指数分布のパラメータを変動させる EEP の収束特性と戦略パラメータ無しの新 EEP の収束特性につき検討するものであるが、同様に,CEP および FEP についても 5000 世代までの特性につき記述する. これは EEP との性能比較という目的もさることながら,CEP や FEP についての性能記述が多くの関連文献において 1500 世代か 2000 世代まで位のものであり、5000 世代までのものは殆んどないからである. 例えば,FEP の方が CEP よりは一般によい特性を示すものとされているが, このことは 1500 世代あるいは 2000 世代までの特性であり, それ以降のものについては殆んど言及されていない. しかし本実験では 2000 世代以降においては CEP の方が良好な結果を齎すことも明確になっている. これらの結果は基本的に Gauss 分布と Cauchy 分布の特徴から類推しうるものである.

本実験での収束特性は各世代毎に各個体の関数値を 100runs の平均値として評価値とした. ここでは関数最適化問題を取り扱うので関数値が小さい程望ましい状態といえる. 進化の過程で, 戦略パラメータ $\sigma(j)$ は新しい解を決定する場合の変異巾とに関与するものであるが,$|\sigma(j)| = 0$ は進化の停止を意味する. この状況を避けるために,$|\sigma(j)|$ の下限を ϵ とし,$|\sigma(j)| < \epsilon$ であれば強制的に $|\sigma(j)| = \epsilon$ として進化を続行させる. 今回の実験では各問題に適した下限値についても検討し, 最終世代での適応度を比較検討する.

6 実験結果とその考察

実験結果は表 2, 表 3, 表 4 の通りである. 表 2 は *lin*EEP の最終解, 表 3 は *exp*EEP の最終解, 表 4 は *ns*EEP の最終解を示すものである. 表 2,3,4 は $(\epsilon, \lambda_1, \lambda_2)$ の best tuning における解を示している. これからわかるように CEP や FEP も問題により ϵ の値は異なり,*lin*EEP や *exp*EEP では ϵ は勿論,λ_1, λ_2 の値はそれぞ

れ異なったものとなっている．それぞれが最適な値を採用したときの解を示している．*lin*EEP では 12 問中 8 問において CEP や FEP よりも良質な解が求められており，その有効性を示している．また，*exp*EEP では 12 問中 10 問において CEP や FEP より良質な解が求められており，これも EEP の有効性を示している．また同時に，解の質を良くするような問題固有の best-tuning なパラメータ値が存在していることがわかる．特に表 4 は戦略パラメータなしの *ns*EEP の結果を示したものであるが，適切な λ_1, λ_2 を選択できれば CEP や FEP と比較して桁違いに良質な解が求められている．さらに 12 問において良質な解が求められるようなパラメータ値が存在していることがわかった．また，これまでの実験で解が効率良く進化しているときに λ を大きく変えて解探索能力が落ちたこと，ユニモーダル関数のように進化が落ち着く最適値周辺で λ を大きく変えることによって解探索能力が上がった事ことから，解が効率良く進化するであろう進化初期から中期あたりでは λ の値の変化の小さい方が効果的な解探索が可能になるという期待ができる．以上のことから，*exp*EEP の場合は ϵ が 6 個，λ_1 が 6 個，λ_2 が 4 個の 144 通りの組合せとし，*ns*EEP では 8×8 の 64 通りの組合せ結果を示すものであることから *ns*EEP の方が計算資源は少なくして良質な結果を求められていることを示す．一方，CEP や FEP については 6 通りの組合せの計算結果であるが，Intel(R)Pentium(R)4 CPU 3.2GHz 1.96GB RAM のコンピュータ 1 台でこの種の問題を解くと仮定すると，平均的に 1 回の処理に 1200 秒，*ns*EEP の場合戦略パラメータ σ の処理をしない分 1 つの問題に対する処理時間 $\frac{1}{4}$ の 300 秒程度で済むことから，表 2 の CEP，FEP は $6 \times 1200 = 7200$ 秒に対し，*ns*EEP は $64 \times 300 = 19200$ 秒となって，約 2.6 倍の処理時間が必要であることを示している．メタヒューリスティックの観点から見ると，多少余分な処理時間がかかってもより高精度の解が求められるならば，その方が望ましいこともあり得るわけで，*ns*EEP の有効さが期待し得るものと思われる．ちなみに，CEP や FEP などの従来の EP では 5000 世代でこのような質の解は絶対に求められないし，その何十倍の時間をかけても不可能と思われる．また表 2~4 のパラメータ値より，今回のように λ を各問題によって最適なパラメータ値を与えなくても，λ を設定をある程度の範囲内で固定すれば良質な解を得ることができると思われる．

参考文献

[1] T.Back and H.-P.Schwefel,"An overview of evolutionary algorithms for paramenter optimization", Evolutionary Computation,1(1):1-23,1993.

[2] X. Yao and Y. Liu, "Fast Evolutionary Programming," Proc. of the 5th Annual Conference on Evolutionary Programming, MIT Press, pp.451–460, 1996.

[3] K.Kohmoto,H.Narihisa and K.Katayama,"Evolutionary programming using eponential mutation", Proc.of The 6th World Multiconference on Systemics, Cybernetics and Informatics(SCI2002), vol.XI,Computer Science ,pp.405-410,2002.

[4] H.Narihisa,K.Kohmoto,and K.Katayama,"Evolutionary Programming with Double Exponential Probability Distribution", Proc.of the Second IASTED International Conference on Artificial Intelligence And Applications, September9-12,Spain,358-363,2002.

[5] K.Kohmoto,H.Narihisa,and K.Katayama,"Performance of Evolutionary Programming Using Exponential Mutation", Proc.of the 4th Asia-Pasific Conference on Simulated Evolution And Learnig, (SEAL'02)vol.2,pp454-458,2002.

[6] H.Narihisa,K.Kohmoto,T.Kumon, and K.Katayama,"Performance of Exponential Evolutionary Programming", Proc.of IASTED International Conference on Artificial Intelligence and Soft Computing(ASC2003), pp.243-248,2003.

[7] H.Narihisa,K.Kohmoto,M.Tsuda, and K.Katayama,"CONVERGENCE CHARACTERISTICS OF EXPONENTIAL EVOLUTIONARY PROGRAMMING", Proc.of the 8th IASTED International Conference on Artificial Intelligence and Soft Computing(ASC2004), pp.426-431,2004.

[8] H.Narihisa, T.Taniguchi, M.Thuda and K.Katayama, "Efficiency of Parallel Exponential Evolutionary Programming", Proc. of the 2005 International Conference on Parallel Processing Workshops, pp.588-595, June.13-17, 2005, Oslo, Norway.

[9] H.Narihisa, T.Taniguchi, M.Ohta and K.Katayama, "EVOLUTUIONARY PROGRAMMING WITH EXPONENTIAL MUTATION", Proc. of the Ninth IASTED International Conference ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING, pp.55-60, 2005.

表 1: TEST FUNCTION

Test function	S	f_{min}
$f_1(x) = \Sigma_{i=1}^{30} x_i^2$	$[-100, 100]^{30}$	0
$f_2(x) = \Sigma_{i=1}^{30} \lvert x_i \rvert + \Pi_{i=1}^{30} \lvert x_i \rvert$	$[-10, 10]^{30}$	0
$f_3(x) = \Sigma_{i=1}^{30} (\Sigma_{j=1}^{i} x_j)^2$	$[-100, 100]^{30}$	0
$f_4(x) = \underset{i}{max} \lvert x_i \rvert, 1 \leq i \leq 30$	$[-100, 100]$	0
$f_5(x) = \Sigma_{i=1}^{29} [100(x_{x+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]^{30}$	0
$f_6(x) = \Sigma_{i=1}^{30} (\lfloor x_i + 0.5 \rfloor)^2$	$[-100, 100]^{30}$	0
$f_7(x) = \Sigma_{i=1}^{30} i x_i^4 + random[0, 1)$	$[-1.28, 1.28]^{30}$	0
$f_8(x) = -\Sigma_{i=1}^{30} \left(x_i \sin \left(\sqrt{\lvert x_i \rvert} \right) \right)$	$[-500, 500]^{30}$	-12569.5
$f_9(x) = \Sigma_{i=1}^{30} [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^{30}$	0
$f_{10}(x) = -20exp(-0.2\sqrt{\frac{1}{30}\Sigma_{i=1}^{30} x_i^2}) - exp(\frac{1}{30}\Sigma_{i=1}^{30} \cos 2\pi x_i^2) + 20 + e$	$[-32, 32]^{30}$	0
$f_{11}(x) = \frac{1}{4000}\Sigma_{i=1}^{30} x_i^2 - \Pi_{i=1}^{30} \cos \left(\frac{x_i}{\sqrt{i}} \right) + 1$	$[-600, 600]^{30}$	0
$f_{12}(x) = \frac{\pi}{n}\{10 \sin^2(\pi y_i) + \Sigma_{i=1}^{n-1}(y_i - 1)^2[1 + 10 \sin^2(\pi y_{i+1} + (y_n - 1)^2\}$ $+\Sigma_{i=1}^{n} u(x_i, 10, 100, 4),\ y_i = 1 + \frac{1}{4}(x_i + 1)$	$[-100, 100]^{30}$	0

表 2: Final Solution of *lin*EEP

f	CEP		FEP		*lin*EEP			
	ϵ	f	ϵ	f	ϵ	λ_1	λ_2	f
f_1	10^{-6}	2.293×10^{-7}	10^{-8}	6.359×10^{-4}	10^{-8}	10^{-3}	10	$\mathbf{2.7 \times 10^{-14}}$
f_2	10^{-8}	1.195×10^{-3}	10^{-8}	5.406×10^{-3}	10^{-8}	0.05	10	$\mathbf{9.531 \times 10^{-8}}$
f_3	10^{-2}	6.165	10^{-8}	31.09	10^{-2}	10^{-2}	10	$\mathbf{2.081 \times 10^{-2}}$
f_4	10^{-2}	3.832×10^{-2}	10^{-8}	6.167×10^{-2}	10^{-2}	10^{-2}	10	$\mathbf{3.882 \times 10^{-3}}$
f_5	10^{-4}	101.8	10^{-8}	83.56	10^{-6}	10^{-1}	10	**44.04**
f_6	10^{-2}	1.424	10^{-6}	0.7002	10^{-4}	0.05	10	**0**
f_7	10^{-4}	6.587×10^{-2}	10^{-8}	6.072×10^{-2}	10^{-2}	0.05	10	$\mathbf{1.923 \times^{-2}}$
f_8	10^{-4}	-7637.8	10^{-4}	$\mathbf{-12560.6}$	10^{-8}	0.05	10	-10900
f_9	10^{-4}	12.509	10^{-8}	**0.208**	10^{-8}	0.05	10	17.97
f_{10}	10^{-4}	2.039	10^{-8}	4.590×10^{-3}	10^{-4}	10^{-1}	10	$\mathbf{1.154 \times 10^{-4}}$
f_{11}	10^{-2}	4.105×10^{-2}	10^{-6}	$\mathbf{1.013 \times 10^{-2}}$	10^{-2}	10^{-4}	10	1.125×10^{-2}
f_{12}	10^{-8}	0.1630	10^{-6}	$\mathbf{5.058 \times 10^{-4}}$	10^{-8}	0.05	10	6.532×10^{-2}

表 3: Final Solution of *exp*EEP

f	CEP		FEP		*exp*EEP			
	ϵ	f	ϵ	f	ϵ	λ_1	λ_2	f
f_1	10^{-6}	2.293×10^{-7}	10^{-8}	6.359×10^{-4}	10^{-6}	0.05	100	$\mathbf{9.2 \times 10^{-14}}$
f_2	10^{-8}	1.195×10^{-3}	10^{-8}	5.406×10^{-3}	10^{-8}	1.0	100	$\mathbf{1.195 \times 10^{-8}}$
f_3	10^{-2}	6.165	10^{-8}	31.09	10^{-2}	1.0	100	$\mathbf{1.190 \times 10^{-2}}$
f_4	10^{-2}	3.832×10^{-2}	10^{-8}	6.167×10^{-2}	10^{-2}	0.05	100	$\mathbf{7.737 \times 10^{-4}}$
f_5	10^{-4}	101.8	10^{-8}	83.56	10^{-2}	1.0	100	**75.28**
f_6	10^{-2}	1.424	10^{-6}	0.7002	10^{-2}	1.0	100	**0**
f_7	10^{-4}	6.587×10^{-2}	10^{-8}	6.072×10^{-2}	10^{-2}	1.0	100	$\mathbf{1.894 \times 10^{-2}}$
f_8	10^{-4}	-7637.8	10^{-4}	-12560.6	10^{-4}	10^{-4}	20	**-12565.9**
f_9	10^{-4}	12.509	10^{-8}	**0.2082**	10^{-8}	0.05	10	5.522
f_{10}	10^{-4}	2.039	10^{-8}	4.590×10^{-3}	10^{-6}	0.05	100	$\mathbf{3.771 \times 10^{-7}}$
f_{11}	10^{-2}	4.105×10^{-2}	10^{-6}	$\mathbf{1.013 \times 10^{-2}}$	10^{-2}	1.0	100	1.284×10^{-2}
f_{12}	10^{-8}	0.1630	10^{-6}	5.058×10^{-4}	10^{-2}	10^{-4}	20	$\mathbf{1.081 \times 10^{-5}}$

表 4: Final Solution of *ns*EEP

f	CEP		FEP		*ns*EEP		
	ϵ	f	ϵ	f	λ_1	λ_2	f
f_1	10^{-6}	2.293×10^{-7}	10^{-8}	6.359×10^{-4}	5.0	5×10^{22}	$\mathbf{7.9 \times 10^{-44}}$
f_2	10^{-8}	1.195×10^{-3}	10^{-8}	5.406×10^{-3}	5.0	5×10^{22}	$\mathbf{1.223 \times 10^{-21}}$
f_3	10^{-2}	6.165	10^{-8}	31.09	5×10^{1}	5×10^{4}	$\mathbf{1.280 \times 10^{-7}}$
f_4	10^{-2}	3.832×10^{-2}	10^{-8}	6.167×10^{-2}	5.0	5×10^{22}	$\mathbf{1.305 \times 10^{-22}}$
f_5	10^{-4}	101.8	10^{-8}	83.56	5.0	5×10^{2}	**46.86**
f_6	10^{-2}	1.424	10^{-6}	0.7002	5×10^{1}	5×10^{10}	$\mathbf{0}_{(657generation)}$
f_7	10^{-4}	6.587×10^{-2}	10^{-8}	6.072×10^{-2}	5.0	5×10^{2}	$\mathbf{5.109 \times 10^{-3}}$
f_8	10^{-4}	-7637.8	10^{-4}	-12560.6	5×10^{-3}	5×10^{10}	**-12569.4**
f_9	10^{-4}	12.509	10^{-8}	0.2082	5×10^{-4}	5×10^{4}	$\mathbf{7.827 \times 10^{-6}}$
f_{10}	10^{-4}	2.039	10^{-8}	4.590×10^{-3}	5×10^{-1}	5×10^{16}	$\mathbf{1.715 \times 10^{-7}}$
f_{11}	10^{-2}	4.105×10^{-2}	10^{-6}	1.013×10^{-2}	5×10^{-1}	5×10^{4}	$\mathbf{3.203 \times 10^{-3}}$
f_{12}	10^{-8}	0.1630	10^{-6}	5.058×10^{-4}	5.0	5×10^{22}	$\mathbf{0}_{(3634generation)}$

Efficiency of new Exponential Evolutionary Programming

Takahiro Taniguchi, Kengo Katayama*, Hideo Minamihara* and Hiroyuki Narihisa*

Graduate School of Engineering,
**Department of Information and Computer Engineering,*
Faculty of Engineering.
Okayama University of Science
1-1 Ridai-cho, Okayama, 700-0005, Japan

(Received September 30, 2005; accepted November 7, 2005)

The term evolutionary computing refers to the study of the foundation and applications of certain heuristic technigues and based on the principles of natural evolution;thus the aim of designing evolutionary algorithms is to mimie some of the processes taking place in natural evolution. These algorithms are classified into three main categories,depending more on historical development than on major functional techniques. In fact,their biological basis is essentially the same.

These algorithms contain evolutionary programming(;EP),evolutionary stratefies(;ES),genetic algorithms(;GA) and genetic programming(;GP).

An evolutionary algorithm(;EA) is an iterative and stochsstic process that operates on a set of individuals(called population). Each individual represents a potential solution to the problem being solved.Initially, the population is randomly generated. Every indovidual in the population is assigned, by means of a fitness function, a mesures of its goodness with respect to the problem under consideration, which guides the search.

Evolutionary Programming is one of main branch of of evolutionary computation which mimic biological evolution and natural selections. The main feature of Evolutionary programming is that evolution process uses only mutation operator. This property can be considered to be easily applicable to miscellaneous type optimization problems. Originally, this evolutionary programming is proposed by D.B.Fogel. Later, Yao et al modified this as fast evolutionary programming(FEP) by using Cauchy mutation instead of Gaussian mutation.

In 2002, Narihisa proposed exponential evolutionary programming (EEP) by using double exponential mutation. This EEP aims efficient evolving process by controlling step length of optimization. In these years, the efficiency of EEP has reported for function optimization problems. Almost existing evolutionary programming use strategy parameter for sake of self-adaptiviness.

In this paper, we propose new evolutionary programmings by using double exponential distribution parameter λ varying with evolutionary generation. The experimental results show excellent high qualty solution on applying to function optimization problems.