한국 퍼지 및 지능시스템학회 논문지 2000, Vol. 10, No. 6 (2000)542~547

# 방향성 벡터를 갖는 개선된 진화프로그래밍

# The Improved Evolutionary Programming with Direction Vectors

박진현 · 배준경

**Jin-Hyun Park and Jun-Kyung Bae**

진주산업대학교 산업자동화공학과

## 요 약

진화프로그래밍(Evolutionary Programming : EP)은 최적화 문제에 있어서 매우 유용한 기법으로 자연선택의 원리를 모방한 탐색알고리즘이다. EP는 기존의 최적화 알고리즘에 비하여 여러해를 동시에 탐색하는 전역탐색(global search)방법이므로 국부수렴(local convergence)의 가능성이 줄어들고, 최적화 파라메터 영역의 연속성과 미분치의 존재성과 같은 조건이 필요 없는 장점을 갖는다. 이러한 장점에도 불구하고, EP의 탐색영역이 초기조건 및 최적화 파라메터들의 랜덤 생성 그리고 최적화에 필요한 전략적 파라메터들에 의하여 탐색 영역이 결정되고, 수렴성이 느린 단점을 갖는다. 이러한 문제를 해결하기 위하여, 본 연구에서는 빠른 수렴성과 다양성을 갖는 개선된 EP을 제안하고, 제안된 방향성 벡터를 갖는 개선된 EP를 함수 최적화 문제에 적용하여 그 성능의 유용성을 보이고자 한다.

## ABSTRACT

Evolutionary programming(EP) has become very attractive ways as search algorithms which imitate the principles of natural evolution, and have been used as methods to solve parameter optimization problems. EP has low possibility to be stuck on the local optimum points. Also, EP does not require conditions such as continuities and derivatives, which are essential to the conventional optimization methods. Although they have these merits, the search space of EP is mainly confined within a specified region by the initial condition, the random variables and strategy parameters. In order to resolve these problems, this paper proposes the improved EP with direction vectors, which has fast convergence speed and diversity. It is shown that the improved EP with direction vectors gives outstanding performance when it is applied to function minimization.

## 1. 서 론

60년대 중반부터 지속적으로 연구가 진행되어 온 진화알고리즘(Evolutionary Algorithms : EAs)은 최적화 문제에 있어서 매우 유용한 기법으로 자연선택의 원리를 모방한 탐색알고리즘이다[1-3]. 특히, 기존의 최적점 탐색방법에서와 같은 국부탐색(local search)이 아닌, 여러해를 동시에 탐색하는 전역탐색(global search)방법이므로 국부수렴의 가능성이 줄어든다. 이러한 진화알고리즘은 유전알고리즘(Genetic Algorithm : GA), 진화전략(Evolution Strategy : ES) 그리고 진화프로그래밍(Evolutionary Programming : EP)으로 크게 나누어지는데, 유전알고리즘은 1970년대 초 John Holland에 의해 도입되었으며, 1975년 Holland는 자연의 진화과정이 인공 시스템에 어떻게 적용되는가를 설명하여 유전알고리즘을 체계화하였다[1]. 그리고 진화전략은 60년대 Rechenberg에 의해 개발되어 Schwefel에 의하여 크게 발전되었으며, 같은 시기에 David Fogel에 의하여 진화프로그래밍이 제안되었다[4,5]. 진화전략과 진화프로그래밍은 유전알고리즘과 달리 코딩과 디코딩 과정이 필요없이 실제의 값을 갖는 유전자들로 구성된 벡터를 염색체로 사용하고, 돌연변이 연산자만을 사용하는 특징이 있다. 이러한 진화알고리즘은 파라메터 영역의 연속성과 미분치의 존재성과 같은 조건이 필요없으며, 다른 탐색방법에서 필요로 하는 보조적인 정보도 필요치 않는 장점이 있다. 그러므로 이러한 진화알고리즘은 제어기의 자동동조 및 최적화와 관련된 여러 분야에 많이 사용되고 있으며, 다른 최적화 방법에 비하여 뛰어난 성능을 보인다[2-4].

그러나 유전알고리즘은 전역탐색에 의해 국부수렴의 확률은 줄어드나, 집단의 크기가 작을 경우 조기수렴의 확률이 있다. 그러므로 적합도의 크기를 일정하게 조정(scale)하거나, 가장 높은 적합도를 갖는 염색체(chromosomes)를 다음 세대에 살리고 교차의 확률과 돌연변이 확률을 높이며, 집단의 크기를 적응적으로 변화시키는 연구가 진행되었다[3,6]. 그리고 진화전략과 진화프로그래밍은 수렴 속도가 떨어지는 단점

이 있으나, 돌연변이나 표준편차값들을 자기적응(self adaptation)시켜 해결하려는 방향으로 연구가 진행되었다[3-5,8].

본 연구에서는 이러한 진화알고리즘 중 진화프로그래밍의 빠른 수렴 속도를 얻기 위하여 2가지 측면에서 새로운 방향성의 도입을 적용하였다. 첫번째는 진화프로그래밍 각각의 개체에 의하여 방향성을 얻는 방법과 두번째는 각 세대의 방향성을 얻는 방법을 들 수 있다. 이러한 방향성 백터의 적용으로 제안된 진화프로그래밍의 수렴 속도를 향상시킴을 테스트 함수들을 통하여 보이고자 한다.

## 2. 제안된 진화프로그래밍

본 절에서는 진화프로그래밍의 빠른 수렴 특성과 문제점을 해결하기 위하여 방향성 벡터를 갖는 진화프로그래밍을 제안한다.

표준 진화프로그래밍은 일반적으로 몇 가지 문제점이 제기된다[3].

첫째 전역 최소값이 '0'이 아니라면, 정확한 최적점을 찾기 어렵다.

둘째 적합도 값이 매우 클 경우, 큰 돌연변이에 의해 무작위 탐색이 기대되어 표준편차의 감소로 나타난다.

셋째 최적화 문제의 정확한 전역 최소값을 모른다면 적합도와 변화분의 선형 변환이나 scale 함수의 적절한 조정으로 최적점을 찾는 것이 불가능하다.

메타 진화프로그래밍 역시 메타 파라메터들의 결정이 어렵다. 만약, 메타 파라메터 $\zeta$값이 너무 크게 될 경우, 랜덤 탐색과 같이 작용하여 수렴 속도가 떨어지고, 임의의 '0'에 가깝게 할 경우에도 수렴 속도가 크게 떨어진다. 그러므로 보다 빠른 수렴 특성을 구하기 위하여 각각의 개체에 진화의 방향성을 도입하여 수렴 속도를 향상시킨 연구가 있었다[8].

방향성은 부모 세대가 자식 세대를 만들어낼 때, 확률적으로 더 좋은 자식을 만들 수 있도록 높여줌으로써 더 좋은 진화를 유도한다. 즉 일반적인 진화프로그래밍은 집단의 진화에만 의존하지만, 방향성의 도입은 각 개체의 진화에도 방향성을 가지고 진화하게 된다. 그러므로 각 개체와 집단의 진화를 가속화하는 장점을 갖는다.

본 연구에서는 더 빠른 수렴 속도를 얻기 위하여 2가지 측면에서 새로운 방향성의 도입을 적용하였다. 첫번째는 진화프로그래밍 각각의 개체에 의하여 방향성을 얻는 방법과 두번째는 각 세대의 방향성을 얻는 방법을 들 수 있다. 각각의 개체에 의한 방향성은 부모 세대와 자식 세대간의 일대일 비교에 의하여 방향성을 얻는 방법이며, 각 세대의 방향성은 집단 내의 가장 좋은 개체의 방향으로 방향성을 얻는 방법이다. 특히 각 세대의 방향성을 얻는 방법은 각 집단들이 방향성을 가짐으로 인하여 빠른 수렴 속도를 얻을 수 있고, 일반적인 진화프로그래밍이나 기존의 방향성을 갖는 진화프로그래밍 역시 무성생식만을 하나, 이러한 집단들이 방향성을 갖도록 함으로써 유성생식이 가능하다. 즉 가장 좋은 개체의 방향으로 랜덤변수를 발생하므로 각각의 개체는 유성생식의 효과를 가진다. 그리고 진화프로그래밍의 성능을 좌우하는 랜덤변수의 크기에 큰 영향을 미치는 것은 가우시안 랜덤 분포의 표준편차뿐만 아니라 평균값이므로, 각 개체의 방향성을 가우시안 랜덤 분포의 평균값에 적용하여 보다 빠른 수렴 특성을 구할 수 있다. 즉 일반적인 진화프로그래밍은 평균이 0인 랜덤변수를 발생시키나, 제안한 진화프로그래밍에서는 평균값을 방향성 벡터에 따라 가변시켜 빠른 수렴 성능을 구하였다.

### 2.1 각각의 개체에 대한 방향성 벡터

최적화 함수의 최소값을 나타내는 추정 파라메터 벡터를 다음과 같이 정의한다.

$$\boldsymbol{a}_i = [x_{i,1},\ x_{i,2},\ \cdots,\ x_{i,n},\ dir(x_{i,1}),\ dir(x_{i,2}),\ \cdots,\ dir(x_{i,n})]^T \tag{1}$$

단, $dir(x_{i,j})$는 파라메터 $x_{i,j}$의 방향성 벡터이고, $x_{i,j}$는 한 세대 내의 $i$번째 개체의 $j$번째 파라메터를 나타낸다.

새로이 정의된 벡터 $dir(x_{i,j})$와 돌연변이 연산은 다음과 같다.

IF $(\Phi(\boldsymbol{a}_i[k]) < \Phi(\boldsymbol{a}_i[k-1]))$

$$dir(x_{i,j}) = x_{i,j}[k] - x_{i,j}[k-1]$$
$$\sigma_{i,j} = \beta_{1,i} \cdot \Phi(\boldsymbol{a}_i[k])$$
$$Av_{i,j} = \alpha_{1,j} \cdot dir(x_{i,j})$$
$$x_{i,j}[k] = x_{i,j}[k-1] + N(Av_{i,j},\ \sigma_{i,j})$$

ELSE

$$dir(x_{i,j}) = 0$$
$$\sigma_{i,j} = \beta_{2,j} \cdot \Phi(\boldsymbol{a}_i[k])$$
$$x_{i,j}[k] = x_{i,j}[k-1] + N(0,\ \sigma_{i,j})$$

$i = 1,\ 2,\ \cdots,\ \mu.\ j = 1,\ 2,\ \cdots,\ n.$

단, $\alpha_{1,j}$, $\beta_{1,j}$, $\beta_{2,j}$는 양의 실수이며, $k$는 $k$번째 세대를 의미한다. 예를 들어서 $dir(x_{i,j})[k]$는 $k$번째 세대에서 $i$번째 개체의 $j$번째 벡터의 파라메터의 방향성을 나타내는 방향 벡터이다.

$N(Av_{i,j}, \sigma_{i,j})$는 평균이 $Av_{i,j}$이고 표준편차가 $\sigma_{i,j}$인 Gaussian 확률 분포를 갖는 랜덤변수로 정의된다. 이 돌연변이 연산자가 의미하는 것은 새로이 만들어진 벡터들이 그 평가 함수가 큰 값으로 변경된 경우, 변경되기 전에 저장된 각 파라메터의 방향성 벡터대로 돌연변이를 가해 줌을 의미한다. 그리고 그 평가 함수가 부모 세대 보다 나쁠 경우, 표준 진화프로그래밍과 같이 방향성을 가지지 않은 돌연변이를 가한다는 것이다. 다시 말하면 평균 $Av_{i,j}$가 방향성 벡터에 의하여 크기와 방향이 결정되어, 그 결과에 따라 돌연변이 연산을 시킨다는 점이다.

그림 1은 2차원의 파라메터 영역에 대하여 최소값을 찾는 문제로 표준 진화프로그래밍과 각각의 개체에 방향성 벡터를 갖는 진화프로그래밍의 탐색 영역을 표시하였다. 탐색 영역의 크기는 두 진화프로그래밍이 같다고 가정하였다. $k-1$ 세대에서의 파라메터 벡터를 $\boldsymbol{x}_i(x_1, x_2)[k-1]$이라 하고, $k$세대에서의 파라메터 벡터를 $\boldsymbol{x}_i(x_1, x_2)[k]$이라고 할 때, $k+1$ 세대에서의 진화프로그래밍의 탐색구간은 표준 진화프로그래밍은 A 그리고 각각의 개체에 방향성 벡터를 갖는 진화프로그래밍은 B로 표시하였다. 표준 진화프로그래밍에 비하여 각각의 개체에 방향성 벡터를 갖는 진화프로그래밍의 파라메터들이 최소화되는 방향으로 탐색 영역이 방향성을 가지고 있음을 보인다.

### 2.2 세대에 대한 방향성 벡터

최적화 함수의 최소값을 나타내는 추정 파라메터 벡터를 다음과 같이 정의한다.

$$\boldsymbol{a}_i = [x_{i,1}, x_{i,2}, \cdots, x_{i,n}, dir_{best}(x_{i,1}), dir_{best}(x_{i,2}), \cdots, dir_{best}(x_{i,n})]^T \quad (2)$$

단, $dir_{best}(x_{i,j})$는 파라메터 $x_{i,j}$가 가장 좋은 개체의 방향으로 향하는 방향성 벡터이다.

새로이 정의된 변수 $dir_{best}(x_{i,j})$와 돌연변이 연산은 아래와 같다.

IF $(best == i)$

$dir(x_{i,j}) = 0$

$\sigma_{i,j} = \beta_{3,j} \cdot \Phi(\boldsymbol{a}_i[k])$

$x_{i,j}[k] = x_{i,j}[k-1] + N(0, \sigma_{i,j})$

ELSE

$$dir_{best}(x_{i,j}) = \frac{x_{best,j}[k] - x_{i,j}[k]}{|x_{best,j}[k] - x_{i,j}[k]|}$$

$\sigma_{i,j} = \beta_{4,j} \cdot \Phi(\boldsymbol{a}_i[k])$

$Av_{i,j} = \alpha_{2,j} \cdot dir_{best}(x_{i,j})$

$x_{i,j}[k] = x_{i,j}[k-1] + N(Av_{i,j}, \sigma_{i,j})$

$i = 1, 2, \cdots, \mu.\ j = 1, 2, \cdots, n.$

단, $\alpha_{2,j}$, $\beta_{3,j}$, $\beta_{4,j}$는 양의 실수이고, $dir_{best}(x_{i,j})[k]$는 $k$번째 세대에서 $i$번째 개체의 $j$번째 파라메터가 가장 좋은 적합도를 갖는 개체 $x_{best,j}$의 방향으로 나타내는 방향 벡터이다.

이 돌연변이 연산자가 의미하는 것은 세대 내의 벡터들 중 가장 좋은 개체의 방향으로 방향 벡터를 결정하여 세대 내의 집단들이 최적의 방향으로 방향성을 갖도록 함을 의미한다. 즉, 평균 $Av_{i,j}$가 최적의 적합도를 갖는 개체의 방향으로 단위벡터를 발생하여, 돌연변이 연산을 시킨다는 점이다.

그림 2.3은 2차원의 파라메터 영역에 대하여, 표준 진화프로그래밍과 세대에 대한 방향성 벡터를 갖는 진화프로그래밍의 탐색 영역을 표시하였다. $n+1$ 세대에서의 최적화 알고리즘의 탐색구간은 일반적인 진화

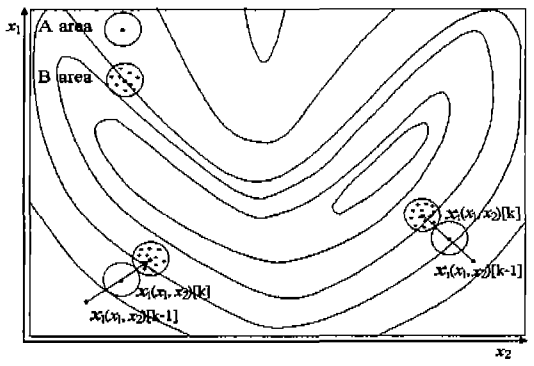


그림 1. 진화프로그래밍의 탐색 영역 비교(각각의 개체에 대한 방향성 벡터)

Fig. 1. Comparison with search space of Evolutionary Programming (Direction vectors of each individuals)

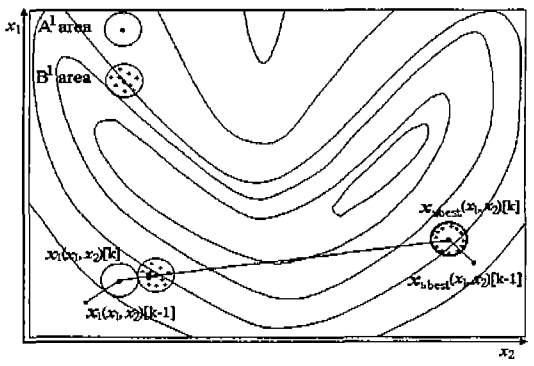


그림 2. 진화프로그래밍의 탐색 영역 비교(세대에 대한 방향성 벡터)

Fig. 2. Comparison with search space of Evolutionary Programming (Direction vectors of each generation)

프로그래밍은 $A^1$ 그리고 세대에 대한 방향성 벡터를 갖는 진화프로그래밍은 $B^1$로 표시하였다. 표준 진화프로그래밍에 비하여 방향성을 갖는 진화프로그래밍의 탐색 영역이 최적화 되는 방향으로 방향성을 가지고 있음을 알 수 있다.

### 2.3 각각의 개체와 세대에 대한 방향성 벡터의 합성

위에서 제시한 2가지 방향성 벡터를 벡터합을 행하여 최적화 문제에 적용하였다. 평가 함수의 최소값을 나타내는 추정 파라메터 벡터를 다음과 같이 정의한다.

$$\boldsymbol{a}_i=[x_{i,1}, x_{i,2}, \cdots, x_{i,n}, dir_{prod}(x_{i,1}), dir_{prod}(x_{i,2}), \cdots, dir_{prod}(x_{i,n})]^T \tag{3}$$

여기서, $dir_{prod}(x_{i,j})$는 파라메터 $x_{i,j}$가 각각의 개체에 대한 방향성 벡터와 가장 좋은 개체의 방향으로 향하는 방향성 벡터의 합이다.

정의된 $dir_{prod}(x_{i,j})$와 돌연변이 연산은 아래와 같다.

IF (best $=$ $i$)
  IF ($\Phi(\boldsymbol{a}_i[k]) < \Phi(\boldsymbol{a}_i[k-1])$)
    $dir_{prod}(x_{i,j}) = x_{i,j}[k] - x_{i,j}[k-1]$
    $\sigma_{i,j} = \beta_{1,j} \cdot \Phi(\boldsymbol{a}_i[n])$
    $Av_{i,j} = \alpha_{1,j} \cdot dir_{prod}(x_{i,j})$
    $x_{i,j}[k] = x_{i,j}[k-1] + N(Av_{i,j}, \sigma_{i,j})$
  ELSE
    $dir(x_{i,j}) = 0$
    $\sigma_{i,j} = \beta_{2,j} \cdot \Phi(\boldsymbol{a}_i[k])$
    $x_{i,j}[k] = x_{i,j}[k-1] + N(0, \sigma_{i,j})$
ELSE
  IF ($\Phi(\boldsymbol{a}_i[k]) < \Phi(\boldsymbol{a}_i[k-1])$)
    $dir(x_{i,j}) = x_{i,j}[k] - x_{i,j}[k-1]$

$$dir_{best}(x_{i,j}) = \frac{x_{best,j}[k] - x_{i,j}[k]}{|x_{best,j}[k] - x_{i,j}[k]|}$$

$$dir_{prod}(x_{i,j}) = \alpha_{1,j} \cdot dir(x_{i,j}) + \alpha_{2,j} \cdot |x_{i,j}[k] - x_{i,j}[k-1]| \; dir_{best}(x_{i,j})$$

    $\sigma_{i,j} = \beta_{1,j} \cdot \Phi(\boldsymbol{a}_i[n])$
    $Av_{i,j} = dir_{prod}(x_{i,j})$
    $x_{i,j}[k] = x_{i,j}[k-1] + N(Av_{i,j}, \sigma_{i,j})$
  ELSE
    $dir(x_{i,j}) = 0$

$$dir_{best}(x_{i,j}) = \frac{x_{best,j}[k] - x_{i,j}[k]}{|x_{best,j}[k] - x_{i,j}[k]|}$$

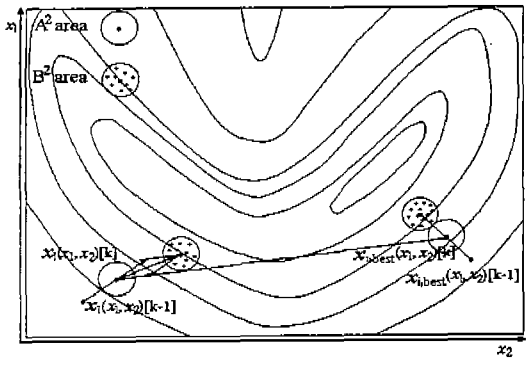


그림 3. 진화프로그래밍들의 탐색 영역 비교(각각의 개체와 세대에 대한 방향성 벡터)

Fig. .3. Comparison with search space of Evolutionary Programming (Direction vectors of each individuals andgeneration)

$$dir_{prod}(x_{i,j}) = \Phi_{best}[k] \cdot dir_{best}(x_{i,j})$$
$$\sigma_{i,j} = \beta_{2,j} \cdot \Phi(a_i[k])$$
$$Av_{i,j} = dir_{prod}(x_{i,j})$$
$$x_{i,j}[k] = x_{i,j}[k-1] + N(Av_{i,j}, \sigma_{i,j})$$

$i = 1, 2, \cdots, \mu$. $j = 1, 2, \cdots, n$.

그림 3은 2차원의 파라메터 영역에 대하여, 앞에서의 설명에서와 같이 진화프로그래밍의 탐색 영역을 표시하였다. $k+1$ 세대에서의 2가지 방향성 벡터를 갖는 최적화 알고리즘의 탐색구간은 $B^2$로 표시하였다. 표준 진화프로그래밍 보다 탐색 영역이 훨씬 빠르게 최적화되는 방향으로 방향성을 가지고 있음을 알 수 있다. 본 장 이후의 제안된 진화프로그래밍은 본 절에서 제시한 각각의 개체와 세대에 대한 방향성 벡터의 합성에 의한 진화프로그래밍을 사용하였다.

## 3. 제안된 진화프로그래밍의 성능비교

제안된 진화프로그래밍과 기존의 진화프로그래밍의 성능비교를 위하여 몇가지 테스트 함수를 사용하였다. 테스트 함수는 최적화 문제의 성능평가를 위해 자주 사용되는 함수로 sum of squares 함수, Bohachevsky 함수 그리고 Rosenbrock 함수를 사용하였다. 다음 식 (4)~(6)은 테스트 함수들을 나타내고, 각 테스트 함수에 사용된 변수 $x_i(i = 1, 2)$의 범위는 $-3 \le x_i \le 3$이다.

Sum of squares function :

$$F(x_1, x_2) = \sum_{i=1}^{2} x_i^2 \tag{4}$$

Bohachevsky function :

$$F(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7 \quad (5)$$

Rosenbrock function :

$$F(x_1, x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 \quad (6)$$

식 (4)는 일반적인 제곱근합(sum of squares) 함수로 최소점이 $x_1 = 0$, $x_2 = 0$인 함수이다. 그리고 식 (5)는 Bohachevsky 함수로 최소점이 $x_1 = 0$, $x_2 = 0$인 최소점을 가지며, 극소점 근처에 여러개의 국부 최소점(local minimum)을 갖는 multimodal 함수이다. 또한 식 (6)은 Rosenbrock 함수로 최소점 $x_1 = 1$, $x_2 = 1$의 근처에 완만한 경사를 갖는 국부 최소값이 없는 unimodal 함수이다. 이러한 함수들은 최적화 방법에서 자주 사용되는 함수로 극소점에 수렴하는 수렴속도의 비교에 사용된다.

진화프로그래밍의 성능비교를 위하여, 진화프로그래밍은 메타 진화프로그래밍을 사용하였다. 그리고 제안된 진화프로그래밍은 2.3절에서 제시한 각각의 개체와 세대에 대한 방향성 벡터의 합성에 의한 진화프로그래밍을 사용하였다. 알고리즘들의 공정한 성능비교를 위하여 초기의 집단은 모두 같이 구성하고 집단의 수는 40개씩으로 고정하였다. 그러므로 $(\mu+\lambda)$ - EP는 $\mu = 40$, $\lambda = 40$이다. 알고리즘의 성능비교는 일정한 100 세대를 지난 후, 두 알고리즘이 찾은 가장 작은 적합도값을 가지는 벡터 $\hat{x}_1$, $\hat{x}_2$와 실제 최소점간의 거리를 Euclidean distance로 구하였다. 성능평가의 공정성을 두기 위해 50번의 반복실험을 통하여 성능을 비교하였다. 표 1은 진화프로그래밍의 성능평가를 위해 사용된 파라메터를 나타내었다.

그림 4-6은 제곱근합 함수로 제안된 진화프로그래밍과 메타 진화프로그래밍의 가장 작은 적합도를 갖

표 1. 진화프로그래밍의 성능평가를 위해 사용된 파라메터
Table 1. Parameters of Evolutionary Programming for the performance valuatio

| 비교 내용 | | 제안된 진화프로그래밍 | 진화프로그래밍 |
|---|---|---|---|
| Type | | $(\mu+\lambda)$ - EP | $(\mu+\lambda)$ - EP |
| $\mu$와 $\lambda$의 수 | | $\mu = 40, \lambda = 40$ | $\mu = 40, \lambda = 40$ |
| 집단 | | 40 | 40 |
| 세대수 | | 100 | 100 |
| 반복횟수 | | 50 | 50 |
| 적합도(minimize) | | 식 (4)-(6) | 식 (4)-(6) |
| 진화프로그래밍의 메타 파라메터 | | · | 6. |
| 제안된 진화프로그래밍의 돌연변이 상수 | $\alpha 1$ | [1, 1] | · |
| | $\alpha 2$ | [1, 1] | |
| | $\beta 1$ | [0.5, 0.5] | |
| | $\beta 2$ | [0.5, 0.5] | |

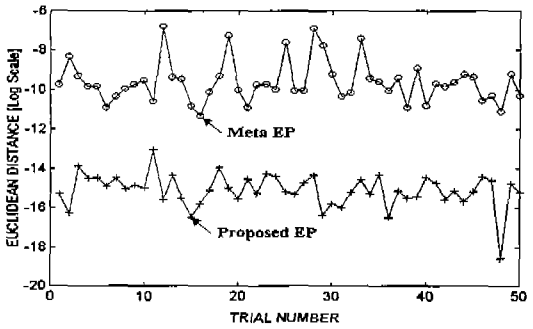


그림 4. 진화프로그래밍의 Euclidean distance (제곱근합 함수)
Fig. 4. Euclidean distance of Evolutionary Programming(sum of squares function)

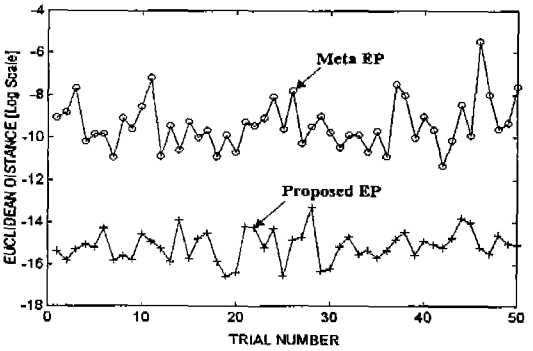


그림 5. 진화프로그래밍의 Euclidean distance(Bohachevsky 함수)
Fig. 5. Euclidean distance of Evolutionary Programming (Bohachevsky function)

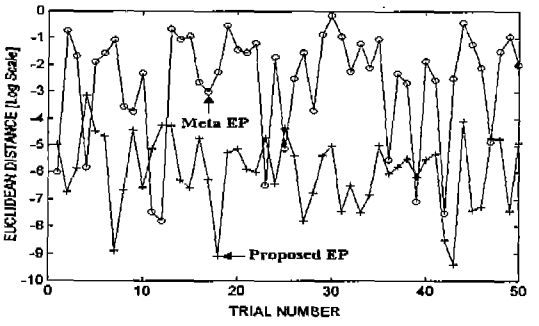


그림 6. 진화프로그래밍의 Euclidean distance(Rosenbrock 함수)
Fig. 6. Euclidean distance of Evolutionary Programming (Rosenbrock function)

는 벡터와 실제 극소점간의 Euclidean distance를 로그(log)를 취한 결과이다. 그림 4의 제곱근합 함수에 대하여 제안된 진화프로그래밍과 메타 진화프로그래밍과의 성능은 $10^{-6}$정도 더 정밀히 수렴함을 보였다. 이는 각 개체가 방향성을 가지고 있으므로, 최적점의 방향으로 빠르게 수렴함을 보인다. 또한 그림 5에서와 같이 국부 최소값이 존재하는 함수에 대해서도 제안된 진화프로그래밍이 메타 진화프로그래밍에 비하여 최적화 성능이 더 뛰어남을 보였다. 그림 6은 최소점 $x_1=1$, $x_2=1$의 근처가 완만한 경사를 갖는 국부 최소값이 존재하지 않는 함수로 앞에서의 다른 테스트 결과와 같이 제안된 진화프로그래밍이 기존의 진화프로그래밍에 비하여 더 정확한 최소값에 수렴함을 보인다. 이는 제안된 진화프로그래밍이 기존의 방법에 비하여 최적화 능력이 뛰어남을 보이며, 빠르게 최적점 근처에 도달함을 알 수 있다.

## 5. 결 론

앞에서의 성능 평가 결과들로 볼 때, 제안된 진화프로그래밍이 기존의 진화프로그래밍에 비하여 빠른 수렴특성과 수렴의 정확성을 보였다. 특히 각 세대의 방향성을 얻는 방법은 각 집단들이 방향성을 가짐으로 인하여 빠른 수렴 속도를 얻을 수 있고, 각 개체에 대한 방향성은 일반적인 진화프로그래밍이나 기존의 방향성을 갖는 진화프로그래밍이 갖는 무성생식만의 특징을 집단들이 방향성을 갖도록 함으로써 각각의 개체는 유성생식의 효과를 가지도록 하여 빠른 수렴 속도를 얻을 수 있었다. 따라서 기존의 진화프로그래밍에 비하여 수렴성능이 좋은 제안된 진화프로그래밍을 사용하여 제어기 파라메터의 자동동조와 같은 최적화 문제에 유용하게 적용될 수 있다.

## 참고문헌


[1] David E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company Inc., 1989.

[2] Zbigniew Michalewicz, Genetic Algorithms + Data Structures= volution Programs, Springer-Verlag, 1994.

[3] Tomas Bäck, Evolutionary Algorithms in Theory and Practice : Evolution Strategied, Evolutionary Programming and Genetic Algorithms, Oxford University Press, 1996.

[4] Kumur Chellapolla, David B. Fogel, "Fitness Distributions in Evolutionary Computation : Analysis of Local Extrema in the continuous Domain," Proceeding of the 1999 Congress on Evolutionary Computation," pp. 1885-1892, 1999.

[5] N. Saravanan and D. B. Fogel, "Evolving Neurocontrollers using Evolutionary Programming," Proceedings of the first IEEE Conference on Evolutionary Computation, pp. 217-222, 1994.

[6] Jaroslaw Srabas, Zbigniew Michalewicz and Jan Mulawka, "GAVaPS-a Genetic Algorithm with Varying Population Size," Proceedings of the first IEEE Conference on Evolutionary Computation, pp. 73-77, 1994.

[7] David B. Fogel, Evolutionary Computation : Toward a New Philosophy of Machine Intelligence, IEEE Press, 1995.

[8] Jong-Hwan Kim, Jeong-Yeol Jeon, Hong-Kook Chae and Kwangil Koh, "A Novel Evolutionary Algorithm with Fast Convergence," International Conference on Evolutionary Computation, pp. 2281-233, 1995.

**박 진 현 (Jin-Hyun Park)**

1992년 : 부산대학교 전기공학과 졸업
1994년 : 부산대학교 대학원 전기공학과 졸업(석사)
1997년 : 부산대학교 대학원 전기공학과 졸업(박사)
1997년~1999년 : 부산정보대학 전기전자계열 전임강사
1999년~현재 : 진주산업대학교 산업자동화공학과 전임강사
주관심분야 : 지능제어, 진화알고리즘, 가변구조제어, 로보틱스

**배 준 경 (Jun-Kyung Bae)**

1982년 : 국민대학교 전자공학과 졸업
1984년 : 경희대학교 전자공학과 졸업(석사)
1990년 : 경희대학교 전자공학과 졸업(박사)
1990년~현재 : 진주산업대학교 산업자동화공학과 부교수
주관심분야 : 로보틱스, 적응제어, 가변구조제어, 퍼지제어