자가 적응 세대차를 이용한 진화 알고리즘

Evolutionary Algorithm using Self-Adaptation Generation Gap

최준석, 서기성

서울시 성북구 서경대학교 전자공학과
E-mail: cjceous@hanmail.net, ksseo@skuniv.ac.kr

요 약

본 논문은 최적 탐색 알고리즘중의 하나인 실수 표현 진화 알고리즘에 자가 적응 세대차 조절을 이용하여 보다 빠른 연산으로 우수해에 접근하기 위한 새로운 방식을 소개한다. 알고리즘의 성능에 영향을 끼치는 진화 속도를 기존 진화 방식과 유전연산자의 수정을 통해 조절하여 탐색 성능을 개선 한다. 조기 수렴의 방지 및 탐색성능의 향상을 위하여 선택과 대치를 포함한 진화방식을 개선하고, 유전 연산자에 의하여 생성된 자손의 대치확률에 따라서 자손의 생성범위를 자가 적응적으로 조절하여, 보다 적은 계산량으로 전역 최적화를 찾고자 한다. 제안된 방법을 벤치마크 테스트 문제에 적용하여 G3 알고리즘, CMA-ES 그리고 DE 등과 성능을 비교하였다.

Key Words : Evolutionary Algorithm, Real Number Optimization, Self-adaptation Generation Gap, PCX(Parent-centric crossover)

1. 서 론

실수 최적화 문제에 대해서 기존의 비트 스트링으로 이루어진 유전자 알고리즘에 비해서 실수 코딩 GA 나 진화 알고리즘이 더 우수한 성능을 보이고 있다[1,3].

관련 연구로는 공분산 행렬을 이용하여 ES(Evolutionary Strategy)를 개선한 CMA-ES[1]등과, 실수 연산자를 개선한 UNDX[2], PCX[3] 그리고 SPX[4]등이 있다. 그중에서 PCX 연산자를 사용한 지역탐색과 수정된 점진적인(steady state) 진화방식인 G3 방식을 결합한 알고리즘[3]이 가장 우수한 성능을 나타내고 있다.

특히, 위와 같이 실수 최적화 문제에 우수한 성능을 보이고 있는 관련 알고리즘들은 세대형(generational) 진화 방식보다는 점진적인(steady state) 진화방식을 택하고 있는 특징이 있다.

그리고 교차, 돌연변이, 재조합등의 유전 연산자가 탐색 성능에 큰 영향을 미치고 있다[2-4].

점진적 진화 방식내에서도 선택(selection)과 대체(replacement) 방법에 따라 다시 성능의 차이를 보일수 있다. 더욱이 진화 방식에 유전 연산자가 결합되면 관계는 매우 복잡해진다.

진화 방식과 유전연산자의 여러 조합에 대한 상당한 연구가 진행되었지만[3], 아직까지 각각의 주요 특성이나 이들의 결합관계에 대해서는 뚜렷이 규명된 바가 없다.

사전 분석을 통해, 기존의 가장 우수한 G3+PCX 알고리즘[3]이 지역탐색 방식이기 때문에 생성된 자손들이 지역최적화로 집중될 수 있으며, 이에 따라 진화속도가 빠른 특성을 보이는 점을 발견하였다.

따라서, 본 연구에서는 알고리즘의 성능에 영향을 끼치는 진화 속도를 기존 진화 방식과 유전연산자의 수정을 통해 자가 적응적으로 조절하여 탐색 성능을 개선하고자 한다.

제안된 기법을 실수 벤치마크 문제인 Rosenbrock, Schwefel 그리고 Ellipsoidal 함수에 대하여 실험하고 기존의 우수한 알고리즘과 비교한다.

2. PCX 연산자 기반 SGG(Self-adaptation Generation Gap)알고리즘

2.1 PCX 연산자 분석

현재까지 실수 최적화 문제에 가장 우수한 방법 중의 하나라고 알려진 진화알고리즘은 G3(Generalized Generation Gap)와 PCX 연산자를 조합하고 있다[3]. PCX 의 연산자의 특성은 우수한 염색체와 μ개의 임의의 부모 염색체를 재조합하여 우수한 부모를 중심으로 자손을 발생시키는 교배 연산자로서 새로운 자손은 식 (1) 에 의해서 구해진다.

$$\vec{y}=\overrightarrow{x^{(p)}}+\omega_{\zeta}\overrightarrow{d^{(p)}}+\sum_{i=1, i\neq p}^{u}\omega_{\eta}\overrightarrow{De^{(i)}} \qquad (1)$$

표 1. PCX 연산자

1) P세대에서 우수한 부모 $\overrightarrow{x^{(p)}}$를 선택한다.
2) 하나의 우수한 부모와 임의의 μ개의 중심 $\vec{g}$를 계산한다.
3) 우수한 부모와 중심과의 거리 $\overrightarrow{d^{(p)}}=\overrightarrow{x^{(p)}}-\vec{g}$를 구한다.
4) μ개의 임의의 부모와 $\overrightarrow{d^{(p)}}$와의 직교거리 평균 D를 계산한다.
5) $\overrightarrow{d^{(p)}}$와 μ개의 부모와 단위직교 벡터 $\overrightarrow{e^{(i)}}$를 계산한다.

PCX 연산자는 전체의 군집에서 우수한 부모를 중심으로 임의의 부모와의 거리, 벡터의 성분 그리고 확률변수를 이용하여 자손을 생성하며, 표 1에 자세한 설명이 나와 있다.

G3+PCX 연산자 알고리즘[3]을 분석한 결과, 자손이 부모와 대치될 확률이 평균적으로 92%의 수치가 나온다. PCX 연산자가 부모 근방의 지역탐색의 특성을 가지고 있고, G3 방식이 우수한 자손개체로 랜덤한 부모를 대체하는 특성이 있으므로, 진화속도를 빠르게 하는 경향이 있다. 이 경우 군집의 우수한 개체가 지역최적화를 벗어나지 못하면 빠른 속도로 조기 수렴될 수 있으며, 지역최적화로 개체들이 집중되면 지역최적화를 벗어나기 위하여 더 많은 계산량을 필요로 하게 된다.

이러한 현상을 방지하기 위하여 진화속도를 적절히 조절하여, 군집의 개체들의 다양성을 함께 유지할수 있는 기법이 필요하다.

본 논문에서는 진화속도의 조절 지표로서 유전연산후 자손이 부모를 대치하는 비율을 사용하며, 이를 적정한 수준에서 유지하기 위한 진화 속도의 자가 적응형 진화 알고리즘을 구현하였다.

2.2 제안된 SGG 알고리즘

적절한 진화속도(자손의 부모 대치 비율)를 자가적응적으로 조절하여 탐색 성능을 향상시킬수 있는 SGG(Self-adaptation Generation Gap) 이 표 2 에 설명되어 있다.

표 2. SGG 알고리즘

1) 군집 P에서 가장 우수한 부모 한 개와 μ개의 임의의 부모를 선택한다.
2) μ+1개의 부모에 PCX 연산자를 이용하여 λ/2개의 자손을 생성한다.
3) 1)과 2)를 반복하여 나머지 λ/2의 자손을 생성한다.
4) 군집 P에서 자손과 비교할 부모를 임의로 λ/2개 선택한다.
5) 군집 P에서 자손과 비교할 부모를 토너먼트(k = 12) 선택을 통하여 λ/2개 선택한다.
6) 4),5)를 통하여 선택된 부모와 λ개의 자손을 비교하여 높은 만족도를 가지는 염색체를 대치한다.
7) 일정한 세대에 자손의 부모 대치확률에 따라서 ω_{ζ}를 조절한다.

이 알고리즘은 진화속도를 조절하기 위하여 현 세대까지 누적된 자손의 대치확률을 이용하여 PCX 연산자의 ω_{ζ}를 적응적으로 조절하여 자손 생성범위를 조절하고, PCX

연산자에 사용되는 부모를 매 자손마다 다르게 선택하여 자손이 부모로부터 다양한 방향에 발생하도록 하였다.

생성된 자손과 대치될 부모의 선택은 랜덤한 부모와 토너먼트 선택을 이용한 부모를 사용하며 자손의 대치확률은 줄여서 자손의 생성을 제한하기 위하여 사용하였다.

최근의 실수 최적화 알고리즘에서는 기존의 세대형 유전 알고리즘과 달리 점진형 (steady state)이기 때문에, 교배 연산자에 사용되는 부모의 수와 생성된 자손의 수가 일치 하지 않으며, 연산자와 알고리즘의 특성에 따라 사전 실험을 통하여 최적의 부모, 자손의 수를 정의하여야 한다[3][6].

SGG 알고리즘에서 실험을 통하여 가장 우수한 경우의 λ, μ 그리고 k는 실험을 통하여 구하였다(λ = 2, μ = 2, k = 12). 군집의 크기에 대한 성능을 실험해 보면 기존의 알고리즘은 문제의 크기가 증가하면 군집의 크기를 크게[7] 하여야 성능이 우수하였지만 SGG 알고리즘에서는 문제의 크기와 군집 크기가 선형적인 관계가 없으며 군집의 크기 P = 100으로 설정할 때가 가장 우수한 결과가 나왔다.

3. 자손의 대치확률 조절과 대치 방법의 영향

2.2절에서 제안한 자가 적응 세대차 알고리즘인 SGG에서, 자가 적응방식을 사용하지 않고, PCX 연산자의 기본 ω_ζ = 0.1 값을 사용하였을 경우, 자손의 대치확률은 평균적으로 70%의 수치가 나오며 G3 알고리즘(평균 92%)과 비교하여 자손의 대치확률이 낮아지며 성능도 우수해 짐을 알수 있었다.

기본 SGG 알고리즘의 영향에 의해서, 같은 PCX 연산자를 사용한 G3 보다 대치확률이 약 18%정도 낮아짐을 확인하였고, 이를 토대로 하여 PCX 연산자의 ω_ζ 를 변화시킴으로서 진화속도의 한 지표인 부모의 대치확률을 조절할수 있는 가능성을 발견하였다.

따라서, 우수한 대치 확률값을 결정하기 위하여 기본 SGG 알고리즘에 ω_ζ 값을 변화시켜 가며 실험하였고, Rosenbrock 함수에 대하여 70%, 65% 그리고 60%의 자손의 대치확률이 나올때의 성능값을 구할수 있었다.

아래 표 3은 Rosenbrock 함수에 대하여, 10번씩 실험하여, 전역최적해가 10^{-20}까지 될 때의 평가 계산량을 표시한 것으로, 수치가 낮을수록 성능이 우수함을 의미한다.

표 3. 자손의 대치확률에 대한 실험

	F_{ros}(n=20)
G3-(92%), ω_ζ = 0.1	21,452
SGG-(70%), ω_ζ = 0.1	19,992
SGG-(65%), ω_ζ = 0.18	**19,200**
SGG-(60%), ω_ζ = 0.3	22,452

G3 및 기본 SGG 알고리즘에 대해서 자손의 대치확률이 65%에서 가장 우수한 결과가 나왔으며, 60%로 더 낮아지게 되면 오히려 성능이 떨어짐을 알수 있다. 그림 1에 SGG 의 각 대치 비율에 대한 수렴 양상이 나와 있다.

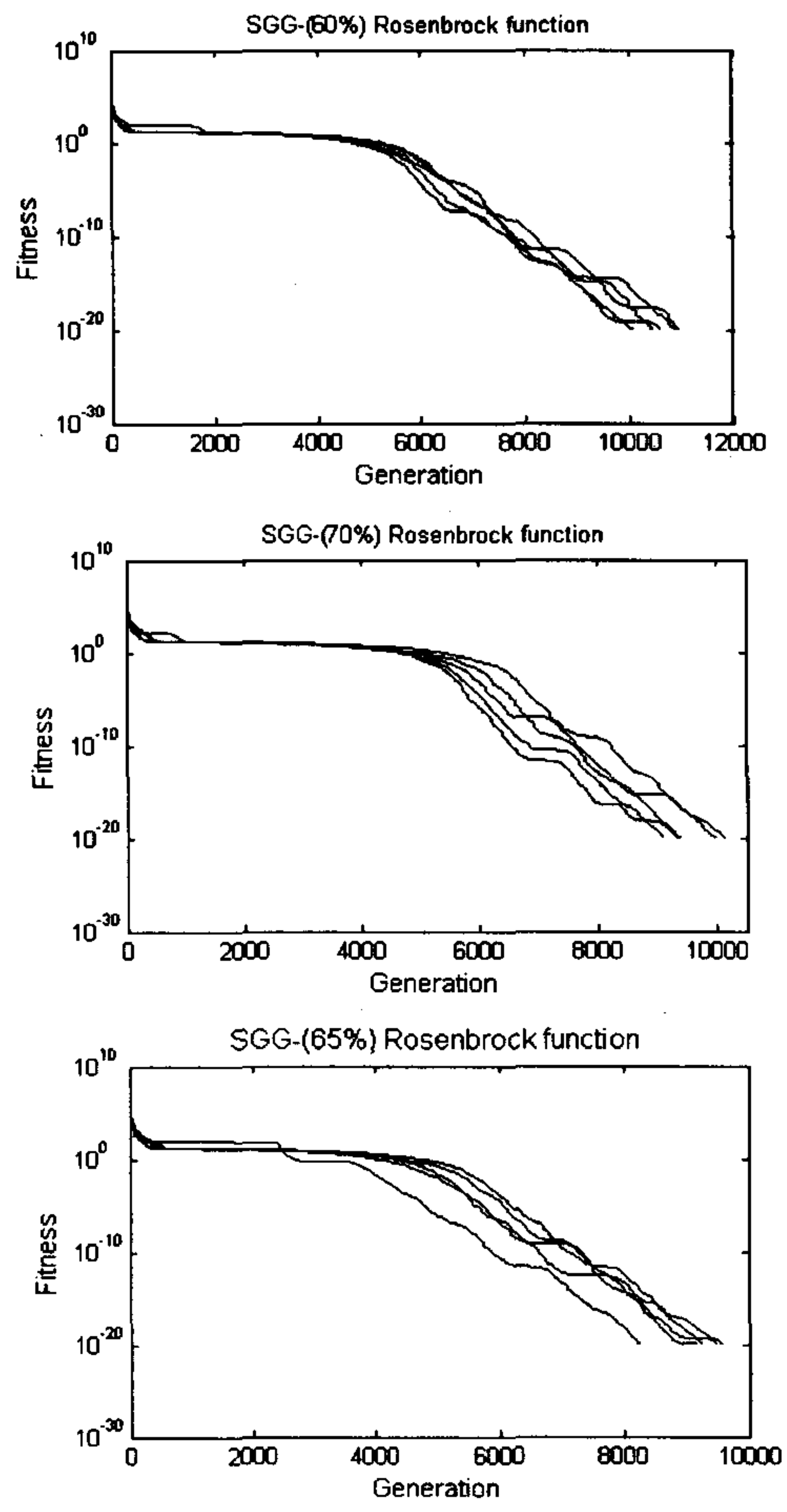

그림 1. 기본 SGG 알고리즘의 수렴양상

탐색성능이 가장 우수한 65% 대치확률을 꾸준히 유지하기 위하여는 고정된 ω_ζ 로서

는 불가능하며, 진화과정동안 상황에 따라 적응적으로 ω_ζ를 적절하게 조절할수 있어야 한다. 본 연구에서는 교배 연산자에 선택되는 부모간의 거리와 현 세대까지 누적된 자손의 대치확률을 이용하여 자가 적응적인 조절 식을 다음 식 (2)와 같이 구현하였다.

$$Dist_{(g)} = Dist_{(g-s)} + \sum_{j=1}^{\mu}\left(\sum_{i=1}^{n} X_{best} - X_j(i)\right)/u$$

$$\omega_\zeta = \left|N(0\ ,\ \frac{\log(\lambda_{Target} - \lambda_{suc}(g))}{Dist(g-s) - Dist(g)})\right| \quad (2)$$

$$\lambda_{Target} = 0.7\ ,\ s = 60$$

여기서, g는 현세대이며 $\lambda_R(g)$은 현세대까지의 누적된 대치확률이다. s는 일정한 세대마다 ω_ζ를 보상해주는 구간이며 개체의 수렴도는 교배연산자에 선택되어진 μ개의 임의의 부모와 가장 우수한 염색체 X_{best}와의 거리를 계산하여 $Dist_{(g)}$의 누적으로 정의한다.

4. 실험 및 성능 비교

실험은 아래 식(3), (4) 그리고(5)의 3가지 벤치마크 문제를 이용하여 전역극소 값이 10^{-20}이하까지 도달할 때까지의 평가 계산량(no. of evaluations)으로 성능을 표시한다. 3개의 벤치마크 문제에 대하여 20번의 실험을 반복하였으며, 문제의 변수 수는 n = 20 으로 설정하였다.

2장에서 언급한 대로 u = 2, λ = 2, k = 12 그리고 군집의 크기 P = 100으로 설정하였으며, ω_ζ를 자가 적응시켜서 자손의 대치확률을 65%로 조절할 수 있게 구현한 SGG 알고리즘을 수행하였다.

$$F_{elp} = \sum_{i=1}^{n} i x_i^2 \quad (Ellipsoidal\ function) \quad (3)$$

$$F_{sch} = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2 \quad (Schwefel\ function) \quad (4)$$

$$F_{ros} = \sum_{i=1}^{n-1}(100(x_{i+1} - x_i^2) + (1 - x_i)^2) \quad (5)$$
$$(Rosenbrock's\ function)$$

기존의 우수한 알고리즘들인 (1,10)-ES, (15,100)-ES[5], CMA-ES[1], DE[8] 그리고 G3[3]의 결과와 비교하였다.

표 4. 성능평가

함수 / EA	F_{elp}	F_{sch}	F_{ros}
(1,10)-ES	40,850	105,630	803,800
(15,100)-ES	108,400	218,400	837,804
CMA-ES	8,472	15,602	33,048
DE	12,033	119,170	587,920
G3	6,800	15,600	21,452
SGG	**6,202**	**14,700**	**18,962**

표 4 와 같이 3가지 벤치마크 문제에 모두에 대하여 기존의 알고리즘보다 우수한 성능을 확인 할 수 있다. 기존 결과중 가장 우수한 G3(+PCX) 알고리즘과 비교하여 F_{elp}에서는 8.7%, F_{sch}에선 5.7% 그리고 F_{ros}에서는 11.6%의 성능이 향상되었다.

5. 결 론

본 논문은 실수 표현 진화 알고리즘의 진화 속도를 자가 적응적으로 조절하여 탐색 성능을 개선시킬수 있는 SGG(Self-adaptation Generation Gap)을 제안하고 구현하였다. SGG 알고리즘은 선택과 대치를 포함한 진화방식을 개선하고, 유전 연산자에 의하여 생성된 자손의 대치확률에 따라서 자손의 생성범위를 자가 적응적으로 조절하여, 보다 적은 계산량으로 전역 최적화를 찾는다. 제안된 방법을 3가지 주요 벤치마크 테스트 문제에 적용하여 실험하였으며, 기존의 다른 알고리즘에 비해서 모두 향상된 성능을 나타내었다. 향후 진화속도에 대한 심화된 고찰과 부모의 대치확률에 대한 자가적응 조절방식의 체계화가 필요하다.

감사의 글 : 이 논문은 2007년 정부재원(교육인적자원부 학술연구조성사업비)으로 한국학술진흥재단의 지원을 받아 연구되었음 (KRF-2007-314-D00176)

참 고 문 헌

[1] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies," IEEE Conference. on Evolutionary Computation. pp. 312 - 317, 1996.

[2] A. E. Eiben and T. Back, "Empirical investigation of multiparent recombination operators in evolution strategies," Evolutionary Computation. pp. 5(3): 347-365, 1997.

[3] K. Deb, A. Anand and D. Joshi, " A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization." Evolutionary Computation. pp. 10(4): 371-395, 2002.

[4] T. Higuchi, S. Tsutsui and M. Yamamura, "Theoretical analysis of simplex crossover for real-code genetic algorithm." Parallel Problem Solving from Nature (PPSN-VI), pp. 365-374, 2000.

[5] H. G. Beyer, *The Theory of Evolution Strategy*. Springer, Berlin, Germany, 2001.

[6] H. Kita, I. Ono and S. Kobayashi, " Multi-parental Extension of the Unimodal Normal Distribution Crossover for Real-Code Genetic Algorithm" IEEE press, Syst. Man. Cybern, pp. 1581-1587, 1999.

[7] D. E. Goldberg and J. H. Clark " Genetic algorithms, noise and the sizing of populations." Complex Syst. pp. 6(4): 333-362. 1996.

[8] R. Stone and K. Prince, *Difference Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Space.* J. Global Optimization. pp. 341-359, 1997.