DOI: 10.3745/KIPSTB.2010.17B.2.149

유전자알고리즘의 성능향상을 위한 선택적 돌연변이

정 성 훈†

요 약

유전자알고리즘의 조숙수렴현상(premature convergence phenomenon)은 유전자알고리즘의 성능을 크게 저하시키기 때문에 이 문제를 해결하는 것이 성능향상에 크게 영향을 준다. 본 논문에서는 유전자알고리즘의 조숙수렴현상을 완화하여 성능을 향상시키기 위한 선택적 돌연변이 방법을 제안한다. 선택적 돌연변이에서는 유전자알고리즘 개체의 등급에 따라서 염색체의 특정영역에 비트를 추가적으로 돌연변이 시킨다. 이렇게 함으로서 등급이 낮은 개체는 표현형 상에서 많은 변화가 일어나고 등급이 높은 개체는 작은 변화가 일어나게 된다. 결국 좋은 개체는 그 주변을 세부적으로 탐색하며 좋지 못한 개체는 새로운 영역을 탐색할 기회가 높아지게 되어 조숙수렴현상을 완화하면서 성능향상을 꾀할 수 있게 된다. 성능향상을 측정하기 위하여 4개의 대표적 함수 최적화 문제에 적용해서 제안한 방법의 성능을 측정하였다. 실험결과 기존의 유전자알고리즘보다 성능이 크게 향상됨을 확인하였다.

키워드 : 유전자알고리즘, 조숙수렴현상, 함수최적화, 선택적 돌연변이

Selective Mutation for Performance Improvement of Genetic Algorithms

Sung Hoon Jung†

ABSTRACT

Since the premature convergence phenomenon of genetic algorithms (GAs) degrades the performances of GAs significantly, solving this problem provides a lot of effects to the performances of GAs. In this paper, we propose a selective mutation method in order to improve the performances of GAs by alleviating this phenomenon. In the selective mutation, individuals are additionally mutated at the specific region according to their ranks. From this selective mutation, individuals with low ranks are changed a lot and those with high ranks are changed small in the phenotype. Finally, some good individuals search around them in detail and the other individuals have more chances to search new areas. This results in enhancing the performances of GAs through alleviating of the premature convergence phenomenon. We measured the performances of our method with four typical function optimization problems. It was found from experiments that our proposed method considerably improved the performances of GAs.

Keywords : Genetic Algorithms, Premature Convergence Phenomenon, Function Optimization, Selective Mutation

1. 서 론

유전자알고리즘은 수학최적화기법과 같은 제약사항이 없고 다양한 문제에 다양한 방법으로 응용할 수 있는 장점 때문에 많은 공학최적화문제에 응용되어왔다[1-11]. 유전자알고리즘은 문제를 유전자알고리즘의 염색체로 표현하는 인코딩방법, 유전자알고리즘의 교배(crossover)와 돌연변이 (mutation) 연산방법, 그리고 각종 파라미터에 따라서 성능이 크게 좌우되므로 유전자알고리즘을 성공적으로 적용시키기 위해서는 여러 가지 측면에서 많은 검토가 필요하다[1-5]. 특히 유전자알고리즘의 조숙수렴현상 (premature convergence phenomenon)은 유전자알고리즘의 성능을 크게 저하시키기 때문에 이 문제를 해결하는 것이 성능향상에 크게 영향을 준다[5, 12].

조숙수렴현상이란 초기 세대에 지역 최적해 근방에 생성된 개체들이 다른 개체들보다 상대적으로 우수해 지속적으로 선택되어 자식을 생성함으로서 몇 세대 지나지 않아 대부분의 개체가 해당 지역 최적해 주변에 생성되어 수렴되는 현상을 말한다. 이러한 조숙수렴 현상을 해결하기 위해서는 개체가 지역 최적 해에 빠져드는 것을 방지하여 다양성을 유지하는 것이 중요하다. 다양성을 유지하기 위해서는 교배보다는 돌연변이를 강화해야 하지만 돌연변이 강화는 전역 최적해로의 접근을 어렵게 한다. 그러한 이유로 대부분의 기존에 개발된 방법에서는 다양성을 조정하기 위한 방법으로 돌연변이 확률을 세대별로 조정하는 방법을 사용하였다. 돌연변이 확률을 조정하기 위하여 기 개발된 방법으로는 크게 첫 번째로

※ 본 연구는 2009년도 한성대학교 교내연구비 지원으로 수행되었음.
† 정 회 원 : 한성대학교 정보통신공학과 교수
논문접수 : 2009년 9월 23일
수 정 일 : 1차 2009년 12월 28일
심사완료 : 2010년 1월 1일

초기 세대에 큰 돌연변이 확률로 시작하여 세대가 진행됨에 따라서 수식에 따라서 돌연변이 확률을 줄여주는 방법, 두 번째로 적합도를 이용하여 조숙수렴 정도를 측정하고 이를 이용하여 돌연변이 확률을 조정하는 방법, 마지막으로 다양성 지수를 도입하여 이를 이용하여 돌연변이 확률을 조정하는 방법이 개발되었다. 그러나 첫 번째 방법은 조숙수렴정도를 고려하지 않고 돌연변이 확률을 조정하여 효과가 불확실하다는 점이 문제이며 조숙수렴 정도나 다양성 지수를 도입하여 조정하는 방법은 해당 지수를 어떻게 수식으로 전개하느냐에 따라서 추가적인 파라미터를 설정하는 것이 필요하다는 것과 문제에 따라서 적합도나 인코딩 방법이 다르기 때문에 범용적으로 효과를 얻기 어렵다는 단점이 있다.

본 논문에서는 조숙수렴 정도나 다양성지수를 계산할 필요나 새로운 파라미터를 경험적으로 선택할 필요가 없고 적합도나 인코딩 방법에 상대적으로 의존성이 적은 단순하면서도 강력한 새로운 방법을 제안한다. 본 논문에서 제안하는 방법은 개체의 적합도를 이용하여 등급을 매기고 등급에 따라서 염색체에 특정한 부위에 추가적으로 선택적 돌연변이를 수행하는 방법으로 개체의 다양성을 확장시킨다. 보다 자세히 설명하면 등급이 높은 좋은 개체는 염색체의 하위영역에 선택적으로 돌연변이를 추가하고 등급이 낮은 나쁜 개체는 염색체의 상위영역에 선택적으로 돌연변이를 추가한다. 이렇게 함으로서 높은 적합도를 갖는 개체는 비교적 현재 탐색영역 근처에서 탐색하게하고 낮은 적합도를 갖는 개체는 현재의 탐색 영역에서 먼 곳을 탐색하게 한다. 결국 좋은 개체는 그 주변을 세부적으로 탐색하며 그렇지 못한 개체는 새로운 영역을 탐색할 기회가 높아지게 된다. 그러므로 추가적인 지수를 계산할 필요가 없이 기존 방법의 문제점인 조숙수렴 현상을 완화하여 성능을 향상시킬 수 있게 된다.

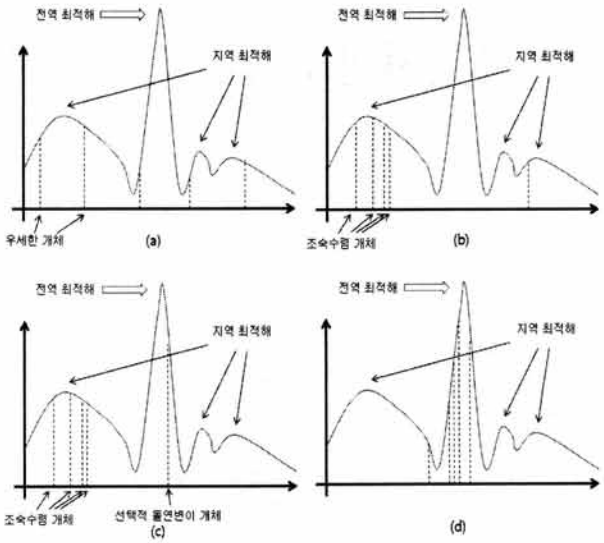
제안한 방법의 성능을 측정하기 위하여 4개의 대표적인 함수최적화 문제를 이용하여 실험하였으며 최초 제안된 유전자 알고리즘과 다양성을 사용한 유전자 알고리즘과 성능을 비교하였다. 실험결과 다양성 사용한 유전자 알고리즘이 가끔 최초 제안된 유전자 알고리즘에 비하여 성능이 좋았으나 그렇게 성능이 많이 향상되지는 않았다. 그러나 본 논문에서 제안한 선택적 돌연변이가 이것을 사용하지 않은 방법에 비하여 2~3배의 성능향상을 가져옴을 보았다. 본 방법은 기존의 어떤 형태의 유전자알고리즘에도 쉽게 추가적으로 적용되어 성능을 향상시킬 수 있는 방법으로서 의미가 있다하겠다.

본 논문은 다음과 같이 구성되었다. 2절은 먼저 조숙수렴 현상과 기존에 제안된 방법에 대하여 자세히 기술한다. 3절에서 선택적 돌연변이를 포함하는 유전자알고리즘에 대하여 설명하며 4절에서는 실험결과와 그 의미를 분석한다. 본 논문은 5절의 결론으로 끝을 맺는다.

2. 조숙수렴현상 및 기존방법

2.1 조숙수렴현상

(그림 1)은 함수 최적화문제에서 조숙수렴현상을 보여준다.



(그림 1) 조숙수렴현상 (a) 초기 세대 (b) 조숙수렴현상에 빠진 상황 (c) 선택적 돌연변이에 의하여 전역 최적해 근처에 개체가 생성 (d) 조숙수렴현상을 빠져나온 상황

(그림 1)의 (a)는 유전자알고리즘의 초기세대의 상황을 보여준다. 초기세대에 개체는 탐색 영역 안에서 무작위적으로 균등하게 생성된다. 만약 어떤 함수 최적화 문제에서 초기개체가 (그림 1)의 (a)처럼 생성되었다고 한다면 지역 최적해에 존재하는 몇몇 개체가 다른 개체에 비하여 우수한 개체가 되고 상대적으로 다음 세대의 부모로 선택될 확률이 높게 된다. 그러한 우수한 개체는 다음 세대의 부모로 선택되어 자신의 주변에 자식 개체를 생성하게 된다. 이런 상황이 몇 세대 지나면 (그림 1)의 (b)와 같이 대부분의 개체가 지역 최적해 근처에 생성되게 되어 더 이상 진화가 어렵게 된다.

이러한 조숙수렴현상은 기존 개체의 염색체를 재조합하는 교배연산으로는 벗어날 수가 없다. 왜냐하면 조숙수렴현상에 빠진 상황에서는 대부분의 개체가 좁은 지역 최적해 영역에 몰려 있기 때문에 유전형으로 매우 유사하며 결국 이를 재조합 해보았자 대부분이 해당 영역 안에 존재하는 개체만 생성할 수 있기 때문이다. 이를 벗어나기 위해서는 돌연변이에 의하여 유전형이 기존의 개체와 매우 다른 것이 생성되어야하며 이 생성된 개체가 기존의 지역 최적해 보다 큰 적합도를 갖아야만 한다. 그러나 돌연변이 확률은 보통 매우 낮기 때문에 (보통 0.05이하로 설정) 돌연변이가 일어나기 힘들며 또한 일어났다고 하더라도 현재 조숙수렴개체의 적합도 보다 낮은 곳에서 일어난 경우 조숙수렴현상을 벗어날 수 없기 때문에 많은 세대 동안에 이 지역 최적해에 머물게 되고 결국 전역 최적해를 찾는데 유전자 알고리즘은 많은 시간을 낭비하게 되어 성능이 급격히 떨어진다. 조숙수렴현상을 완화하기 위하여 돌연변이 확률을 높이게 되면 무작위탐색과 유사하게 동작하여 성능은 더욱 안 좋아 질 수 있다.

2.2 기존방법

조숙수렴현상을 해결하여 성능을 향상시키기 위한 여러

방법들이 고안되었다 [12, 14-15]. 이 방법들의 대부분은 개체의 다양성을 유지함으로서 유전자알고리즘이 초기 개체에 의존한 지역최적화에 빠지지 않게 하는 방법이다. 개체의 다양성을 유지하기 위한 방법은 여러 가지가 가능하다. 예를 들어 하나의 교배 점을 사용하는 것보다 여러 개의 교배 점을 사용하는 것이 자식 개체의 다양성에 도움이 된다. 또한 교배확률을 높이는 것도 다양성에 도움이 된다. 그러나 교배는 염색체의 유전자 자체를 변경하는 것이 아니라 기존에 찾은 염색체의 유전자를 재조합하는 것이기 때문에 다양성 제공에 한계가 있다. 최적화 알고리즘 측면에서 교배와 같이 기존의 찾은 정보를 재조합하는 동작을 개척(exploitation) 이라고 하며 새로운 영역을 찾는 것을 탐험(exploration) 이라고 한다. 교배에 비하여 탐험에 해당하는 돌연변이는 염색체의 유전자를 직접적으로 변경하기 때문에 다양성유지에 상당한 도움이 된다. 다만 돌연변이는 유전자 알고리즘의 핵심인 스키마이론에서 이미 찾은 빌딩 블럭(building block)을 파괴하므로 너무 강력하면 최적해로 접근하는 것을 방해하게 된다. 결론적으로 다양성 유지 측면에서 교배연산이나 교배확률을 조정하는 것보다 돌연변이 연산이나 돌연변이 확률을 조정하는 것이 수월하나 정교하게 조정하지 못하면 오히려 성능을 떨어뜨릴 우려가 있다.

돌연변이는 동작이 단순하기 때문에 염색체의 다양성을 유지하기 위하여 새로운 돌연변이 연산을 도입하는 것보다 돌연변이 확률을 조정하는 방법이 많이 제안되었다[13, 16]. 돌연변이 확률을 조정하는 비교적 단순한 방법으로 조숙수렴이 발생할 수 있는 초기에는 돌연변이 확률을 높게 주고 세대가 진행됨에 따라서 전역 최적해에 접근하게 하기 위하여 이 값을 낮추는 방법 방법을 생각할 수 있다. 논문 [16]에서는 다음 수식 (1)과 같은 방법을 제안하였다.

$$p_m = 0.1 - 0.09 \times \frac{g}{G} \tag{1}$$

수식 (1)에서 보듯이 초기 돌연변이 확률은 $g=0$이므로 0.1 이며 세대 g에 따라서 점점 줄어들다가 G 세대에는 0.01 이 된다. 그러나 조숙수렴이 나타나는 세대는 적용문제나 초기 개체의 분포에 따라서 다르기 때문에 적당한 G 값을 찾는 것도 어려우며 더군다나 개체의 진화정도를 고려하지 않고 단순히 세대수에 따른 돌연변이 확률 조정은 좋은 성능을 기대하기 어렵다.

이러한 문제를 보완하고자 도입된 방법은 개체의 조숙수렴정도를 고려하여 돌연변이 확률을 조정하는 방법이다. 이전에도 언급한 것처럼 조숙수렴이라는 것이 대부분의 염색체가 지역 최적해로 수렴하여 다양성을 상실한 상황을 말하는 것이므로 조숙수렴 정도를 파악하는 척도로 개체의 유사성을 사용할 수 있다. 논문 [13]에서는 조숙수렴 척도로 염색체의 적합도, 평균적합도, 최대 적합도를 이용하여 식 (2)와 같이 돌연변이 확률을 조정하였다.

$$\begin{aligned} p_m &= k2\frac{(f_{\max}-f)}{(f_{\max}-\bar{f})}, && f \geq \bar{f} \\ p_m &= k4, && f < \bar{f} \end{aligned} \tag{2}$$

식(2)에서 k_2와 k_4는 상수로서 사용자가 선택하며 $f_{\max}$는 최대 적합도, f는 해당 염색체의 적합도, $\bar{f}$는 평균 적합도를 의미한다. 먼저 분모 항을 살펴보면 조숙수렴현상이 나타나면 염색체가 대부분 유사해지기 때문에 $\bar{f}$와 $f_{\max}$의 차이가 작아지며 결국 p_m은 커진다. 그러나 좋은 개체에도 높은 p_m을 적용하면 좋은 빌딩블럭이 파괴되어 전역 최적해로의 접근을 방해한다. 이를 위하여 분자 항 $f_{\max}-f$를 적용하여 좋은 개체일 경우에는 낮은 돌연변이 확률을 적용하게 하였다. 적합도 f가 $\bar{f}$인 경우에는 k_2 값을 갖게 되며 평균적합도 보다 낮은 적합도를 갖는 개체는 좀 더 강력하게 돌연변이를 할 수 있게 하기 위하여 일정한 돌연변이 확률 k_4로 설정한다. 그러나 이 방법은 k_2 와 k_4와 같이 새로운 파라미터가 발생해 이를 경험적으로 선택해 주어야하며 이 값이 제대로 주어지지 않으면 성능향상에 문제가 발생한다는 것이다. 또한 k_2와 k_4 같은 변수는 함수에 따라서 최적 값이 다를 것이기 때문에 새로운 문제가 발생한다.

또 다른 방법으로는 돌연변이 확률을 염색체에 포함시켜 염색체와 함께 진화시키는 방법이다. 이러한 방법은 공진화(co-evolution) 혹은 자가-적응(Self-adaptive) 방법이라고 불리는데 [14] 적절한 돌연변이 확률을 갖는 개체는 적합도가 높은 자손을 생성할 것이며 적합도가 높은 자손은 살아남아 퍼질 것이라는 사실에 기반 한다. 여러 가지 방법으로 염색체에 표현된 돌연변이 확률도 같이 진화시킴으로 공진화 방법으로 불리기도 한다. 공진화 방법은 돌연변이 확률을 조정하는 방법에 비하여 너무 복잡하다는 것과 돌연변이 확률이 진화되지 않는 경우에 대처방법이 없다는 단점이 있다.

가장 최근의 방법은 개체들 간의 다양성을 직접 계산하여 이를 사용하는 방법이다 [17]. 먼저 두 개체가 차이가 많이 나는지를 계산하기 위하여 개체 i와 개체 j 사이의 거리 d는 수식 (3)처럼 계산된다.

$$d(i,j) = \sqrt{(g_i(1)-g_j(1))^2 + \cdots + (g_i(N)-g_j(N))^2} \tag{3}$$

수식 (3)에서 g는 개체의 유전자이다. 만약 유전자가 0과 1의 스트링으로 구현되었다면 개체 i와 개체 j의 거리는 해밍거리로 규정될 수 있다. 이 거리를 이용하여 다양성 척도를 다음과 같이 정의한다. 수식(4)에서 div는 다양성, D는 미리 정의된 문턱치, N은 개체수를 나타낸다. 두 개체 사이의 거리가 미리 정의된 문턱치 D보다 크면 1을 더해준다. 만약에 모든 개체가 거리 D이하로 유사하면 $\mathrm{div}=0$ 이 된다. 모든 개체가 거리 D 이상이면 1보다 작은 실수가 된다.

$$\mathrm{div} = \frac{\sum_{i=1}^{N}\sum_{j=i+1}^{N} 1_{d(i,j)>D}}{N^2} \tag{4}$$

이 값을 이용하여 돌연변이 확률을 조정하는 것은 수식 (5)와 같다. 결국 다양성 지수 div가 작으면 작아질수록 p_m은 커지며 div가 커지면 커질수록 p_m은 작아지게 된다. 이

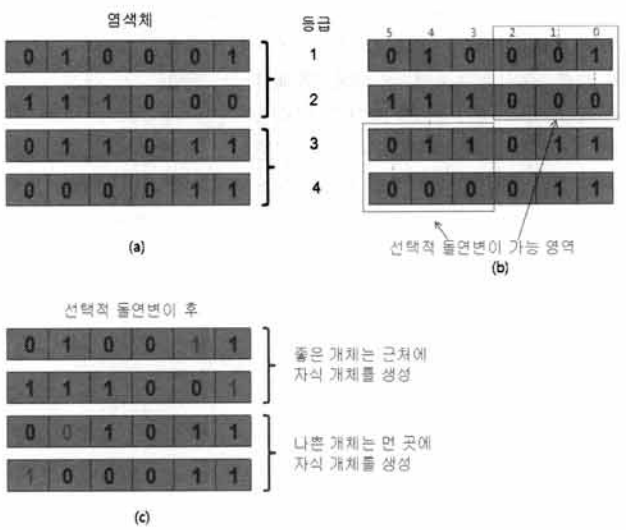
방법은 일일이 개체별 비교를 통하여 다양성을 측정하기 때문에 계산상 많은 시간이 소요된다는 단점이 있다. 특히 개체 수가 클 경우 $O(n^2)$ 복잡도로 다양성을 계산한다는 것은 장점보다는 단점이 더 클 것으로 판단된다.

$$p_m = p_m^0 - div \cdot p_m^0 \tag{5}$$

본 논문에서 제안하는 선택적 돌연변이는 (그림 1)의 (c)에서 보듯이 비교적 높은 적합도를 갖는 좋은 개체는 현재의 표현형의 위치에서 작게 변화하게 하고 낮은 적합도를 갖는 나쁜 개체는 현재의 표현형의 위치에서 많게 변화하게 하는 추가적인 돌연변이를 수행한다. 이렇게 함으로서 새로운 지수를 도입하지 않고 단지 개체의 등급에 따라서 선택적으로 추가적으로 돌연변이를 수행해주는 단순한 방법으로 좋은 개체의 좋은 유전자를 파괴하는 것을 줄이면서도 개체의 다양성을 유도하는 장점이 있다. 추가적인 선택적 돌연변이에 의하여 나쁜 개체들은 더 넓은 영역을 탐색하게 되고 전역 최적해 근처로 돌연변이를 일으킬 확률이 커지게 된다. 일단 전역 최적해 근처로 돌연변이가 일어나면 이 개체는 기존의 조숙수렴개체보다 적합도가 월등하기 때문에 부모로 선택되고 많은 자식을 생성하게 되어 결국 (그림 1)의 (d)처럼 조숙수렴현상을 벗어나 전역 최적해를 찾게 된다.

3. 선택적 돌연변이

선택적 돌연변이의 목적은 2절에서 언급한 것처럼 좋은 개체는 비교적 현재 위치 근처를 나쁜 개체는 먼 곳을 탐색하게 하기위한 것으로 기존의 돌연변이 연산에 추가적으로 실행된다. 먼저 좋은 개체인지 나쁜 개체인지를 판단하기위하여 개체에 등급을 매긴다. 그리고 등급에 따라서 선택적 돌연변이를 할 비트영역을 결정하고 해당 비트 영역내의 비트를 무작위적으로 선택하여 돌연변이 시킨다. 돌연변이는 해당 비트가 1이면 0으로 0이면 1로 만든다. 예를 들어 설명하면 (그림 2)와 같다. 유전자알고리즘의 특정 세대에서의 상황이 (그림 2)의 (a)와 같다고 하자. 각 개체는 평가되어 적합도를 갖게 되며 이 적합도를 이용하여 등급을 매겼을 때 그림과 같다고 하자. 만약 등급을 두 그룹으로 나눈다면 등급 1과 2가 좋은 개체 그룹이 되며 등급 3과 4가 나쁜 개체 그룹이 된다. 개체를 두 그룹으로 나누었기 때문에 염색체의 비트열도 두 영역으로 나눈다. 그림에서는 염색체가 6비트로 구성되어 있기 때문에 5, 4, 3 비트는 상위비트영역을 2, 1, 0 비트는 하위비트영역을 형성한다. 좋은 개체 그룹의 경우에는 현재 위치 주변을 탐색하게 하기 위하여 하위비트영역이 선택적 돌연변이 가능 영역이 되며 나쁜 개체 그룹의 경우에는 현재 위치에서 먼 곳을 탐색하게 하기 위하여 상위비트 영역이 선택적 돌연변이 가능영역이 된다. 일단 선택적 돌연변이 가능영역이 설정되면 해당 영역 안에서 무작위적으로 비트를 선택해 돌연변이 시킨다. 그림 2의 (b)에 화살표로 표시된 비트가 선택적 돌연변이 비트로 선



(그림 2) 선택적 돌연변이 (a) 돌연변이 전 상황 (b) 돌연변이 가능 영역 (c) 돌연변이 후 상황

택되었다고 한다면 결국 해당 비트에 돌연변이가 일어나 그림 2의 (c)와 같이 돌연변이가 일어난다. (그림 2)의 (c)에서 보듯이 좋은 개체의 경우 하위비트가 돌연변이 되어 비교적 부모 개체 근처에 자식 개체가 생성되며 나쁜 개체의 경우는 부모 개체에서 먼 곳에 생성된다.

함수 최적화의 경우는 대부분 탐색 영역을 0과 1의 이진수로 염색체를 표현하기 때문에 위와 같이 비트 영역에 따라서 표현형의 변위차가 크거나 작게 되며 우리가 원하는 선택적 돌연변이 효과를 얻을 수 있다. 그러나 함수최적화 문제가 아니거나 함수최적화 문제임에도 가중치를 갖는 이진수로 코딩하지 않는 경우에는 위와 같이 상위 하위 영역으로 나누어서 해당 효과를 얻기 어렵다. 이런 경우라도 해당 염색체 코딩 방법에 따라서 표현형의 변위차를 크거나 작게 제어할 수 있는 문제라면 똑 같은 원리로 적용하면 된다. 다만 외판원 문제 (traveling salesman problem) 같은 조합최적화 문제의 경우에는 영역별로 가중치를 줄 수 없기 때문에 위와 같은 방법을 적용할 수 없다. 이런 경우에는 영역별 선택적 돌연변이 방법이 아니라 등급별 돌연변이 방법으로 해결할 수 있을 것으로 생각된다. 즉 좋은 등급의 개체는 돌연변이 확률을 낮게 주어 돌연변이가 조금 일어나게 하고 나쁜 등급의 개체는 돌연변이 확률을 높게 주어 돌연변이가 많이 일어나게 하는 방법이다.

선택적 돌연변이를 포함하여 유전자알고리즘을 기술하면 Algorithm 1과 같다.

[Algorithm 1] Genetic algorithm with selective mutation

// t : time

// n : population size //

// s : string length //

// N : the number of grades //

// $P^r_{1,\ldots,N}$: the part of rank //

// $P^s_{1,\ldots,N}$: the part of string //

```
    // r_i : the rank of ith individual //
    // P : populations //
1   t ← 0
2   initialize P(t)
3   evaluate P(t)
4   while (not termination-condition)
5   do
6     t ← t + 1
7     select P(t) from P(t−1)
8     recombine P(t)
9       do crossover
10      do mutation
11    evaluate P(t)
12    do selective mutation (▼)
13      sort P(t) with fitness and rank individuals
14      divide the strings into N parts, P^s
15      divide the rank into N parts, P^r
16      for i=1 to n
17        if r_i∈P_j^r
18            do mutation with a randomly selected bit
              in string part, P^s_{N−j+1}
19        end if
20      end for
21    evaluate P(t)  (▲)
22  end
```

Algorithm 1에서 다른 부분은 기존의 유전자알고리즘과 동일하다. 추가적인 선택적 돌연변이는 12줄에서 21줄까지이다. 위의 예에서 설명한 것처럼 개체군을 적합도에 따라서 등급을 매기고 그룹 수 N에 따라서 개체를 N그룹으로 나누고 염색체 스트링도 동일한 N 영역으로 나눈다. 이후 각 염색체의 선택적 돌연변이 가능영역에서 비트를 무작위적으로 선택해 돌연변이를 수행한다. 이 작업이 끝난 후에는 염색체가 변경되어 있을 것이므로 다시금 evaluate $P(t)$를 수행하여 적합도를 재 산출해야한다. 이렇게 해야 7번째 줄의 다음 세대 부모 선택 시 선택적 돌연변이를 한 결과가 반영된다. 그렇지 않으면 선택적 돌연변이에 의하여 획기적으로 좋아진 개체가 예전의 적합도를 갖게 되어 다음 세대의 부모 개체로 선택되지 않아 선택적 돌연변이의 효과를 보지 못한다.

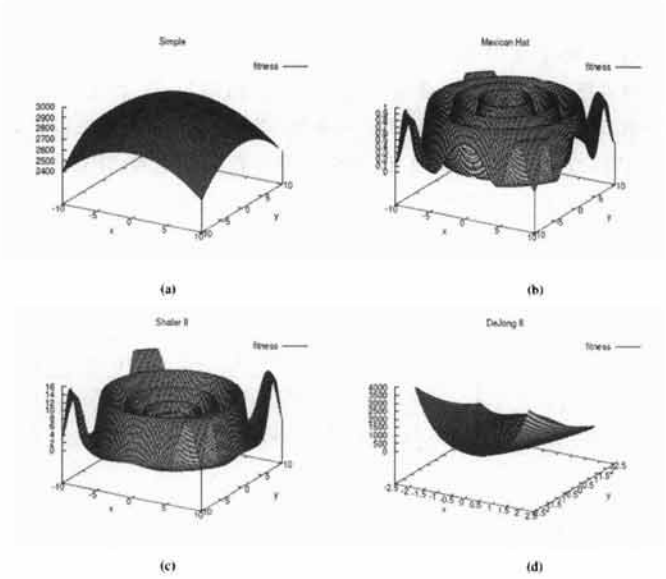
4. 실험 결과

본 논문에서 제안한 선택적 돌연변이가 기존의 유전자알고리즘의 성능향상에 얼마나 기여하는지 알아보기 위하여 4개의 함수최적화 문제를 이용하여 실험하였다. 다음은 4개의 함수 최적화 문제를 기술한 수식이다.

$$
\begin{aligned}
f_1 &= 3000 - 3(x^2+y^2) \\
f_2 &= 0.5 - \frac{\sin(\sqrt{x^2+y^2})\sin(\sqrt{x^2+y^2}) - 0.5}{(1.0+0.001(x^2+y^2))(1.0+0.001(x^2+y^2))} \\
f_3 &= (x^2+y^2)^{0.25}\sin(50(x^2+y^2)^{0.1}+1)^2 \\
f_4 &= 100(x^2-y)^2+(1-x)^2
\end{aligned}
\tag{6}
$$

f_1은 단순(simple) 함수이다. f_2는 멕시코 모자 (Mexican hat) 함수이며 f_3는 Shafer II 함수이고 f_4는 DeJong II 함수이다. 이러한 함수들은 함수최적화 문제로 많이 사용되어 온 함수들이다[12]. 4개의 함수를 그림으로 그리면 (그림 3)과 같다.

(그림 3)에서 보듯이 함수 f_1은 $(0,0)$에 전역 최적해를 갖는 지역최적해가 없는 단순한 함수이다. 함수 f_2는 $(0,0)$에 전역 최적해를 갖으며 그 주변에 많은 지역 최적해를 갖는 함수이다. 이러한 지역 최적해는 유전자 알고리즘이 전역 최적해를 찾아가는데 조숙수렴현상을 발생시켜 성능을 저하시킨다. 함수 f_3은 $(-10,-10)$, $(-10,10)$, $(10,-10)$, $(10,10)$에 대략 14.3의 네 개의 전역 최적해를 갖는 함수이다. 함수 f_4는 $(-2.048,-2.048)$ 영역에 하나의 전역 최적해를 갖는 함수이다. 실험을 위하여 사용한 유전자 알고리즘의 파라미터는 <표 1>과 같다.



(그림 3) 4개의 함수최적화 문제 (a) simple 함수 (b) Mexican hat 함수 (c) Shafer II 함수 (d) DeJong II 함수

〈표 1〉 유전자알고리즘 파라미터 설정

파라미터	값
부모 개체 선택 방법	룰렛 휠 선택
교배 확률 (p_c)	0.6
돌연변이 확률 (p_m)	0.05
개체 수 (n_i)	12, 48, 96
염색체 길이 (l)	8, 10, 12 비트
선택적 돌연변이 수 (n_s)	1, 2, 3비트
실험 반복 수	100

<표 1>에서 보듯이 대부분의 파라미터는 일반적으로 사용하는 파라미터 값을 설정하였다. 개체 수는 12, 48, 96으로 선택하였다. 염색체 길이는 x, y 각 축으로 8, 10, 12 비트로 설정하였다. 그러므로 전체 탐색공간은 $2^{16}=65,536$, $2^{20}=1,048,576$, $2^{24}=16,777,216$이다. 선택적 돌연변이 수는 선택적 돌연변이 영역에 몇 번의 돌연변이를 발생시킬 것인가를 나타낸다. 2인 경우 두 번의 선택적 돌연변이를 시도하는데 만약 무작위적으로 생성된 비트 위치가 다르면 모두 각 위치에 선택적 돌연변이가 일어나며 같으면 선택적 돌연변이가 일어난 곳에 또 선택적 돌연변이가 일어나 원래의 값이 된다. 등급은 염색체 길이가 비교적 짧아서 두 그룹으로 나누어 적용하였다. 즉 개체를 상위그룹과 하위그룹으로 나누고 그룹에 따라서 염색체를 두 영역으로 나누어 선택적 돌연변이를 수행하였다. 일반적으로 유전자알고리즘은 초기 개체에 따라서 최적 값을 찾는 시간이 달라진다. 그러므로 모든 실험 결과는 100번을 실험한 결과를 평균하여 나타내었다. 표준편차도 구했지만 간략하게 표현하기 위하여 생략하였다. 실험은 유전자 알고리즘이 전역 최적해를 찾은 경우 해당 세대수를 기록하는 방식으로 수행되었다. 그러므로 짧은 세대에 전역 최적해를 찾는 유전자 알고리즘이 성능이 좋은 유전자 알고리즘이다.

<표 2>는 염색체길이가 12비트(x, y축으로는 24비트)이며 선택적 돌연변이 수가 1 비트일 때 개체 수별로 실험한 결과를 보여준다. 표에서 OGA, DGA, RGA는 각각 초기 유전자알고리즘 [1], 다양성을 사용한 적용한 유전자 알고리즘 [17], 그리고 본 논문에서 제안한 등급중심 유전자 알고리즘을 나타낸다. 다양성을 사용한 방법에서 다양성을 측정하는데 사용하는 미리 정의된 문턱치 D 는 개체수의 1/3을 사용했다. 다양성을 사용한 방법에서는 돌연변이 확률 이외에 교배확률도 조정하였지만 세 방법의 공정한 비교를 위하여 본 논문에서는 돌연변이 확률만 조정하여 실험하였다.

표 2에서 보듯이 제안한 방법이 기존의 방법보다 대략 2배에서 3배 정도의 성능향상을 보였다. 이러한 경향은 개체수가 증가 하더라도 거의 유지되었다. 그러므로 본 논문에서 제안한 방법은 개체 수에 상관없이 일정하게 성능을 향상시킴을 보았다. 다양성을 이용한 유전자 알고리즘 DGA는 때때로 OGA보다 성능이 좋았지만 우리가 제안한 방법 RGA보다는 성능이 좋지 않았다. DGA는 특히 함수 f_4에서 좋지 않았는데, 이는 (그림 3)에서 보듯이 함수 f_4가 영역이 두 지역최적화 영역으로 크게 나뉘기 때문에 강한 돌연변이 확률이 아니고서는 전역최적화 영역으로 가기 힘들기 때문으로 보인다. 다음으로 개체수가 12개이고 선택적 돌연변이 수가 1비트 일 때 염색체 길이에 따른 성능을 실험해 보았다.

<표 3>에 나타나 있듯이 염색체 길이가 변해도 성능향상 정도는 거의 유사하게 유지 되었다. 특히 염색체 길이가 작은 경우에 대부분의 결과에서 3배 정도의 성능향상을 보였다. DGA는 OGA 보다 성능이 좋은 적도 있지만 RGA보다는 좋지 못했다. 마지막으로 개체수가 12개이고 염색체길이가 12개 일 때 선택적 돌연변이 수에 따른 성능을 측정하여 보았다. DGA 는 f_1을 제외하고 OGA보다 성능이 좋았다. 그러나 RGA의 성능이 가장 좋았다.

등급의 그룹을 두 그룹을 사용했기 때문에 x와 y축별로 12비트는 6비트씩 두 영역으로 나뉜다. 개체 수가 12개이기

〈표 2〉 개체 수별 실험 결과 ($l=12$, $n_s=1$)

개체 수 (n_i)	12		
GA	OGA	DGA	RGA
f_1	288290.3	353886.6	164276.3
f_2	254684.6	209091.7	129838.2
f_3	1857.6	1851.0	837.9
f_4	28357.0	27734.5	8039.6
개체 수 (n_i)	48		
GA	OGA	DGA	RGA
f_1	74839.7	67225.9	38715.2
f_2	73157.2	67343.6	31679.5
f_3	286.8	250.5	163.6
f_4	16538.9	59337.9	5904.0
개체 수 (n_i)	96		
GA	OGA	DGA	RGA
f_1	35315.7	27557.5	17582.3
f_2	38187.6	35646.9	13884.6
f_3	148.1	139.1	79.3
f_4	38328.8	356575.6	12419.3

〈표 3〉 염색체 길이별 실험 결과 ($n_i=12$, $n_s=1$)

염색체길이 (l)	8		
GA	OGA	DGA	RGA
f_1	1742.7	1557.6	578.7
f_2	1851.5	2172.7	668.9
f_3	194.1	189.6	70.4
f_4	2338.1	3871.0	392.9
염색체길이 (l)	10		
GA	OGA	DGA	RGA
f_1	28018.3	21343.9	7921.1
f_2	19625.1	22896.5	8327.9
f_3	1763.8	1429.6	678.6
f_4	3485.5	4724.2	941.0
염색체길이 (l)	12		
GA	OGA	DGA	RGA
f_1	288290.3	353886.6	164276.3
f_2	254684.6	209091.7	129838.2
f_3	1857.6	1851.0	839.2
f_4	28357.0	27734.5	8039.6

〈표 4〉 선택적 돌연변이 수별 실험 결과 (n_i=12, l=12)

선택적 돌연변이 수 (n_s)	-		1	2	3
GA	OGA	DGA	RGA		
f_1	288290.3	353886.6	164276.3	122758.7	111168.7
f_2	254684.6	209091.7	129838.2	133583.4	115564.4
f_3	1857.6	1851.0	839.2	860.6	1053.4
f_4	28357.0	27734.5	8039.6	6244.5	4598.2

때문에 1등부터 6등까지의 상위그룹은 12비트 중 하위6비트 즉 5, 4, 3, 2, 1, 0 비트를 선택적 돌연변이하며 7등부터 12등까지는 상위 6비트 즉 11, 10, 9, 8, 7, 6 비트를 선택적 돌연변이 한다. 돌연변이 개수에 따라서 이 영역 내에서 무작위적으로 선택하여 돌연변이를 한다. f_3를 제외하고는 돌연변이 수가 증가함에 따라서 성능이 증가하는 것을 볼 수 있다. f_3에서만 이런 현상이 나타나는 이유는 f_1, f_2, f_4의 최적해가 한 곳에 있는데 비하여 f_3에서는 지역적으로 떨어진 4곳에 분산하여 존재하기 때문에 나타나는 현상으로 보인다. 선택적 돌연변이 수가 커질수록 좋은 개체의 하위비트가 많이 변하여 개체의 위치가 많이 이동하게 된다. f_3의 경우 지역적으로 떨어져 있는 4개의 전역 최적해로 개체가 분산이 될 가능성이 높으며 4개의 모서리 근방에 있던 좋은 개체들이 많이 이동하여 가운데 영역에 지역 최적해 쪽으로 빠지면 다시 전역 최적해 쪽으로 가기가 힘들어진다. 반면 다른 함수에서는 전역 최적해가 한 곳에 있기 때문에 개체가 몰려있어 이런 현상이 많이 완화될 것으로 판단된다.

5. 결 론

본 논문에서는 유전자알고리즘의 조숙수렴현상을 완화하여 유전자알고리즘의 성능을 향상시키는 선택적 돌연변이 방법을 제안하였다. 선택적 돌연변이는 기존의 돌연변이에 추가하여 실행되는 것으로 좋은 개체는 해당 개체 주변을 탐색하게하고 나쁜 개체는 먼 곳을 탐색하게 함으로서 조숙수렴현상을 쉽게 벗어나면서도 전역 최적해에 접근하게 한다. 좋은 개체와 나쁜 개체를 구분하기 위하여 개체에 등급을 매기고 등급에 따라서 두 그룹으로 나누었다. 좋은 개체들은 하위비트 영역을 선택적으로 돌연변이 하게하고 나쁜 개체들은 상위비트 영역을 선택적 돌연변이 하게 하였다. 4개의 함수최적화 문제에 적용하여 알고리즘의 성능을 측정하였다. 실험결과 선택적 돌연변이를 추가한 유전자알고리즘의 성능이 기존의 방법보다 최대 3배까지 좋아짐을 관찰할 수 있었다. 다양성을 이용하여 돌연변이 확률을 조정하는 방법도 같이 실험하였다. 다양성을 이용한 방법은 때때로 기존 방법보다 성능이 약간 좋아지는 것을 볼 수 있었으나 대부분의 실험에서 우리가 제안한 방법보다는 성능이 좋지 못했다. 이런 결과로 볼 때 본 논문에서 제안한 선택적 돌연변이가 유전자 알고리즘 성능 향상에 기여함을 확인 하였다. 본 논문에서 제안한 방법은 기존 유전자알고리즘의 틀을 유지하고 추가적으로 실행하는 것이기 때문에 쉽게 다른 유전자 알고리즘에 적용될 수 있는 장점이 있다.

참 고 문 헌

[1] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley, 1989.

[2] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: A Survey," *IEEE Computer Magazine*, pp.17-26, June, 1994.

[3] J. L. R. Filho and P. C. Treleaven, "Genetic-Algorithm Programming Environments," *IEEE Computer Magazine*, pp.28-43, June, 1994.

[4] D. Beasley, D. R. Bull, and R. R. Martin, "An Overview of Genetic Algorithms: Part 1, Fundamentals," Technical Report obtained from http://home.ifi.uio.no/~jimtoer/GA_Overview1.pdf.

[5] D. B. Fogel, "An Introduction to Simulated Evolutionary Optimization," *IEEE Transactions on Neural Networks*, Vol.5, pp.3-14, Jan., 1994.

[6] H. Szczerbicka and M. Becker, "Genetic Algorithms: A Tool for Modelling, Simulation, and Optimization of Complex Systems," *Cybernetics and Systems: An International Journal*, Vol.29, pp.639-659, Aug., 1998.

[7] R. Yang and I. Douglas, "Simple Genetic Algorithm with Local Tuning: Efficient Global Optimizing Technique," *Journal of Optimization Theory and Applications*, Vol.98, pp.449-465, Aug., 1998.

[8] C. Xudong, Q. Jingen, N. Guangzheng, Y. Shiyou, and Z. Mingliu, "An Improved Genetic Algorithm for Global Optimization of Electromagnetic Problems," *IEEE Transactions on Magnetics*, Vol.37, pp.3579-3583, Sept., 2001.

[9] J. A. Vasconcelos, J. A. Ramirez, R. H. C. Takahashi, and R. R. Saldanha, "Improvements in Genetic Algorithms," *IEEE Transactions on Magnetics*, Vol.37, pp.3414-3417, Sept., 2001.

[10] E. Alba and B. Dorronsoro, "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms," *IEEE Transactions on Evolutionary Computation*, Vol.9, pp.126-142, Apr., 2005.

[11] V. K. Koumousis and C. Katsaras, "A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance," *IEEE Transactions on Evolutionary Computation*, Vol.10, pp.19-28, Feb., 2006.

[12] J. Andre, P. Siarry, and T. Dognon, "An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization," *Advances in engineering software*, Vol.32, No.1, pp.49-60, 2001.

[13] M. Srinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, Vol.24, No.4, pp.656-667, Apr., 1994.

[14] A. E. Eiben, Z. Michalewicz, m. Schoenauer, and J. E. Smith "Parameter Control in Evolutionary Algorithms," *Studies in Computational Intelligence*, Vol.54, pp.19-46, 2007.

[15] Silja Meyer-Nieberg and Hans-Georg Beyer, "Self-Adaptation in Evolutionary Algorithms," *Studies in Computational Intelligence*, Vol.54, pp.47-75, 2007.

[16] C. W. Ho, K. H. Lee, and K. S. Leung, "A Genetic Algorithm Based on Mutation and Crossover with Adaptive Probabilities," *Proceedings of the 1999 Congress on Evolutionary Computation*, Vol.1, pp.768-775, 1999.

[17] Zhihua Tang, Youtuan Zhu, Guo Wei, and Jinkang Zhu, "An Elitist Selection Adaptive Genetic Algorithm for Resource Allocation in Multiuser Packet-based OFDM Systems," *Journal of Communications*, Vol. 3, No. 3, pp.27-32, July 2008.

정 성 훈

e-mail : shjung@hansung.ac.kr

1988년 한양대학교 전자공학과(공학사)

1991년 한국과학기술원 전기및전자공학과 (공학석사)

1995년 한국과학기술원 전기및전자공학과 (공학박사)

1995년~1996년 한국과학기술원 전기및전자공학과 위촉연구원

1996년~현 재 한성대학교 정보통신공학과 교수

관심분야 : 진화연산, 신경망, 퍼지, 지능시스템, 시스템생물학, 뇌공학