

# 유전자 알고리즘의 성능향상을 위한 비례-적분-미분 평가방법

## Proportional-Integral-Derivative Evaluation for Enhancing Performance of Genetic Algorithms

정성훈

Sung Hoon Jung

한성대학교 정보공학부

### 요 약

본 논문에서는 유전자 알고리즘의 성능향상을 위한 비례-적분-미분 평가방법을 제안한다. 비례-적분-미분 평가방법에서는 평가함수에 의하여 계산된 적합도와 더불어 각 개체의 부모 적합도, 초기세대로 부터 이전세대까지의 최소, 최대 적합도를 이용하여 평가함으로서 유전자 알고리즘의 성능저하를 가져오는 조속수렴 (premature convergence) 확률을 줄여주어 결과적으로 유전자 알고리즘의 성능을 향상시키게 된다. 비례-적분-미분 평가방법의 성능을 보이기 위하여 유전자 알고리즘 성능 검증에 많이 사용되어온 대표적인 함수 최적화 문제들을 적용하여 실험해본 결과 제안한 방법이 유전자 알고리즘의 성능을 크게 향상 시킬수 있음을 확인하였다. 제안한 평가방법은 다른 형태의 유전자 알고리즘의 성능향상을 위해서도 쉽게 적용될수 있다.

### Abstract

This paper proposes a proportional-integral-derivative (PID) evaluation method for enhancing performance of genetic algorithms. In PID evaluation, the fitness of individuals is evaluated by not only the fitness derived from an evaluation function, but also the parents fitness of each individual and the minimum and maximum fitness from initial generation to previous generation. This evaluation decreases the probability that the genetic algorithms fall into a premature convergence phenomenon and results in enhancing the performance of genetic algorithms. We experimented our evaluation method with typical numerical function optimization problems. It was found from extensive experiments that our evaluation method can increase the performance of genetic algorithms greatly. This evaluation method can be easily applied to the other types of genetic algorithms for improving their performance.

**Key words** : 유전자 알고리즘, 비례-적분-미분 평가, 최적화

### 1. 서 론

유전자 알고리즘은 적자생존의 원리를 적용한 탐색 알고리즘으로서 빠르게 최적해에 근접하고 탐색공간이 매우 큰 문제에 유용해 많은 분야에 응용되어 왔다 [1-5]. 그러나, Goldberg [1] 가 제안한 간단 유전자 알고리즘 (SGA: Simple Genetic Algorithm)은 몇 가지 해결해야할 문제를 갖고 있는데 그 중 성능에 큰 영향을 미치는 것으로 몇 세대 진화 후에 대부분의 개체가 초기 개체중 좋은 적합도를 갖는 개체쪽으로 수렴하는 조속수렴 (premature convergence) 문제를 안고 있다 [6,7]. 이러한 문제를 완화 시키고 성능을 향상시키고자 변형된 연산자를 사용하는 방법 [6,8-12] 연산자 확률을 동적으로 바꾸는 방법 [9,13-17] 또는 다른 방법과 결합하는 방법 [18]등이 연구 되어 왔다.

우리는 유전자 알고리즘에대한 분석을 통하여 조속수렴

현상이 나타나는 주된 이유는 다음세대의 부모 개체를 위한 선택동작에서 비록 룰렛휠과 같이 적합도에 비례한 확률적인 선택을 하더라도 적합도가 좋은 개체가 선택될 확률의 높음으로 인하여 발생함을 파악하였다. 즉, 개체를 진화하게하는 조건 (좋은 적합도를 갖는 개체를 다음 세대의 부모개체로 선택하는 조건)이 또한 개체를 조속수렴하게 하는 조건도 됨을 의미한다. 이러한 현상은 기존의 유전자 알고리즘이 고정된 평가함수에 의하여 계산된 고정된 적합도를 사용함으로 인하여 해당 개체의 적합도 변화추이 및 개체들의 분포에 따른 상대적인 적합성등을 평가하지 못하는데 원인이 있음을 분석하였다. 결국 기존의 평가방법은 룰렛휠 방법등과 같이 확률적인 선택을 하더라도 주로 좋은 적합도를 갖는 개체들이 다음세대의 부모개체로 선택되게 함으로서 최종해가 초기 개체에 민감하게 되고 조속수렴 문제 (premature convergence)를 야기하므로 인하여 결국 성능의 저하를 가져온다 [2,3,5,7,19,20].

우리는 이러한 분석을 통하여 개체의 적합도를 부모 적합도와 평가함수에의하여 계산된 적합도를 이용하여 계산하는 방법을 제안하였다 [21]. 논문 [21]에서는 이러한 문제를 해결하기위하여 평가함수에 의하여 계산된 적합도 (비례항)와 부모개체의 적합도로부터 적합도가 변화된 값 (미분항)을 이

접수일자 : 2002년 11월 13일

완료일자 : 2003년 7월 2일

본 연구는 2003학년도 한성대학교 공학연구센터 특별연구비 지원과제임

용하여 적합도를 계산하는 연구를 수행하였다. 논문 [21] 방법을 통하여 초기 개체의 민감성과 조속 수렴문제를 어느정도 해결하였으나 이 방법은 개체들의 진화과정상의 전체적인 적합도 변화추이를 반영하지 못하는 단점이 있었다. 또한 이 방법에서는 비례항과 미분항이 일대일의 가중치를 가지고 적합도를 계산함으로써 주로 비례항에 크게 의존하는 현상으로 인하여 성능향상에 한계가 있었다.

본 논문에서는 이러한 문제를 해결하기 위하여 새로운 평가 항목인 적분항과 각항에 대한 가중치를 도입하였다. 적분항을 도입하기 위하여 본 논문에서 제안한 비례-적분-미분 평가방법에서는 논문 [21] 에서 사용한 평가함수에 의해 계산된 적합도, 부모 개체의 적합도와 더불어 초기세대로부터 이전 세대까지의 최대, 최소 적합도를 최종 평가에 이용한다. 이를 이용하여 해당 개체가 현재 얼마나 적합한지 (비례평가), 부모 개체보다 얼마나 진화했는지 (미분평가), 초기세대로부터 적합도 변화추이가 어떠한지 (적분평가)를 계산하고 이를 적절히 반영하여 새로운 적합도를 산출한다. 그러므로, 비례-적분-미분 평가 방법에서는 비록 두개의 개체가 동일한 염색체를 갖고 있다할지라도 진화과정의 동일하지 않는 한 미분평가와 적분평가에 따라서 적합도가 다르게 평가되며 이러한 평가는 초기 조수의 큰 적합도를 갖는 개체들이 반복적으로 선택되어 몇세대 후에는 거의 전체의 개체가 한 곳으로 수렴하는 조속수렴현상을 방지할 수 있다. 즉, 현재 적합도가 좋지 않다고 하더라도 진화가 급진적으로 일어나고 있는 개체나, 꾸준히 진화하고 있는 개체의 적합도가 커져서 초기 개체에 대한 민감성과 조속수렴 문제를 해결한다. 또한 비례, 적분, 미분항의 영향에 따른 성능향상을 살펴보기 위하여 각항에 가중치를 두어 실험하였는데 이는 제어분야에서 사용되는 PID 제어기술과 유사한 것으로서 문제에 따라서 적절한 가중치가 존재한다.

본 논문에서 제시한 평가방법의 특성 및 유효성을 살펴보기 위하여 유전자 알고리즘 성능 검증에 많이 사용되어온 대표적인 함수 최적화들을 적용하여 실험해 보았다. 실험결과 비례-적분-미분 각 항의 가중치를 적절히 선택할 경우 제안한 방법이 유전자 알고리즘의 성능을 크게 향상시킬 수 있음을 확인하였다. 단, 이후의 연구에서 비례-적분-미분 각 항의 가중치를 적절히 선택하는 방법에 대한 연구가 필요하다 하겠다. 본 논문은 다음과 같이 구성된다. 2절에서는 비례-적분-미분 평가방법에 대하여 설명한다. 최적화 문제에 대하여 적용한 실험 결과는 3절에 있으며 4절의 결론으로 끝을 맺는다.

## 2. 비례-적분-미분 평가방법

비례-적분-미분 평가방법을 설명하기 위하여 먼저 다음과 같은 새로운 적합도를 정의한다. 아래의 정의에서 부모 적합도와 미분 적합도는 논문 [21]에서의 정의와 동일하다. 혼란을 피하기 위하여 이후의 기술에서 평가함수에 의하여 계산된 적합도를 원시 적합도라한다.

### 정의 1: 부모 적합도, 미분 적합도, 적분 적합도

$t$ 세대에서  $i$ 번째 개체의 원시 적합도를  $f_i(t)$ 라하고  $t-1$ 세대에서  $i$ 번째 개체의 부모 개체들의 원시 적합도를 각각  $f_j(t-1)$  과  $f_k(t-1)$  이라고 하자. 또한 초기 세대부터  $t-1$ 세대까지의 평가중 가장 적은 원시 적합도와 가장 큰 원시 적합도를 각각  $f_{\min}(t-1)$  과  $f_{\max}(t-1)$  이라고 하고 현재의 세대를  $g$ 세대라고 할 경우,  $i$ 번째 개체의 부

모 적합도  $f_i^p(g)$ , 미분 적합도  $f_i^d(g)$ , 적분 적합도  $f_i^i(g)$ 는 각각 다음과 같이 정의된다.

$$f_i^p(g) = \frac{f_j(g-1) + f_k(g-1)}{2} = \text{avg}(f_j(g-1), f_k(g-1)), \quad (1)$$

$$f_i^d(g) = (f_i(g) - f_i^p(g)), \quad (2)$$

$$f_i^i(g) = f_i^i(g-1) + (f_i(g) - \frac{f_{\max}(g-1) - f_{\min}(g-1)}{2} + f_{\min}(g-1)) \quad (3)$$

단,  $i$ 번째 개체의 이전 세대의 적분적합도  $f_i^i(g-1)$ 는 이전 세대의 부모 개체들의 적분적합도의 평균으로서  $f_i^i(g-1) = \text{avg}(f_j^i(g-1), f_k^i(g-1))$  이되며 무작위적으로 생성된 초기 세대는 1세대로서 초기세대에서 이전세대의 적합도들 즉  $f_j(0), f_k(0), f_j^i(0), f_k^i(0), f_i^i(0), f_{\max}(0), f_{\min}(0)$  는 0으로 간주한다.  $\Delta$

부모 적합도는 어떤 개체의 부모 개체를 대표하는 하나의 적합도로서 논문 [21]에서와 같이 평균치를 사용하였다. 미분 적합도는 부모개체보다 얼마나 진화했는지를 나타내는 척도로서 부모적합도보다 적합도가 클경우에 양수가 되어 부모개체로부터의 진화정도를 반영한다 [21]. 또한 적분 적합도는 해당 개체의 적합도가 현재의 세대까지 진화될 동안 찾아낸 최저 적합도와 최고 적합도의 중간보다 클경우에 양수가 되며 세대를 거치면서 부모개체들의 적분 적합도의 평균과 함께 누적되어 초기세대로부터 계속적인 진화 정도를 평가한다.

### 정의 2: 비례-적분-미분 적합도

부모 적합도, 미분 적합도, 적분 적합도가 정의 1과 같이 정의될 경우  $t$ 세대에서  $i$ 번째 개체의 비례-적분-미분 적합도  $f_i^t(t)$  는 다음과 같이 정의된다.

$$f_i^t(t) = P \cdot f_i(t) + I \cdot f_i^i(t) + D \cdot f_i^d(t) \quad (4)$$

다음과같은조건하에서

$$f_i^i(t) = \begin{cases} f_i^i(t) & f_i^i(t) > 0 \text{인 경우} \\ 0 & \text{그외의 경우} \end{cases}$$

$$f_i^d(t) = \begin{cases} f_i^d(t) & f_i^d(t) > 0 \text{인 경우} \\ 0 & \text{그외의 경우} \end{cases}$$

단, 식 (1)에서  $P, I, D$  는 양의 실수로서 각각 비례, 적분, 미분항의 가중치이다.  $\Delta$

정의 1과 2에 의하여, 초기세대인 1세대 비례-적분-미분 적합도  $f_i^t(1)$ 는  $P \cdot f_i(1) + I \cdot f_i^i(1) + D \cdot f_i^d(1)$  이되고 2세대 이후부터 미분항과 적분항이 개체들의 세대간 개체간 적합도 변화를 적용하게 된다. 계산된 비례-적분-미분 적합도는 다음세대의 개체선정을 위한 선택에 사용된다 1). 이렇게 재 계산된 적합도를 다음세대의 부모개체 선정을 위한 선택에 사용함으로써 유전자 알고리즘의 개체 진화에 도움을 주게된다. 식 (1)에서 보듯이 비례-적분-미분 적합도는 원시 적합도가 커질수록 비례 가중치  $P$ 에 비례하여 계산된다. 본 논문에서는 이를 비례효과라고 부른다. 즉 비례효과는 원시 적합도가 좋은 개체를 현재의 진화상에서 얼마나 좋게 볼 것 인지를 결정한다. 함수 최적화 문제를 놓고 보았을때 최적화하려는 함수의 국부해 (local optimum)가 많을수록  $P$ 가 작

1) 비례-적분-미분 적합도는 다른 곳에는 사용되지 않으며 오로지 다음 세대 부모 선택시에만 사용된다.

아아함을 직관적으로 알수있다. 즉 국부해가 많은 함수 최적화 문제에서  $P$  가 클 경우 국부해로 조속수렴 (premature convergence)할 가능성이 높아진다.

적분항은 적분 가중치  $I$ 에 곱해져서 계산되는데 이는 계속적으로 진화해온 개체에 가중치를 주는 효과를 발생시킨다. 이를 적분효과라고 부른다. 적분효과는 진화가 지속적으로 일어난 개체를 다음 세대에 계속적으로 살아남게 함으로써 진화를 가속시키며 진화가 불규칙한 개체나 진화가 잘 일어나지 않는 개체는 적합도를 낮춤으로서 다음 세대로 전이되는 것을 막는다. 미분효과는 어떤 개체가 비록 원시 적합도가 작다고 해도 부모개체로부터 많은 진화가 되었다면 가중치를 주어 다음 세대에 살아남을 가능성을 높여준다. 미분효과에 의하여 조속수렴 문제를 어느정도 미연에 방지하며 조속수렴에 빠졌다고 하더라도 벗어나게하는 효과를 줄수 있다. 조속수렴 상태에서는 대부분의 개체가 국부해 근처에 머물고 있음으로서 대부분의 개체의 염색체가 유사내지 동일하게 되며 이는 결국  $f_i(t) \approx f_i^*(t), f_i(t) \approx f_{\max}(t-1)$ 가 된다. 이런 상태에서는 대부분의 개체가 동일해 동종교배가 주로 일어나기 때문에 교배 (crossover)에 의한 진화는 거의 불가능하다. 또한 돌연변이 (mutation)에 의한 진화는 가능하나 돌연변이가 일어날 확률이 매우 낮고 2) 돌연변이가 일어났다 하더라도 해당 개체의 적합도가 기존 개체에 비해 크지 않을 경우 바로 사라지며, 돌연변이 개체의 적합도가 매우 크더라도 돌연변이에 의하여 생성된 개체의 수가 적기 때문에 확률상 살아남을 가능성이 적어진다. 그러나 본 논문에서 제안한 비례-적분-미분 평가에서는 돌연변이된 개체의 적합도가 클경우에 미분효과로 인하여 다른 개체보다 상당히 큰 적합도를 가지게 되어 계속적으로 살아남을 가능성이 높다. 이러한 미분효과는 유전자 알고리즘이 조속수렴에서 탈출 가능성을 높여준다. 이러한 비례, 적분, 미분 효과에 의하여 비례-적분-미분 평가방법이 적용된 유전자 알고리즘은 개체가 빠르게 진화하면서도 조속수렴 확률을 줄이고 조속수렴 상태에서 빨리 벗어날수 있게 된다. 조속수렴문제는 초기개체의 분포에 따라서 특정 국부해로 수렴할 가능성이 높음으로 인하여 초기 개체에 따라서 수렴되는 최종해가 틀려지는 초기개체 민감성을 야기한다. 비례-미분-적분 평가방법은 조속수렴 확률을 줄임으로 인하여 초기개체의 민감성도 줄이는 부가적인 효과도 나타난다.

### 3. 실험 결과

비례-적분-미분 평가방법의 효용성을 보이기 위하여 2개의 비교적 단순한 함수최적화 문제와 하나의 조합 최적화 문제 그리고 여러 논문에서 사용한 대표적이며 복잡한 3개의 함수 최적화 문제를 이용하여 실험하였다. 적용한 6개의 함수 최적화 문제는 다음과 같다.

$$f_1 = x^2, \text{ where } -20 \leq x \leq 30 \quad (5)$$

$$f_2 = 10x \cdot \text{sgn}(x) (\sin(\pi x) + 1) (\sin(10\pi x) + 1), \text{ where } -20 \leq x \leq 10 \quad (6)$$

$$f_3 = \sum_{j=1}^k m_j \begin{cases} m_j = 1 & \text{if } T_j = I_j \\ m_j = 0 & \text{if } T_j \neq I_j \end{cases} \quad (7)$$

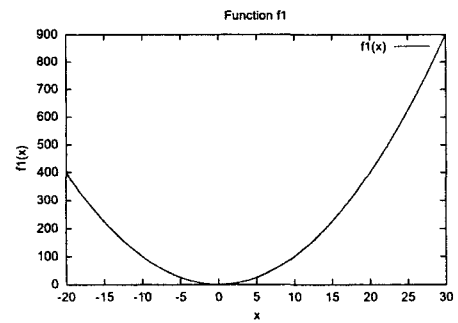
2) 돌연변이 확률을 높일경우 무작위적인 검색 (random search) 경향으로 특정해에 수렴하기 어렵다. 이를 위해 일반적으로 유전자 알고리즘에서는 돌연변이 확률을 0.05 정도로 낮게 준다.

$$f_4 = 100(x_1^2 - x_2)^2 + (1 - x_1)^2, \text{ where } -2.048 \leq x_i \leq 2.048 \quad (8)$$

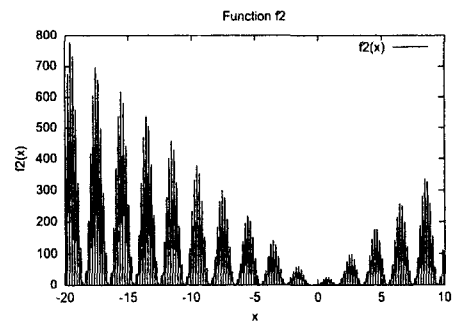
$$f_5 = 0.5 - \frac{\sin(\sqrt{x_1^2 + x_2^2}) \sin(\sqrt{x_1^2 + x_2^2}) - 0.5}{(1.0 + 0.001(x_1^2 + x_2^2))(1.0 + 0.001(x_1^2 + x_2^2))}, \text{ where } -10 \leq x_i \leq -10 \quad (9)$$

$$f_6 = (x_1^2 + x_2^2)^{0.25} \sin(50(x_1^2 + x_2^2)^{0.1} + 1)^2, \text{ where } -10 \leq x_i \leq -10 \quad (10)$$

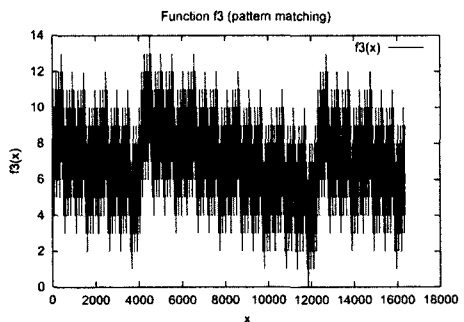
$f_1$  은 하나의 국부해만을 갖는 단순한 함수이며  $f_2$  는 비교적 복잡하나  $x$  하나만 갖는 1차원 함수이다.  $f_3$  는 조합 최적화 문제의 하나로 각 개체의 0과 1의 문자열을 패턴으로 보고 찾으려는 특정 패턴을 찾는 문제이다.  $f_3$  에서  $T_j$  는 찾으려는 문자열의  $j$ 번째 문자값이고  $I_j$  는 임의의 개체  $I$  의  $j$ 번째 문자값이다. 그러므로  $I_j$  가  $T_j$  와 일치할때  $m_j$  는 1이되고, 불일치할때  $m_j$  는 0이 되며  $f_3$ 의 최대 값은 개체가 코딩된 비트수  $h$ 와 일치한다.  $f_4, f_5, f_6$  은 여러 다른 논문에서 함수 최적화 문제로 많이 사용된 함수로서  $f_4$  은 DeJong 함수 2이고  $f_5$ 는 멕시칸 모자 (Mexican hat) 함수, 그리고  $f_6$ 은 Schafer 함수이다.



(a)



(b)



(c)

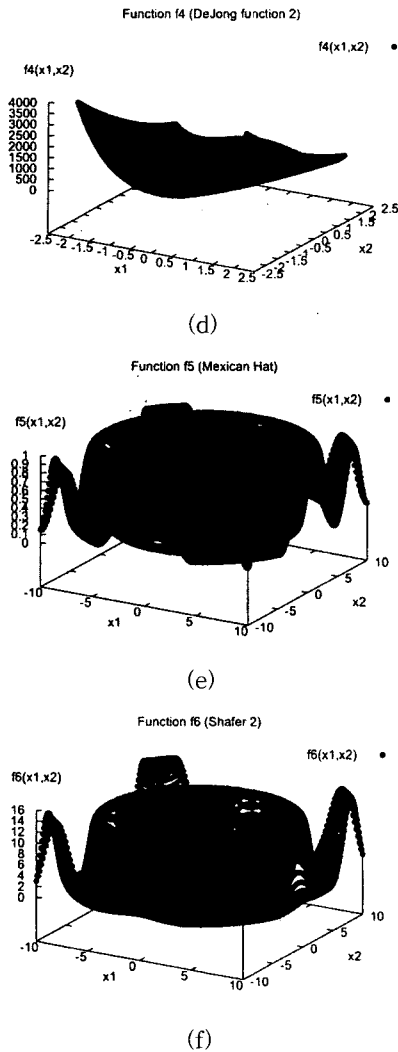


그림 1. 실험함수 (a)  $f_1$  (b)  $f_2$  (c)  $f_3$  (d)  $f_4$  (e)  $f_5$  (f)  $f_6$  (Experimental functions (a)  $f_1$  (b)  $f_2$  (c)  $f_3$  (d)  $f_4$  (e)  $f_5$  (f)  $f_6$ )

그림 1은 6개의 적용한 문제의 입출력 관계를 나타낸다. 입출력 관계의 데이터를 추출하기 위하여 개체 코딩 비트수는 14비트로 했으며 각 함수별로 모든 비트가 0에서 부터 모든 비트가 1이되게 입력하여 출력을 구하였다. 특히  $f_3$  에서의 찾을 문자열은 각 비트별로 0.5의 확률로 0과 1을 무작위적으로 선정하여 설정하게 하였고 결국 최대 적합도는 개체 코딩 비트수와 같은 14가 된다.

위에 기술한 6개의 최적화 문제를 이용하여 비례-적분-미분 평가방법의 성능을 평가하였으며 성능 평가에 사용된 파라미터 값은 표 1과 같다.  $P, I, D$  가중치는 가중치 값에 따른 성능을 파악하기 위하여 임의의 5개를 선택하였으며 그 이외에 파라미터는 실험의도에 문제가 없는 일반적으로 많이 사용되는 값을 선택하여 사용하였다. 5개의 가중치는 가중치 값에 따른 성능의 차이를 살펴보기 위하여 임의로 선택한 것으로서 차후 연구에서 해당 문제에 적합한 가중치를 찾는 연구가 필요하다 하겠다.

표 1. 실험을 위한 파라미터 설정 (Parameter setting for experiments)

파라미터	값
교배확률 ( $p_c$ )	0.6
돌연변이 확률 ( $p_m$ )	0.05
개체수	10
개체 코딩 비트수	12, 18 bits
$P, I, D$ 가중치	0, 0.01, 0.1, 1, 10

개체 코딩 비트수와  $P, I, D$  가중치에 따른 실험을 수행하여 총 실험결과수는 1488개 (6 문제  $\times$  2 비트수  $\times$  5개  $P$  가중치  $\times$  5개  $I$  가중치  $\times$  5개  $D$  가중치 - 6 문제  $\times$  2 비트수  $\times$  1개의  $P, I, D$  가 모두 0인 경우) 이다. 이중에  $P=1, I=0, D=0$  인 것은 원래의 유전자 알고리즘의 결과 값이며  $P=1, I=0, D=1$  인 것은 논문 [21]에서 사용한 방법이다. 실험결과수가 매우 많으므로 본 논문에서는 각 문제별로 최저/최고의 결과치, 원래의 유전자 알고리즘 결과, 그리고 논문 [21]의 결과만을 표 2에 나타내었다. 실험 결과값은 10번의 실험결과를 평균한 값이며 표준편차도 같이 나타내었다. 물론 10번의 실험시 초기개체는 모두 다르게 생성된다. 표 2에서 논문 [21]의  $P, I, D$  는 모두 1,0,1 이며 원방법의  $P, I, D$  는 모두 1, 0, 0 으로서 생략하였다.

표 2. 실험결과 (Experimental results)

함수		$f_1$		$f_2$		$f_3$	
비트수		12	18	12	18	12	18
최고	평균	25.40	82.00	109.20	393.20	10.90	22.50
	편차	15.49	89.18	90.65	365.28	3.27	11.47
	$P$	0	0	0.1	0.01	0.01	0.01
	$I$	0	0	0.01	0	0.01	0
	$D$	0.01	0.01	0.01	10	10	10
최저	평균	538.80	20518.70	413.70	8756.90	1078.10	66462.40
	편차	621.84	14849.54	271.36	9820.01	1292.70	81678.80
	$P$	0.01	0	0.1	0.1	0.01	0.01
	$I$	10	10	0.1	10	0.01	1
	$D$	0.1	10	0.01	0.1	0	0
논문[21]	평균	100.40	2503.00	300.10	793.60	58.10	737.70
	편차	126.38	2395.46	169.69	961.76	47.49	566.73
원방법	평균	72.40	2873.90	180.00	1546.80	60.80	1148.30
	편차	40.34	2459.29	120.69	1277.87	47.72	958.46

함수		$f_4$		$f_5$		$f_6$	
비트수		12	18	12	18	12	18
최고	평균	54.80	114.40	401.10	10971.50	268.10	3289.00
	편차	47.72	103.77	297.34	16216.72	354.70	1825.89
	$P$	0	0	0.1	0	0.01	0.01
	$I$	0	0	0.01	0	0.01	0
	$D$	0.01	0.01	1	0.01	1	1
최저	평균	13371.50	12354.30	3193.00	57048.80	1947.60	35716.70
	편차	16634.75	14624.59	2137.77	40707.54	2556.28	41325.64
	$P$	0.1	1	10	0.01	0.1	10
	$I$	0	0	0	0.1	0	0.01
	$D$	10	10	0.1	1	10	0
논문[21]	평균	2157.20	1504.20	1375.20	26774.20	473.20	7434.00
	편차	2441.23	2107.25	1113.54	30345.85	477.74	5148.40
원방법	평균	2694.80	3365.10	1274.10	17394.90	1027.70	4341.90
	편차	3697.48	4297.97	1292.24	14809.49	774.45	3038.51

표 2에서 최고라고 표기된 행은 실험결과 최고의 성능을 보인 경우에 관련된 값을 보여준다. 예를들어,  $f_5$ 에서 12비트로 개체를 나타낸 경우에 최고 성능은  $P, I, D$  가 각각 0.1, 0.01, 1일 경우에 발생하며 이때 최적해를 찾는 세대는 평균 401.1세대이다. 최저의 성능은  $P, I, D$  가 각각 10, 0, 0.1일 경우이고 이때 최적해를 찾는 세대는 평균 3193 세대이다. 모든 실험함수에서 최저의 결과는 논문 [21] 나 원방법보다 못하다. 이는  $P, I, D$ 를 잘못 선정하게 되면 오히려 성능이 저하 될 수 있음을 나타낸다. 그러나 반대로 최고의 성능에서는 논문 [21]이나 원방법보다 좋은 것으로 보아 적절한  $P, I, D$ 를 선정하는 알고리즘을 개발할 경우 본 논문에서 제안한 방법이 유전자 알고리즘 성능향상에 큰 효과를 나타낼 수 있음을 보여준다.

표 3. 원방법과 최고/최저 결과간의 성능비교  
(Performance comparison between SGA and the best/the worst results)

함수		$f_1$		$f_2$		$f_3$	
비트수		12	18	12	18	12	18
최고	평균	25.40	82.00	109.20	393.20	10.90	22.50
최저	평균	538.80	20518.70	413.70	8756.90	1078.10	66462.40
원방법	평균	72.40	2873.90	180.00	1546.80	60.80	1148.30
최고	원방법/최고	2.85	35.05	1.65	3.93	5.58	51.04
최저	최저/원방법	7.44	7.14	2.29	5.66	17.73	57.88

함수		$f_4$		$f_5$		$f_6$	
비트수		12	18	12	18	12	18
최고	평균	54.80	114.40	401.10	10971.50	268.10	3289.00
최저	평균	13371.50	12354.30	3193.00	57048.80	1947.60	35716.70
원방법	평균	2694.80	3365.10	1274.10	17394.90	1027.70	4341.90
최고	원방법/최고	49.18	29.42	3.18	1.59	3.83	1.32
최저	최저/원방법	4.96	3.67	2.50	3.28	1.90	8.23

우리는 이후의 논문에서 적절한  $P, I, D$ 를 선정하는 연구를 수행할 예정이다. 논문 [21]의 방법이 몇개의 결과에서 원방법 보다 나쁜 성능을 보이는데 반하여 비례-적분-미분 평가 방법의 최고의 결과는 원방법 과 논문 [21]의 결과보다 월등히 좋아짐을 볼수 있다. 표 3은 원방법에 비하여 최고의 성능이 얼마나 좋아졌는지와 원방법에 비하여 최저의 성능이 얼마나 나쁜지를 비율로 나타낸 결과이다. 표 3에서 보면  $f_1$  과 같이 복잡도가 높지 않은 (다른 말로 국부해가 많지 않은) 함수의 경우 비트수가 많아져서 탐색공간이 증대될 경우 비례-적분-미분 방법이 상당한 효과가 있음을 알수 있다. 전반적으로  $f_1, f_2, f_3$ 와 같이 비교적 단순한 문제에서는 탐색공간이 증대될수록 비례-적분-미분 방법이 더욱 좋은 성능을 보였다. 그러나,  $f_4, f_5, f_6$  과 같이 복잡한 문제에서는 탐색공간이 증대될수록 성능향상이 적어짐을 볼수 있다. 이는 문제의 복잡도가 높아지고 탐색공간이 증대될수록 개체선택 방법의 차이가 주는 효과가 줄어들음을 의미하는 것으로 직관적으로 이해할수 있다. 그러나, 문제의 복잡도가 높고 탐색공간이 큰 곳에서 유전자 알고리즘의 성능향상이 어려움을 감안할때 표 3에서 보여준 정도의 성능향상은 의미있는 것으로 판단된다.

실험결과에서 또하나의 특이한 점은  $f_1$  과  $f_4$  의 실험결과가 원방법에 비해 매우 우수하게 나온 것과 두 방법에서 모두 최고결과의  $P, I, D$  가 0, 0, 0.01 로 동일하다는 것이다. 이것은 다른 함수에 비해  $f_1$  과  $f_4$  의 국부해 영역이 작고 국부해와 최적해의 적합도 차이가 커서 발생된 것으로 판단된다. 즉 국부해와 최적해의 적합도 차이가 클 경우 미분 효과가 실험결과에 많은 영향을 미치는 것으로 판단된다.

위 실험에 이어서 우리는 보다 많은 개체수에서의 비례-적분-미분평가 방법의 효용성을 보고자 개체수를 30으로 하여 실험하였다. 일반적으로 개체수가 많아지면 조속수렴 현상은 줄어들는데 그 이유는 다음과 같다.

- 개체수가 많음으로 인하여 초기 개체들이 전역 최적해 근처에 생성될 가능성이 높게 된다. 이렇게 되면 개체 진화가 전역최적해 쪽으로 일어날 가능성 또한 높아지게 되어 대부분의 개체가 초기에 지역 최적해로 수렴하는 조속수렴 현상이 줄어들게 된다.
- 개체수가 많으면 개체 분포가 보다 조밀하게 되어 좁은 영역에 걸쳐 존재하는 전역 최적해를 발견할 가능성이 높아진다.
- 개체수가 많음으로 인하여 상대적으로 적합도가 좋은 몇개의 개체가 집중적으로 선택되는 확률을 낮추어 준다. 즉 개체수가 많음으로 총 적합도가 커지고 이는 총 적합도에 대한 해당 개체의적합도의 비율로 차세대 부모를 선택을 하는 룰렛휠방법에서 좋은 개체의 선택 확률을 낮추기 때문이다.

그러나 개체수가 많아져도 조속수렴 현상이 줄어드는 것 일뿐 사라지는 것은 아니며 조속수렴이 사라질 정도가 되게 하려면 탐색공간에 비해 개체수가 매우 많아져야만 한다. 그러나 개체수를 무한정 많게 할수는 없으며 개체수를 많게 할수록 조속수렴 현상은 줄어드나 점점더 전체탐색 (full search) 경향이 발생되어 탐색의 비효율성을 가져온다. 즉, 개체수가 많아지면 각 세대에서 개체 진화를 위한 처리에 대

표 4. 개체수가 30개 일때의 실험결과 (Experimental results when the number of individuals is 30)

함수		$f_1$		$f_2$		$f_3$	
비트수		12	18	12	18	12	18
최고	평균	8.60	14.50	30.70	147.90	6.30	7.70
	편차	3.50	4.94	14.53	138.85	1.42	0.78
	$P$	0.10	0	0.01	0.01	0.01	0
	$I$	0.01	0	0.10	0	0	0
	$D$	10.0	0.01	10.0	0.10	10	0.01
최저	평균	138.50	6001.30	219.60	1576.30	218.70	15418.20
	편차	185.44	4881.77	361.35	1053.44	165.47	18146.33
	$P$	0.01	0.1	10	0.1	0.01	10
	$I$	0.1	10	0	10	1	1
	$D$	0.1	0.1	1	0.01	0.1	0.01
논문[21]	평균	20.00	609.30	139.80	270.80	19.90	178.40
	편차	10.64	296.61	157.81	161.62	12.27	130.17
원방법	평균	33.30	1125.70	54.20	305.30	21.50	354.10
	편차	29.78	1030.88	38.32	289.73	11.28	351.40

3) 극단적인 예로 개체수를 탐색공간크기와 같게하고 개체 생성을 무작위적이 아니라 탐색공간 범위로 한다면 전체 탐색이 된다.

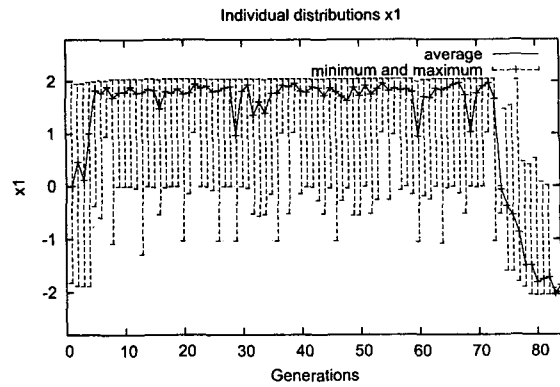
함수		$f_4$		$f_5$		$f_6$	
비트수		12	18	12	18	12	18
최고	평균	48.00	24.80	135.00	2026.40	91.70	995.60
	편차	33.53	24.83	85.59	2338.11	73.98	896.37
	P	0	0	0.01	0	0	0.01
	I	0	0	0.01	0	0.10	0
	D	0.01	0.01	1	0.01	0.01	0.10
최저	평균	44470.20	66112.20	1319.80	19961.40	4183.50	8387.30
	편차	73877.26	92965.88	976.10	13360.30	5525.01	7614.45
	P	0.01	0.1	0.01	0	0.01	10
	I	0	0	0	0.1	0	10
	D	10	10	10	0.01	10	0.01
논문[21]	평균	3002.00	1533.40	319.30	10019.20	1075.90	2179.40
	편차	3437.33	2281.22	211.93	8771.22	815.21	1702.18
원방법	평균	1099.90	4901.00	410.60	5637.80	761.10	1994.70
	편차	1760.39	6769.90	327.47	3330.21	634.15	2092.07

부분의 시간을 소모하기 때문에 빠른 세대간의 진화로 인한 빠른 해로의 접근이라는 유전자 알고리즘의 장점을 희생해 오히려 전체적인 성능은 떨어지는 결과를 가져올 수 있다. 그러므로, 진화를 희생하지 않는 개체수 범위에서 조속수렴 현상을 피해갈 수 있어야 한다. 실험에서 개체수를 30으로 한 것은 이와같이 적절한 개체수에서 본 논문에서 제안한 비례-적분-미분 방법이 얼마나 효과를 미치는지 보기 위함이다. 표 4는 개체수가 30개 일때의 실험결과를 보여준다.

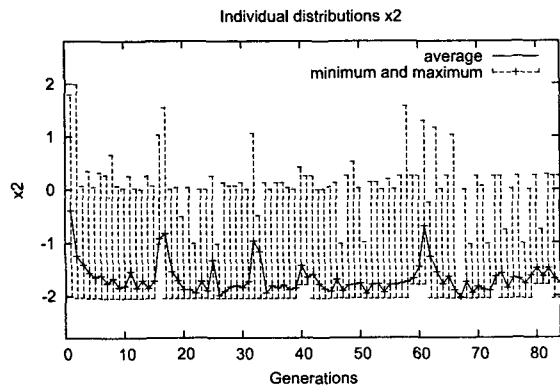
표 4에서 보여주듯이 개체수가 30개에서도 최고성능은 대부분의 경우에 논문 [21] 나 원방법보다 월등히 좋을 수 있다. 특히 함수  $f_4$ 에서 좋은데, 이 함수를 이용하여 제안한 방법이 기존의 방법들보다 성능이 좋아지는 이유를 설명하고자 한다. 본 논문에서 제안한 비례-적분-미분 평가방법은 동적으로 개체를 평가함으로써 비록 초기 개체 의존성에 의하여 지역 최적해에 조속수렴하더라도 빠르게 지역 최적해에서 빠져나오게 된다. 실제로 이렇게 동작하는지를 보기 위하여 함수  $f_4$ 에서 최고의 성능의 경우와 논문 [21]의 경우에 세대별 개체의 분포를 나타내어 보았다 4). 즉, 세대가 진행되어감에 따라서 개체값이 어떻게 진화되어 가는지를 추적하여 개체가 지역최적해에 빠져있는지와 언제쯤 지역최적해를 빠져나오는지를 관찰하였다. 개체수는 30개로 하였고 함수는  $f_4$ 를 이용했으며 개체 코딩 비트수는 18로 하여 최고 성능의 경우와 논문 [21]의 경우에 대하여 한번씩 실험하였다. 실험에서 최고 성능의 경우는 84세대에서 최적해를 찾았고 논문 [21]의 경우는 4816세대에서 최적해를 찾았다 5).

그림 2는 최고의 성능의 경우에 세대별 개체분포를 나타낸다. 초기세대인 1세대에서 각 개체는 무작위적으로 생성된다. 그러므로 초기세대에서 개체 분포의 평균은 해당 개체가 갖을 수 있는 최소/최대값 중간에 위치할 것이며 최소, 최대치는 해당 변수가 갖을 수 있는 최소/최대값에 근접해 있을 것이다. 이는 그림 2에서 확인할 수 있다. 그림 2에서 보면 초기세대인 1세대에서는  $x_1$ 과  $x_2$ 의 최소/최대값은  $x_1$ 과  $x_2$ 의 변수영역  $-2.048$ 과  $+2.048$ 에 가깝게 분포하고 평균은 중간값 0에 근접해 있음을 볼 수 있다. 그림 1에서 보듯이 함

- 4) 분포는 각세대에서 개체들이 표현하는  $x_1$ 과  $x_2$ 값들의 평균과 최소/최대값으로 나타낸다.
- 5) 이러한 결과는 표 4에서 10번씩 수행하여 통계를 낸 결과와 비교하여 조금 큰 값이다.



(a)



(b)

그림 2. 제안한 방법에서 세대따른 개체분포 (a)  $x_1$  (b)  $x_2$ (Individual distribution according to the generations in the proposed method (a)  $x_1$  (b)  $x_2$ )

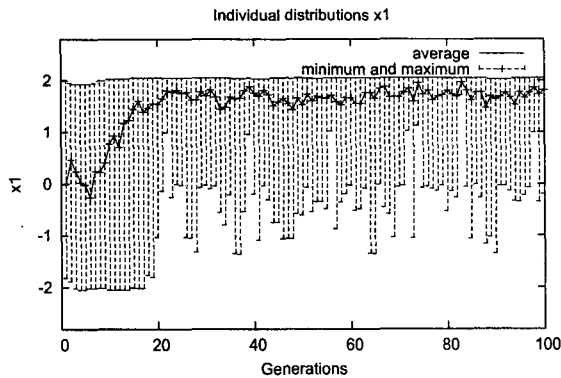
수  $f_4$ 의 최적해는  $x_1$ 과  $x_2$ 가 모두  $-2.048$ 인 곳에 있는데,  $x_2$ 는  $-2.048$  쪽으로 진화를 해 나가는데 반하여  $x_1$ 은  $+2.048$  쪽으로 진화를 하고 있음을 알 수 있다. 이는  $x_1$ 과  $x_2$ 가 모두  $-2.048$  근처에 생성된 초기 개체가 없거나 혹은 생성된 개체는 있으나 해당 개체의 적합도가 크에도 불구하고 확률적 선택에서 선택되지 않은 경우로 볼 수 있다. 또한  $x_1$ 이  $+2.048$  쪽으로 가 있는 개체들의 적합도가  $-2.048$  쪽으로 가 있는 개체보다 상대적으로 커서 이쪽의 개체들이 살아남은 것으로도 볼 수 있다. 비록  $x_1$ 과  $x_2$ 의 최소값이 모두  $-2.048$  근처라 하더라도 하나의 개체가 이 둘을 동시에 만족하는 개체(즉 전역 최적해 근처에 생성된 개체)가 없을 가능성이 높다는 것이다.

초기에  $x_1$ 이  $+2.048$  쪽으로 진화가 일어나더라도, 최고 성능의 P, I, D가 설정된 비례-적분-미분 평가 방법의 경우 그림 2에서 보여 주듯이 70세대 근처에서부터 급격히 최적해 쪽으로 진화해 감을 볼 수 있다. 이는 교배 및 돌연변이에 의하여  $x_1$  및  $x_2$ 가  $-2.048$  근처에 있게끔 진화된 개체가 70세대 근처에서 생성되었으며 비례-적분-미분 평가에 의하여 이 개체의 평가가 매우 우수하게 나왔고 또한 다음 세대 부모 개체로 선택됨으로서 급격히 대부분의 개체가 최적해 쪽으로 진화된 것으로 판단된다. 만약 기존의 개체평가 방법을 사용했을 경우라면  $x_1$  및  $x_2$ 가  $-2.048$  근처에 있게끔 진화된 개체가 발생하더라도 기존의 개체보다 월등히 좋다고 판단되

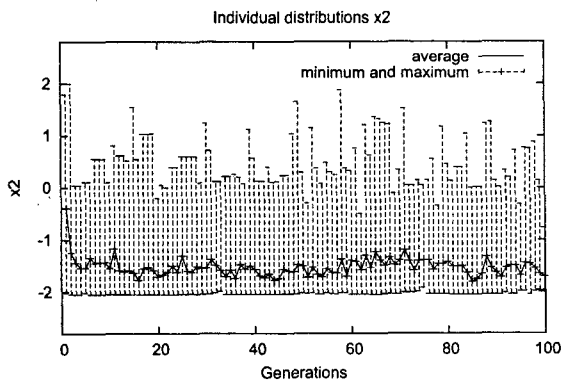
지 않기 때문에 다음세대 부모 개체로 선택될 가능성이 적어서 최적해 쪽으로의 진화가 일어나지 않을 가능성이 높다.

두번째 실험이 이러한 경우를 보여준다. 두번째 실험은 논문 [21] 방법을 사용한 것으로서 이 방법에서는 4816 세대만에 최적해를 찾았다. 비례-적분-미분 평가와의 비교를 위하여 그림 3에서는 초기세대부터 100 세대까지의 분포를 보여준다. 그림에서 보듯이  $x_1$  이 +2.048 쪽의 국부최적해로 수렴한 개체들은 70 세대에서 100세대까지 여전히 해당 영역에 머물러서 빠져나오지 못함을 볼 수 있다. 이러한 경향은 최적해를 찾는 4816세대까지 계속되는데 그림 4는 종료세대 근방에서의 개체들의 분포를 보여준다. 그림 4에서 보면 4792 세대 정도에서  $x_1$  이 -2.048 쪽으로 이동하려고 하다가 4800 세대 정도에서 다시 +2.048 쪽으로 진행하고 4805 세대 정도에서야 비로소 -2.048 쪽으로 확실하게 진화함을 볼 수 있다. 이는 기존의 방법이 얼마나 국부해에서 벗어나기 힘들지를 단적으로 말해 준다고 볼 수 있다.

기존의 평가방법에 비하여 비례-적분-미분 평가의 이러한 특성은 미분과 적분항에서 주어진다 6). 즉 미분항에 의하여  $x_1$  이 -2.048 근처에 있게끔 진화된 개체는 +2.048 근처에서



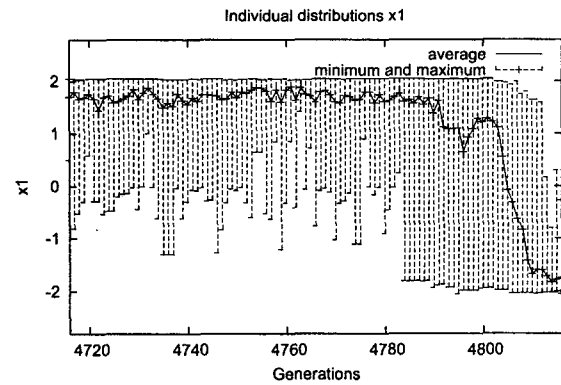
(a)



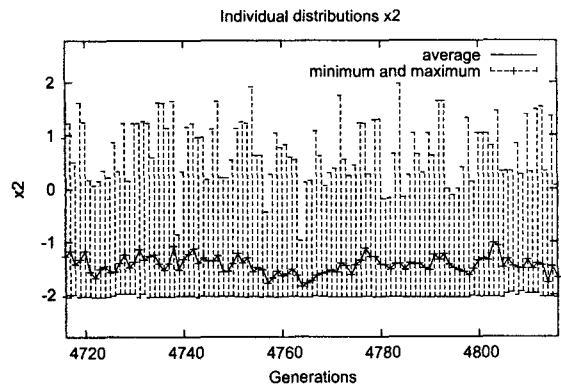
(b)

그림 3. 논문 [21] 방법에서 세대에 따른 개체분포 (초기 과정) (a)  $x_1$  (b)  $x_2$  (Individual distribution according to the generations in the method of paper [21] (initial process) (a)  $x_1$  (b)  $x_2$ )

6) 비록 실험에서 사용한 최고의 성능이  $P, I, D$  가 각각 0, 0, 0.01 이라서 적분항이 없지만 일반적으로 미분항과 적분항이 이와같은 특성을 준다.



(a)



(b)

그림 4. 논문 [21] 방법에서 세대에 따른 개체분포 (최종 과정) (a)  $x_1$  (b)  $x_2$  (Individual distribution according to the generations in the method of paper [21] (final process) (a)  $x_1$  (b)  $x_2$ )

머물고 있는 다른 개체보다도 적합도가 월등히 좋게 평가되고 이러한 평가가 국부 최적해를 벗어나는데 크게 기여한다. 또한 바로  $x_1$  이 -2.048 쪽으로 가지 않았다 하더라도 지속적으로 -2.048 쪽으로 진화한 개체는 적분항에 의하여 지속적으로 좋은 평가를 유지해 국부 최적해를 벗어나게 하는데 기여할 수 있다.

비례-적분-미분 평가 방법의 이러한 장점을 다른 측면으로도 고려해 볼 수 있다. 즉, 개체가 최적해 근처에 머물러 있음에도 불구하고 다른 국부 최적해 쪽으로 가는 경향이 높아져 성능 저하 요인이 될 수도 있다고 볼 수도 있다. 그러나, 비례-적분-미분 평가를 보면 다른 국부 최적해 쪽으로 이동한 개체의 적합도가 기존에 전역 최적해 근처에 있는 개체보다 조금 좋거나 좋지 않음으로 미분 적분항에 의하여 해당 개체가 살아남을 가능성은 작아진다. 즉, 비례-적분-미분 평가 방법은 대부분의 개체가 국부 최적해에 있을 때에는 전역 최적해 쪽으로 빠르게 이동하며, 전역 최적해 근방에서는 다른 국부 최적해 쪽으로 거의 이동하지 않고 전역 최적해 근방을 계속 탐색하는 효과를 준다. 7) 이러한 효과로서 비례-적분-미분 평가가 유전자 알고리즘의 성능향상에 크게 기여함을 볼 수 있다.

7) 물론 이러한 효과가 적절히 발생하는 적당한  $P, I, D$  값이 존재하며 따라서 적당한  $P, I, D$  값을 찾아내는 것이 중요하다.

#### 4. 결 론

본 논문에서 우리는 유전자 알고리즘이 조속수렴현상에 빠지지 않고 최적해를 빠르게 찾아가게 하기 위한 비례-적분-미분 평가 방법을 제안하고 실험을 통하여 비례-적분-미분 방법의 효용성 및 문제점을 살펴 보았다. 실험결과 비례, 적분, 미분 가중치  $P, I, D$  가 적절히 선택되었을 경우 평가 함수만에 의한 적합도 산정을 하는 SGA 나 가중치 1의 비례, 미분항만을 사용하는 방법보다 유전자 알고리즘의 성능을 크게 향상시킬수 있음을 보였다. 그러나, 적절치 못한 가중치를 사용할 경우 유전자 알고리즘의 성능이 오히려 낮아지는 결과를 초래하였다. 이후의 연구에서는 이러한 문제를 해결하기 위하여 문제의 특성에 따른 적절한  $P, I, D$ 를 찾아서 해당 가중치를 적용시키는 방법 및 매 세대별 적절한 가중치를 찾아서 세대별로 적용시키는 방법등의 연구가 필요하다 하겠다.

#### 참 고 문 헌

- [1] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley, 1989.
- [2] K. Kristinsson and G. A. Dumont, "System Identification and Control Using Genetic Algorithms," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 22, pp. 1033-1046, Sep/Oct 1992.
- [3] C. L. Karr and E. J. Gentry, "Fuzzy Control of pH Using Genetic Algorithms," *IEEE Trans. on Fuzzy Systems*, vol. 1, pp. 46-53, Jan. 1993.
- [4] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: A Survey," *IEEE Computer Magazine*, pp. 17-26, June 1994.
- [5] 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.
- [6] T. kuo and S.-Y. Hwang, "A genetic algorithm with disruptive selection," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 26, no. 2, pp. 299-307, 1996.
- [7] 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.
- [8] A. J., M. Z., and M. J., "GAVaPS-a Genetic Algorithm with Varying Population Size," *Proceedings of the Evolutionary Computation Conference, part of the IEEE World Congress on Computational Intelligence*, June 1994. Orlando.
- [9] J. E. Smith and T. C. Fogarty, "Operator and parameter adaptation in genetic algorithms," *Soft computing : a fusion of foundations, methodologies and applications*, vol. 92, no. 2, pp. 81-87, 1997.
- [10] S. H. Jung, "A Genetic Algorithm with Ageing Chromosomes," *Journal of Fuzzy Logic and Intelligent Systems*, vol. 7, pp. 16-24, June 1997.
- [11] 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.
- [12] S. H. Jung, "Eugenic Genetic Algorithm," *Journal of Fuzzy Logic and Intelligent Systems*, vol. 9, pp. 81-88, Feb. 1999.
- [13] L. Davis, "Adapting Operator Probabilities in Genetic Algorithms," in *Proceedings of the 3rd International Conference on Genetic Algorithms and their Applications*, pp. 61-69, 1989.
- [14] M. Srinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 24, pp. 656-667, Apr. 1994.
- [15] A. Tuson and P. Ross, "Adapting Operator Settings In Genetic Algorithms," *Evolutionary Computation*, vol. 6, no. 2, pp.161-184, 1998.
- [16] C. W. Ho, K. H. Lee, and K. S. Leung, "A Genetic Algorithm Based on Mutation and Crossover with Adaptive Probabilities," in *Proceedings of the 1999 Congress on Evolutionary Computation*, vol. 1, pp. 768-775, 1999.
- [17] S. H. Jung, "Self-tuning of Operator Probabilities in Genetic Algorithms," *전자공학회 논문지*, vol. 37, pp. 29-44, Sept. 2000.
- [18] I.-K. Jeong and J.-J. Lee, "Adaptive Simulated Annealing Genetic Algorithm for System Identification," *Engineering Applications of Artificial Intelligence*, vol. 9, pp. 523-532, Oct. 1996.
- [19] G. A. Vignaux and Z. Michalewicz, "A Genetic Algorithm for the Linear Transportation Problem," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 21, pp. 445-452, MARCH/APRIL 1991.
- [20] M. McInerney and A. P. Dhawan, "Use of Genetic Algorithms with Back Propagation in Training of Feed-Forward Neural Networks," *International Conference on Neural Network*, pp. 203-208, 1993.
- [21] S. H. Jung, "Two Dimensional Evaluation Scheme in Genetic Algorithms," *Journal of Electrical Engineering and Information Science*, vol. 4, pp. 561-570, Oct. 1999.



저 자 소 개



**정성훈(Sung Hoon Jung)**

1988년 : 한양대학교 전자공학과 (공학사)

1991년 : 한국과학기술원 전기및전자공학과 (공학석사)

1995년 : 한국과학기술원 전기및전자공학과 (공학박사)

1996년 ~ 현재 : 한성대학교 정보공학부 부교수

관심분야 : 뉴로-퍼지 및 소프트웨어 컴퓨팅, 진화연산알고리즘, 인공지능, 지능제어, 패턴인식, 시뮬레이션

Phone : +82-2-760-4344

Fax : +82-2-760-4435

E-mail: shjung@hansung.ac.kr