論文99-36C-3-7

신경회로망의 학습 알고리듬을 이용하여 돌연변이를 수행하는 새로운 진화 프로그래밍 알고리듬

(A New Evolutionary Programming Algorithm using the Learning Rule of a Neural Network for Mutation of Individuals)

林 種 和 *, 崔 斗 鉉 *, 黃 燦 植 *

(Jong Hwa Lim, Doo Hyun Choi, and Chan Sik Hwang)

요 약

진화 프로그래밍은 두 가지 요소로 특징지을 수 있다. 하나는 선택 방법이고 나머지는 돌연변이 규칙이다. 본 논문에서는 신경회로망의 역전파 학습 알고리듬을 이용하여 돌연변이 연산을 수행하는 새로운 진화 프로그래밍 알고리듬을 제안한다. 신경회로망의 학습 알고리듬에서 현재 오차는 진화 프로그래밍의 개체가 진화해 나가야 할 방향을 지정해 주고, 관성은 개체의 변이에 지금까지의 진화 경향을 더해 주어서 빠르게 전역 최적해를 찾도록 하였다. 표준 테스트 함수를 이용하여 제안된 알고리듬의 성능과 강건함을 확인하였다.

Abstract

Evolutionary programming is mainly characterized by two factors; one is the selection strategy and the other the mutation rule. In this paper, a new mutation rule that is the same form of well-known backpropagation learning rule of neural networks has been presented. The proposed mutation rule adapts the best individual's value as the target value at the generation. The temporal error improves the exploration through guiding the direction of evolution and the momentum speeds up convergence. The efficiency and robustness of the proposed algorithm have been verified through benchmark test functions.

Ⅰ. 서 론

지난 30년 동안 적응과 최적화의 문제를 풀기 위해서 생물학적 진화 체계를 모방하려는 많은 시도들이 있었다. 이런 모든 시도들을 일컬어 진화 연산이라 한다. 이는 문제의 해를 많은 수의 개체로 구성되는 집단으로 표현하고 더 나은 개체를 찾기 위해서 확률 연산자를 이용한다. 이러한 진화 연산은 개체의 표현 방법, 진화 연산자의 종류, 선택 방법, 그리고 재생산 방법에 따라서 유전 알고리듬(genetic algorithm : GA), 진화 전략(evolutionary strategy : ES), 그리고 진화 프로그래밍(evolutionary programming : EP) 등으로 구분된다. 유전 알고리듬은 1960년대에 미국의 Holland가 처음 제안하였고, De Jong이 이를 변수 최적화에 적용하였다. 진화 전략은 1960년대 독일의 Rechenberg가 처음 제안했고, 그 후 Schwefel에 의해 개선 발전되었으며 진화 프로그래밍은 Fogel이 제안하였다. 이런 진화 연산은 계획, 설계, 모사(simulation), 제어, 분류 등의 다양한 최적화 분야에

* 正會員, 慶北大學校 電子電氣工學部
(School of Electronic & Electrical Engineering, Kyungpook National University)

接受日字:1998年9月21日, 수정완료일:1998年12月28日

서 원하는 목적을 성취할 수 있는 강건한 알고리듬을 개발하기 위해서 계속 연구되고 있다[1].

진화 프로그래밍은 수학적인 해석에 의존하는 전통적인 방법(gradient descent, analytical discovery, etc.)이 잘 풀지 못하는 비선형적이고 미분이 불가능하거나 힘들며, 많은 극값을 가지는 문제에 좋은 성능을 보이는 것으로 알려져 있다[1]. 특히, 주어진 최적화 문제에 대한 사전 지식이 없는 경우에도 적합도 함수만 정의된다면 모든 최적화 문제에 효과적이고도 일반적으로 적용할 수 있다. 이런 장점에도 불구하고 진화 프로그래밍은 지역 최적해로 수렴하는 조기 수렴 등 여러 가지 문제를 가진다. 이러한 문제를 해결하기 위해서 새로운 돌연변이 연산자를 이용하는 알고리듬들이 제안되어 왔다[2, 3].

대표적인 진화 프로그래밍 알고리듬으로 개체의 공간과 파라메터의 공간을 동시에 탐색하는 자기 적응 진화 프로그래밍(self-adaptive evolutionary programming : SAEP)[4]이 있고 새로운 돌연변이 연산으로 성능 향상을 시도하는 가속 진화 프로그래밍(accelerated evolutionary programming : AEP)[2]과 고속 진화 프로그래밍(fast evolutionary programming : FEP)[3]이 있다. AEP나 FEP는 개체에 방향 연산자와 나이 변수를 결합하여 수렴을 가속시킨다.

본 논문에서는 진화 프로그래밍의 성능 향상을 위해서 역전파 학습 알고리듬의 현재 오차(temporal error)와 관성(momentum)의 개념을 도입한 새로운 돌연변이 연산자를 제안한다. 제안된 돌연변이 규칙에서 현재 오차는 개체의 진화 방향을 인도하고 관성은 개체의 수렴 속도를 가속시키는 역할을 수행해서 빠르게 전역해를 찾을 수 있도록 하였다.

본 논문의 구성은 다음과 같다. II장에서는 기존의 대표적인 몇몇 알고리듬과 제안된 알고리듬에 대해서 알아보고, III장에서는 제안된 알고리듬으로 수행한 실험 결과와 이에 대해 고찰한다.

II. 제안된 새로운 진화 프로그래밍 알고리듬

1960년대에 Fogel에 의해서 개발된 진화 프로그래밍은 확률 연산자를 사용하는 확률적인 최적화 전략(stochastic optimization strategy)이다[5]. 진화 프로그래밍은 전통적인 최적화 방법들이 해결할 수 없는 문제에 대해서도 유용하게 사용할 수 있는 최적화 기법이다. 여기에서는 진화 프로그래밍에 사용되는 진화 연산자와 기존의 진화 프로그래밍에 대해서 살펴본다.

1. 진화 프로그래밍에서 사용되는 진화 연산자

진화 프로그래밍에서 사용되는 진화 연산자는 돌연변이 연산자와 선택 연산자뿐이다. 각각에 대해서 살펴보자.

진화 프로그래밍에서 주로 사용되는 돌연변이 연산은 이전의 개체에 평균이 0이고 특정 표준 편차를 가지는 정규 분포를 따르는 난수를 더하는 방법을 사용한다. 대표적인 돌연변이 방법은 식 (1)과 같다.

$$x_i' = x_i + \sigma_i \cdot N_i(0,1) \tag{1}$$

여기에서 x_i는 개체 x의 i번째 변수를 나타내고 σ_i는 i번째 변수의 스텝 사이즈로 모든 변수가 동일한 값을 가진다. Schwefel은 개체의 변수뿐만 아니라 스텝 사이즈까지도 돌연변이를 수행하는 SAEP를 제안했다.

선택 연산자는 다음 세대의 부모를 선택하는 역할을 가진다. 선택 연산자의 종류로는 (1 + 1) 전략, (μ + λ) 전략 등이 있다. (1 + 1) 전략은 부모와 그의 자손 개체가 경쟁을 하여 더 나은 개체를 다음 세대의 부모로 선택한다. 이 방법은 현재는 적합도가 낮지만 전역 최적해를 생성할 가능성이 높은 개체를 남겨둘 수 있는 장점이 있다. (μ + λ) 전략은 μ개의 부모가 λ개의 자손 개체를 만들고 부모와 자손의 합집합에서 다음 세대의 μ개의 부모를 선택한다. 이 방법은 개체들이 지속적으로 진화되도록 하지만 항상 우수한 개체만을 다음 세대의 부모로 선택하기 때문에 깊은 지역해에 빠질 경우 벗어나지 못할 수도 있다.

2. 자기 적응 진화 프로그래밍(SAEP)

Schwefel에 의해 제안된 SAEP는 변수 **x**뿐만 아니라 스텝 사이즈 σ도 돌연변이 연산을 수행하여 해를 빠르게 찾도록 하는 알고리듬이다. 이 알고리듬에서 개체는 다음과 같이 표현된다.

$$a = [x_1, \cdots, x_n, \sigma_1, \cdots, \sigma_i]^T \tag{2}$$

SAEP의 돌연변이 연산은 이전의 스텝 사이즈에 로그-노말(log-normal)형태의 확률 분포를 가지는 난수를 곱해서 스텝 사이즈를 먼저 돌연변이 시킨 후 개체를 돌연변이 시킨다. 돌연변이 방법은 다음과 같다[6].

$$\sigma_i' = \sigma_i \cdot \exp(\tau \cdot N(0,1) + \tau' \cdot N_i(0,1))$$
$$x_i' = x_i + \sigma_i' \cdot N(0,1) \quad (3)$$

여기에서 학습률 $\tau' \propto (\sqrt{2n})^{-1}$과 $\tau \propto (\sqrt{2\sqrt{n}})^{-1}$는 Schwefel에 의해 경험적으로 정해졌지만 이 값은 최적화 목적 함수의 특성에 따라서 다른 값을 가질 수도 있다. 그리고 스텝 사이즈는 지수형의 값이 곱해지므로 항상 양의 값을 유지할 수 있다. Fogel은 위와는 달리 스텝 사이즈의 돌연변이에 로그-노말 형태의 난수 대신 정규 분포를 따르는 난수를 더해 주는 방법을 사용한다. 돌연변이는 아래의 식과 같이 수행된다.

$$\sigma_i' = \sigma_i + \alpha \cdot \sigma_i \cdot N_i(0,1)$$
$$x_i' = x_i + \sigma_i' \cdot N(0,1) \quad (4)$$

여기에서 α는 스케일링 계수로 양의 상수이다. 일반적으로 잡음이 첨가되지 않은 목적 함수의 경우에는 로그-노말 형태의 난수를 곱하는 돌연변이 연산 방법이 우수하고 목적 함수에 잡음이 더해져 있을 경우에는 정규 분포의 난수를 더하는 방법이 더 우수한 성능을 발휘하는 것으로 알려져 있다[1].

3. 가속 진화 프로그래밍(AEP)[2]

AEP는 개체들의 다양성을 감소시키지 않고 수렴 속도를 향상시키기 위해서 개발되었다. AEP는 개체의 적합도에 따라서 각 변수에 방향 연산자(direction operator)와 나이 변수를 개체의 파라메터에 포함한다. 그리고 자손 개체는 SAEP과 마찬가지로 평균이 0이고 표준 편차가 1인 정규 분포의 난수를 부모에 더함으로써 생성한다. 다음 세대의 부모를 선택하기 위해서는 (1 + 1) 전략이 사용된다. AEP에서 개체는 다음과 같이 표현된다.

$$x^i = [x_1^i, \cdots, x_n^i, dir(x_1^i), \cdots, dir(x_n^i), age^i]^T \quad (5)$$

여기에서 $dir(x_j^i) \in \{-1, 1\}$은 i번째 개체의 j번째 변수의 진화 방향을 나타내고, age^i는 i번째 개체가 생존한 기간을 나타낸다. 그리고 AEP에서 개체 변이는 아래와 같이 수행된다.

If $age^i[k] = 1$

then :

$$\sigma^i = \beta_1 \cdot f(x^i[k]);$$
$$x_j^i[k] = x_j^i[k-1] + dir(x_j^i[k-1]) \cdot |N(0, \sigma^i)|; \quad (6)$$

else :

$$\sigma^i = \beta_2 \cdot f(x^i[k]) \cdot age^i[k];$$
$$x_j^i[k] = x_j^i[k-1] + \sigma^i \cdot N(0,1); \quad (7)$$

$\forall_i \in \{1, 2, \ldots, N_p\}, \ \forall_j \in \{1, 2, \ldots, n\}$

식 (6)에서 $N(0, 1)$은 정규 분포의 난수를 나타내고, $|\cdot|$는 절대값을 나타낸다. β_i는 양의 상수를 나타낸다. 그리고 방향 연산자와 나이 변수는 부모와 자손의 적합도에 따라서 다른 값을 가진다. 아래의 식은 방향 연산자와 나이 변수의 계산 방법을 나타낸다.

If $(f(x^i[k]) < f(x^i[k-1]))$

then :

$$dir(x_j^i[k]) = sgn(x_j^i[k] - x_j^i[k-1]);$$
$$age^i[k] = 1; \quad (8)$$

else :

$$age_j^i[k] = age_j^i[k-1] + 1;$$

$\forall_i \in \{1, 2, \ldots, N_p\}, \ \forall_j \in \{1, 2, \ldots, n\}$

식 (8)에서 $f(x^i[k])$는 i번째 개체의 적합도를 나타낸다. 그리고 $sgn()$은 방향 함수이다.

4. 제안된 진화 프로그래밍 알고리듬

본 논문에서는 진화 프로그래밍의 돌연변이 연산에 신경회로망의 역전파 알고리듬을 사용한 새로운 진화 프로그래밍 알고리듬을 제안했다. 이 알고리듬을 통해서 신경회로망의 학습 알고리듬을 EP의 돌연변이 규칙으로 사용할 수 있음을 보였다.

진화 프로그래밍에서 돌연변이는 적합함수를 최적화하기 위해서 개체에 정규 분포의 난수를 더하는 작업이고, 신경회로망에서 학습은 평균제곱오차(RMS error)가 최소가 되도록 각 노드사이의 연결강도를 변화시키는 작업이다. 진화 프로그래밍과 신경회로망은 변화시키는 개체, 방법, 그리고 평가방법은 다르지만 두 알고리듬의 궁극적 목표는 더 빨리 더 나은 해를 찾는 것이다.

본 논문에서는 신경회로망의 역전파 알고리듬[7]에서 연결강도를 갱신하기 위해서 사용하는 알고리듬을 이용하여 개체 변이를 수행하는 새로운 진화 프로그래밍 알고리듬을 제안한다. 역전파 학습 알고리듬은 평균제곱오차가 최소가 되도록 연결강도를 변화시키지만, 제안된 진화 프로그래밍의 돌연변이 규칙은 적합함수의 적합도가 최대 혹은 최소가 되도록 개체를 변화시킨다. 제안된 알고리듬은 역전파 학습 알고리듬에서처

럼 현재 오차를 이용하여 개체의 진화 방향을 인도할 뿐만 아니라 진화의 스텝 사이즈도 능동적으로 적응시킬 수 있다. 또한, 관성을 추가해서 개체의 이전 세대까지의 진화 경향이 현재의 변화에 영향을 주도록 하였다. 여기서 관성은 현재의 변화가 이전의 변화와 동일하면 개체의 변화량을 증가시키고 그렇지 않으면 개체의 변화량을 감소시키는 역할을 한다.

본 논문에서 제안된 진화 프로그래밍은 새로운 개체를 다음과 같은 방법으로 만든다.

$$x_j^i[k+1] = x_j^i[k] + \eta \cdot \Delta x_j^i[k] + \alpha \cdot sx_j^i[k] \quad (9)$$

여기에서 η와 α는 양의 실수로 고정된 값이고, $x_j^i[k]$는 k세대의 i번째 개체의 j번째 변수를 나타낸다. 그리고 $\Delta x_j^i[k]$는 k세대의 i번째 개체의 j번째 변수의 변화량을 나타내고 식 (10)에 따라 계산되어진다.

$$\Delta x_j^i[k] = (x_j^{best}[k] - x_j^i[k]) \cdot |N(0,1)| \quad (10)$$

개체의 변화량은 가장 우수한 개체와 자신의 차이에 정규 분포를 가지는 난수의 절대값을 곱해서 계산된다. 이 방법을 사용하면 개체는 항상 현재의 가장 우수한 개체에 접근하도록 변화되므로 해를 빨리 찾을 수 있다. 그리고 $sx_j^i[k]$는 k세대까지 개체의 진화 경향을 나타내고 식 (11)로 표현된다.

$$sx_j^i[k] = \eta \cdot acc^i[k-1] \cdot \Delta x_j^i[k-1] + \alpha \cdot sx_j^i[k-1] \quad (11)$$

식 (11)은 개체가 현재까지 변화해온 경향을 나타내는 식으로 현재의 변화가 이전의 변화 추세와 동일할 경우에는 그 변화량을 증가시키는 역할을 한다. 여기에서 $acc^i[k]$는 i번째 개체의 수용 파라메터(acceptance parameter)를 나타낸다. 수용 파라메터는 새로운 개체가 다음 세대의 부모로 선택되는가에 따라서 식 (12)와 같은 값을 가진다.

$$acc^i[k] = \begin{cases} 1, & \text{if } f(x^i[k-1]) - f(x^i[k]) > 0 \\ 0, & otherwise \end{cases} \quad (12)$$

제안된 돌연변이 규칙은 개체의 변화량을 계산하기 위해 현재 오차를 계산할 때 고정된 목표 출력이 아닌 현재 세대에서 가장 우수한 개체를 사용하고 입력 대신에 정규 분포의 난수를 사용한다. 그리고 개체의 변화량을 누적하는 계산에서도 수용 파라메터를 두어 가능하면 지속적인 향상을 이루도록 하였다. 이런 차이만 제외한다면 본 논문에서 제안된 돌연변이 연산 방법은 역전파 알고리듬의 학습 방법과 완전히 동일한 모양을 하고 있다. 본 논문에서는 역전파 알고리듬만 사용했지만 신경회로망의 다른 학습 알고리듬도 위의 방법과 동일한 방법으로 진화 프로그래밍의 돌연변이 규칙으로 사용될 수 있으리라 기대된다.

Ⅲ. 실험 결과 및 고찰

제안된 알고리듬의 성능을 평가하기 위해서 여러 표준 테스트 함수에 대해서 실험을 수행하고 그 결과를 고찰해 본다. 먼저 진화 연산에서 그 성능을 확인하기 위해서 자주 사용되는 표준 테스트 함수에 대해서 살펴보자.

1. 표준 테스트 함수

최적화 문제를 풀기 위해서 제안되는 많은 진화 알고리듬의 성능을 정확하게 비교하기 위해서 여러 종류의 표준 테스트 함수들이 있다. 이들은 각 함수마다 다른 특별한 모양을 하고 있으며 이들 각각의 테스트 결과는 제안된 진화 알고리듬의 특성을 잘 반영한다. 본 논문에서 제안된 알고리듬과 다른 알고리듬을 동일한 테스트 함수에 대해서 수행함으로써 제안된 알고리듬과 다른 알고리듬의 성능을 비교하였다. 먼저 본 논문에 사용된 테스트 함수를 살펴보자[8].

함수 1 : Sphere function(De Jong F1)

$$f(x_1, x_2, x_3) = \sum_{i=1}^{3} x_i^2, \quad where \ -6 \le x_i \le 6$$

이 함수는 $(x_1, x_2, x_3) = (0,0,0)$일 때 전역 최소값 0을 가진다.

함수 2 : Rosenbrock function(De Jong F2)

$$f(x_1, x_2) = 100 \cdot (x_1^2 - x_2)^2 + (1 - x_1)^2, \ where -2.047 \le x_i \le 2.048$$

이 함수는 $(x_1, x_2) = (0,0)$일 때 전역 최소값 0을 가진다.

함수 3 : Foxholes function(De Jong F5)

$$f(x_1, x_2) = \frac{1}{\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}}, \quad whrer \ -65 \le x_i \le 65$$

$$[a_{ij}] = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & \cdots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & \cdots & 32 & 32 & 32 \end{bmatrix}$$

이 함수는 $(x_1, x_2) = (-32, -32)$일 때 전역 최소값 0을 가진다.

함수 4 : Colville Function

$$f(x_1,\cdots,x_4)=100\cdot(x_2-x_1^2)^2+(1-x_1)^2+90\cdot(x_4-x_3^2)^2+(1-x_3)^2+10.1\cdot((x_2-1)^2+(x_4-1)^2)+19.8\cdot(x_2-1)\cdot(x_4-1),\quad whrer-10\le x_i\le 10$$

이 함수는 $(x_1,\cdots,x_4)=(1,1,1,1)$일 때 전역 최소값 0을 가진다.

함수 5 : Griewangk Function

$$f(x_1,\cdots,x_{10})=\sum_{i=1}^{10}\frac{x_i^4}{4000}-\prod_{i=1}^{10}\cos(\frac{x_i}{\sqrt{i}})+1,\quad where\ -60\le x_i\le 60$$

이 함수는 $(x_1,\cdots,x_{10})=(0,.\cdots,0)$일 때 전역 최소값 0을 가진다.

위에 열거한 함수 중에서 Sphere function은 오직 하나의 전역해를 가지는 가장 기본적인 테스트 함수로 알고리듬의 수렴 속도를 시험할 수 있다. Rosenbrock function이나 Colville function도 오직 하나의 전역해를 가지지만 완만한 기울기를 가지는 모양으로 조기 수렴 현상 여부를 실험할 수 있다. 그리고 나머지 함수는 많은 지역해를 가지는 함수들이다. 따라서 위에 열거한 테스트 함수들로 제안된 알고리듬의 대략적인 성능을 가늠할 수 있다.

2. 실험 결과 및 고찰

실험에 사용된 모든 알고리듬에 대해서 개체 수는 45개로 하였다. 그리고 임의적인 초기값의 효과를 무시하고 알고리듬의 성능을 확인하기 위해서 10번의 반복 수행 후 평균값을 비교하였다. SAEP에서 초기 표준 편차는 0에서 1사이의 임의의 값으로 설정하였고, 선택은 $(\mu+\mu)$ 전략을 사용하였다. 그리고 AEP에서 식 (6)과 (7)에 사용된 β_1, β_2는 각각 5.0과 0.005로 설정하였다. 그리고 초기 개체의 나이는 모두 1의 값을 가지고 초기 방향은 -1혹은 1의 값으로, 초기 표준 편차는 0에서 1사이의 임의의 값으로 설정하였다. 제안된 알고리듬에 대해서는 $(\mu+\mu)$ 전략과 (1 + 1) 전략으로 실험을 수행하였고, 학습계수와 관성계수는 $(\mu+\mu)$ 전략인 경우에는 1.8과 0.2로 (1 + 1) 전략인 경우에는 1.5와 0.25로 설정하였다. 계수들은 각 알고리듬이 가장 좋은 성능을 보인 계수들을 정한 것이다.

그림 1에서 그림 5까지는 각 함수에 대한 평균 최소 적합도의 진화를 나타낸다. SAEP는 Sphere나 Griewangk function과 같은 포물선 형태(parabolic)를 갖는 함수에 대해서 비교적 우수한 성능을 보였다.

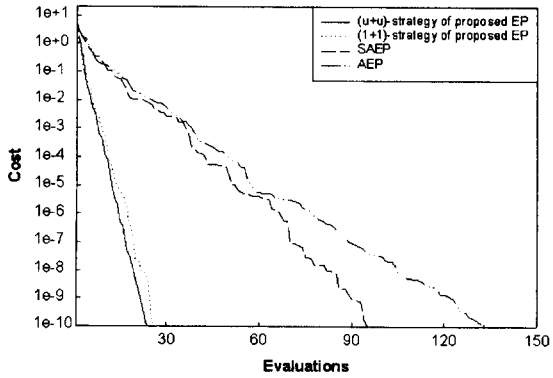


그림 1. 함수 1에 대한 평균 최소 적합도의 진화
Fig. 1. Evolution of the averaged minimum cost for function 1.

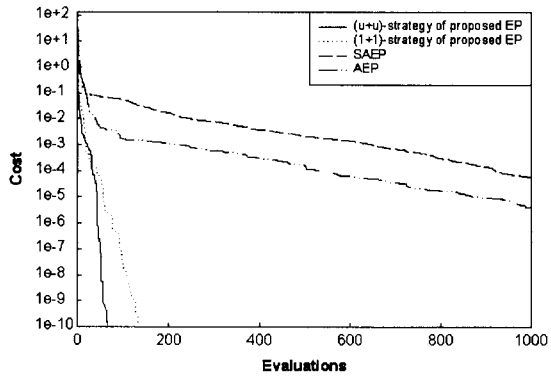


그림 2. 함수 2에 대한 평균 최소 적합도의 진화
Fig. 2. Evolution of the averaged minimum cost for function 2.

그리고 Foxholes function과 같이 깊은 지역해를 가지는 함수(multimodal function)에 대해서는 AEP가 비교적 우수한 성능을 보였다. 그러나 Rosenbrock이나 Colville function에 대해서는 SAEP나 AEP 둘 다 조기 수렴 현상을 보였다. 그러나 제안된 알고리듬은 실험에 사용된 모든 함수에 대해서 전역해를 아주 빠르게 찾아감을 보여주고 있다. Sphere와 Griewangk function에 대해서는 SAEP보다도 훨씬 빠르게 전역 최적해로 수렴했다. 그리고 SAEP나 AEP가 아주 느리게 수렴해 가는 Rosenbrock function에서도 그림 2에서 보이는 바와 같이 제안된 알고리듬은 아주 빠르게 전역해를 찾았고, SAEP나 AEP가 조기 수렴 현상을 보이는 Colville function에 대해서도 그림 4에서 보이는 바와 같이 제안된 알고리듬은 계속해

서 최적해로 수렴해 감을 볼 수 있다. 그리고 Foxholes function과 같이 깊은 지역해를 가지는 함수의 경우에 (1 + 1) 전략은 아주 빠르게 전역해로 수렴하지만 (μ + μ) 전략의 경우에는 전역해를 전혀 찾지 못한다. 이는 (1 + 1) 전략은 부모와 자손간의 경쟁으로 지금은 성능이 열악하지만 나중에 전역해를 생성할 가능성이 있는 개체를 보존하므로 지역해에서 벗어날 기회를 가지는 반면 (μ + μ) 전략은 항상 우수한 개체만을 다음 세대의 부모로 선택하기 때문에 깊은 지역해에 빠지게 되면 지역해를 벗어날 가능성을 잃게 된다.

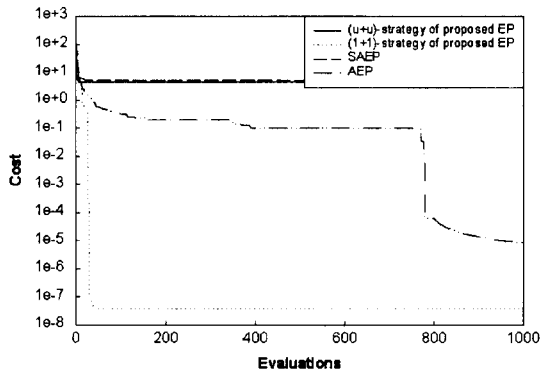


그림 3. 함수 3에 대한 평균 최소 적합도의 진화
Fig. 3. Evolution of the averaged minimum cost for function 3.

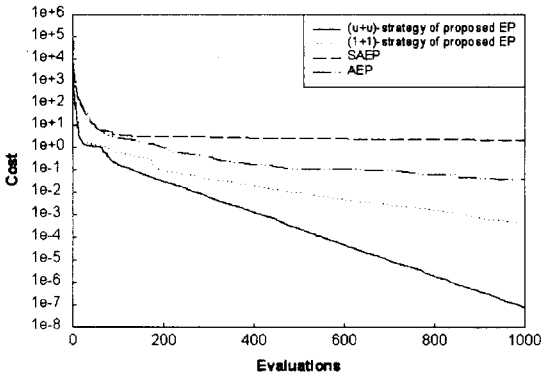


그림 4. 함수 4에 대한 평균 최소 적합도의 진화
Fig. 4. Evolution of the averaged minimum cost for function 4.

실험 결과로 알 수 있듯이 제안된 알고리듬은 SAEP나 AEP에 비해 탁월한 성능을 보였다. 이는 SAEP는 새로운 개체를 탐색하는 방법으로 완전히 임의적인 방법을 사용하지만 제안된 알고리듬은 그 세대에서 가장 우수한 개체를 따라가도록 스텝 사이즈를 능동적으로 조절해 준다. 그리고 AEP은 부모와 자손 사이의 성능에 따라서 스텝 사이즈를 능동적으로 조정하지만 제안된 알고리듬은 스텝 사이즈의 능동적인 조정뿐만 아니라 개체가 현재까지 진화해온 경향을 부모와 자손사이의 관계에 따라서 더해주기 때문에 더욱 빠른 수렴을 보인다.

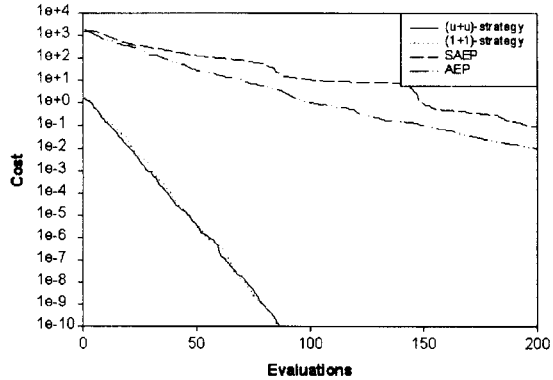


그림 5. 함수 5에 대한 평균 최소 적합도의 진화
Fig. 5. Evolution of the averaged minimum cost for function 5.

Ⅳ. 결 론

본 논문에서는 계획, 설계, 제어, 그리고 분류 등의 다양한 분야에서 나타나는 최적화 문제들을 효율적으로 풀기 위한 새로운 진화 프로그래밍을 제안했다.

제안된 EP 알고리듬은 돌연변이 규칙으로 역전파 학습 알고리듬을 사용한다. 이러한 역전파 학습 알고리듬의 현재 오차는 개체가 변화되어야 할 방향과 스텝 사이즈를 결정하고 관성은 개체가 이전까지 변화되어온 경향을 개체의 변화에 더해서 최적해를 빠르게 찾도록 한다. 제안한 돌연변이 연산을 사용하는 진화 프로그래밍은 다양한 형태의 표준 테스트 함수에 대해서 우수한 성능을 보였다. 제안된 알고리듬은 포물선 형태를 가지는 단순한 함수뿐만 아니라 조기 수렴 현상이 나타날 수 있는 완만한 경사를 가지는 함수 그리고 깊고 많은 지역해를 가지는 함수에 대해서도 빠른 수렴을 보였다. 이런 결과로 볼 때 신경회로망의 학습 알고리듬은 진화 프로그래밍의 돌연변이 연산자로서의 기능을 잘 수행한다고 볼 수 있다. 본 논문에서는 역전파 학습 알고리듬에 대해서만 실험하였지만 다른 학습 알고리듬을 사용하더라도 이와 동일한 성능 향상을

기대할 수 있을 것이다.

앞으로 남아 있는 과제는 진화 프로그래밍의 돌연변이 연산으로 사용하기에 적합한 신경회로망의 학습 알고리듬을 찾아내고, 이를 진화 프로그래밍의 돌연변이 연산을 위해 사용해 보는 것이다. 그리고 테스트 함수에서뿐만 아니라 실제 문제에 대해서도 우수한 성능을 보이는지에 대한 검증이 필요하다.

참 고 문 헌

[1] T. Back and H. P. Schwefel, "Evolutionary Computation : An Overview," *IEEE Int. Conf. on Evolutionary Computation*, pp. 20-29, 1996.

[2] J. H. Kim, H. K. Chae, J. Y. Jeon, and S. W. Lee, "Identification and Control of Systems with Friction using Accelerated Evolutionary Programming," *IEEE Control Systems*, pp. 38-47, 1997.

[3] H. J. Cho, S. Y. Oh, and D. H. Choi, "Fast Evolutionary Programming Through Search Momemtum and MultipleOffspring Strategy," *IEEE Int. Conf. on Evolutionary Computation*, pp. 805-809, 1998.

[4] T. Back, D. B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*, Oxford Univesity Press, 1997.

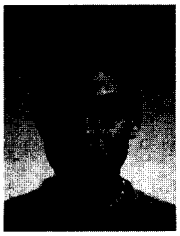
[5] J. Heitkoetter and D. Beasley, The Hitch-Hiker's Guide to Evolutionary Computation, FAQ for comp.ai.genetic, 1997.

[6] M. Palaniswami, Y. Attikiouzel, R. J. Marks II, D. Fogel, and T. Fukuda, *Computational Intelligent : A dynamic System Perspective*, IEEE Press, 1995.

[7] J. A. Freeman and D. M. Skapura, *Neural Networks : Algorithms, Applications, and Programming Techniques*, Addison-Wesley, 1991.

[8] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary programmings*, Spring-Verlag, 1996.

저 자 소 개



林 種 和(正會員)

1997년 2월 경북대학교 전자전기공학부 졸업(공학사). 1999년 2월 경북대학교 전자전기공학부 졸업(공학석사). 주관심분야는 신경회로망, 진화 연산, 진화 프로그래밍

黃 燦 植(正會員)

1977년 2월 서강대학교 전자공학과 졸업(공학사). 1979년 2월 한국과학기술원 전기전자공학과 졸업(공학석사). 1996년 2월 한국과학기술원 전기전자공학과 졸업(공학박사). 1979년 9월 ~ 현재 경북대학교 전자전기공학부 교수. 1991년 8월 ~ 1992년 8월 Univ. of Texas 전기전자공학부 Visiting Prof. 주관심분야는 영상신호처리, 암호통신

崔 斗 鉉(正會員) 第 35卷 第 11號 參照