http://dx.doi.org/10.5392/JKCA.2014.14.11.541

유전자형-표현형 개념을 적용한 수정된 이진 입자군집최적화 (버전 2)

Modified Binary Particle Swarm Optimization using Genotype-Phenotype Concept (Version 2)

임승균, 이상욱
목원대학교 정보통신융합공학부

Seungkyun Lim(tmdrbs9021@naver.com), Sangwook Lee(slee@mokwon.ac.kr)

요약

본 논문에서는 유전알고리즘의 유전자형-표현형 기법을 적용한 수정된 이진 입자군집최적화의 두 번째 버전을 소개한다. 입자군집최적화는 해를 탐색해 나가는 과정에서 주변의 우수한 해의 위치와 자신의 위치 차이 정보를 이용한다. 이러한 위치 차이를 구하는데 있어서 첫 번째 버전의 수정된 이진 입자군집최적화는 표현형을 사용한 반면에 제안하는 버전은 유전자형을 사용한다. 이진 정보만을 제공하는 표현형에 비해 연속 공간 전체를 탐색공간으로 제공하는 유전자형 정보를 사용하여 해 공간을 보다 넓은 공간으로 표시할 수 있다. 벤치마크 함수인 10개의 De Jong 함수에 실험한 결과, 두 번째 버전은 탐색 공간이 넓고 지역 최적해가 많은 함수에서 첫 번째 버전에 보다 우수한 결과를 얻었다.

■ 중심어 : | 이진 입자군집최적화 | 유전자형과 표현형 |

Abstract

In this paper, we introduce a second version of modified binary particle swarm optimization using a concept of genotype-phenotype in genetic algorithms. Particle swarm optimization uses an information of difference between a position of the best solution and one's own position in the process of searching optimum. To obtain this difference of positions, the first version of modified binary particle swarm optimization uses a phenotype but the proposed second version uses a genotype. We can represent the solution space in large search space by using a genotype which provides continuous whole space as search space compared to a phenotype which provides only binary information. Experimental results in 10 De Jong benchmark function show that the second version outperforms the first version in six functions which has a broad search space and many local optima.

■ keyword : | Binary Particle Swarm Optimization | Genotype-phenotype |

I. 서 론

입자 군집 최적화 (Particle Swarm Optimization, PSO)는 연속적인 문제 공간을 위한 최적화 기법으로 1995년 Kennedy와 Eberhart에 의해 처음 소개되었다 [2]. PSO는 작은 유기체들의 행동특성으로부터 영감을

접수일자 : 2014년 09월 11일
수정일자 : 2014년 10월 17일
심사완료일 : 2014년 10월 20일
교신저자 : 이상욱, e-mail : slee@mokwon.ac.kr

얻었다. 대표적인 예로는 꿀벌들의 행동 특징을 말할 수 있다. 벌은 벌통 주변의 먹이를 획득할 적절한 장소(탐색 영역)를 탐색한다. 먹이가 있는 장소를 찾은 벌은 집으로 돌아와 다른 벌에게 각각 자신이 발견한 먹이의 위치 정보(최적해)를 전달한다. 이러한 정보 공유를 통해 벌들은 최적의 먹이 위치를 찾게 된다. 이러한 자연적인 현상의 관찰로부터 만들어진 PSO는 해를 탐색 공간 위의 위치로 표현하며 속도 갱신 함수 및 위치 갱신 함수를 사용하여 해를 탐색해 나간다.

1997년, Kennedy and Eberhart는 PSO를 조합최적화 문제에 적용하기위해 이진 버전의 PSO를 제안하였다 (Binary Particle Swarm Optimization, BPSO)[2]. BPSO는 기존 PSO와 속도 갱신하는 방법은 동일하나 위치 갱신하는 방법에 있어서 현재의 위치와 갱신된 속도의 합으로 위치 갱신을 수행하는 PSO와는 달리 갱신된 속도 정보만 사용하여 위치 정보를 갱신하며 해를 탐색해 나간다.

2008년, Sangwook Lee는 위치 갱신에 현재 위치를 반영하지 않는 기존의 BPSO를 유전알고리즘의 유전자형-표현형 개념을 적용하여 위치 갱신에 현재 위치를 반영하는 수정된 이진 입자군집최적화를 (Modified Binary Particle Swarm Optimization, MBPSO) 소개하였다[1]. 다양한 벤치마크 함수에 적용한 결과 MBPSO는 BPSO보다 매우 우수한 성능을 보임을 확인하였다.

기존 MBPSO는 속도갱신에 있어 표현형의 이진정보를 사용하였다. 이진정보 특성으로 속도 갱신에 있어서 탐색 영역이 제한되는 문제를 해결하기 위해, 본 논문에서 우리는 속도갱신에 유전자형의 연속 공간 정보를 적용한 이진 입자군집최적화 버전 2 (Modified Binary Particle Swarm Optimization (Version 2), MBPSO2)를 소개한다.

본 논문의 구성은 아래와 같다. 2장에서는 PSO 및 BPSO를 설명하고 PSO 알고리즘 변수에 대한 가이드라인을 제시한다. 3장에서는 본 논문에서 제안하는 MBPSO2를 소개한다. 4장에서는 실험과정과 결과를 나타내고 5장의 결론을 끝으로 마친다.

II. 배 경

2.1 PSO

PSO 해집단의 각 입자는 탐색공간 안에 랜덤으로 초기화되어 생성되며 최적의 해를 찾기 위한 이동은 입자의 속도와 위치 값을 적적하게 결합하여 이루어진다.

각 반복마다 모든 입자의 위치와 속도는 이전 세대의 최적해(p_{best}, i, j)와 모든 세대의 최적해(g_{best}, i, j)에 따라 갱신된다. PSO 버전에서 모든 입자들의 속도와 위치는 아래공식에 따라 갱신된다[2][5].

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1R_1(p_{best,i,j} - x_{i,j}(t)) + c_2R_2(g_{best,i,j} - x_{i,j}(t)) \quad (1)$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \quad (2)$$

수식에서 i는 해집단에 위치한 입자의 순서이고(i=1, . . ., n), j는 입자들의 위치다(j=1, . . ., m). t는 반복회수, $v_{i,j}(t)$는 i번째 입자의 속도이고 $x_{i,j}(t)$는 위치다. 여기서 R_1과 R_2는 0과 1사이의 무작위수이고, c_1과 c_2는 관계 상수이다. 마지막으로 w는 이전 속도에 대한 관성 가중치이다[3].

2.2 BPSO

BPSO 알고리즘은 PSO에 이진 탐색 방법을 적용한 것이다. 속도를 갱신하는 방법은 PSO와 같으며 갱신된 속도는 위치가 0 또는 1의 값을 갖도록 하는 확률값으로 사용된다. BPSO에서 속도 갱신을 위한 식(1)은 변하지 않지만 위치 갱신을 위한 식(2)는 식(3)으로 다시 정의 된다[4].

$$x_{i,j}(t+1) = \begin{cases} 0 \text{ if } rand() \geq S(v_{i,j}(t+1)) \\ 1 \text{ if } rand() < S(v_{i,j}(t+1)) \end{cases} \quad (3)$$

식(3)에서 $S(\bullet)$는 속도를 0과 1사이의 값으로 변환하는 시그모이드 함수로 식(4)와 같다.

$$S(v_{i,j}(t+1)) = \frac{1}{1+e^{-v_{i,j}(t+1)}} \quad (4)$$

식(3)의 $rand()$는 난수 발생기로 0.0에서 1.0사이의 값을 리턴한다.

BPSO에서 위치는 0과 1의 값을 가지기 때문에 불연속공간이며 위치 갱신에서 현재의 위치를 반영하지 않으므로 속도 영역과 위치 영역을 각각 탐색 공간으로 볼 수 있다.

2.3 PSO의 변수

2.3.1 가속과 관성 상수

속도 갱신 식(1)에서 가속계수 c_1과 c_2는 현재 속도 값($v_{i,j}(t+1)$) 갱신에 중요한 역할을 한다. c_1은 입자들의 최고값($p_{best,i,j}(t)$), c_2는 이웃한 입자들 중 최고값($g_{best,i,j}(t)$)에 영향을 준다.

관성 변수 w는 현재속도에 이전속도가 미치는 영향을 조절한다. w는 입자의 궤적을 부드럽게 하여 PSO 알고리즘이 최적의 해에 수렴하도록 하는 방법이다. 참고문헌 [5]는 두 상수(가속과 관성)에 의해 좋은 수렴이 보장되는 것을 보여준다. 두 상수의 관계는 중간 변수 ϕ와 함께 아래 수식으로 정의된다.

$$\begin{cases} w = \dfrac{1}{\phi - 1 + \sqrt{\phi^2 - 2\phi}} \\ c1 = c2 = \phi w \end{cases} \tag{5}$$

식(5)에서 ϕ는 2보다 큰 값이고, w는 실수 값을 가진다. 게다가 [6]의 실험결과는 c_1과 c_2는 1보다 반듯이 커야한다고 보여준다. 이로써 ϕ의 범위는 반듯이 [2.01, 2.4]사의 값을 갖도록 한다.

2.3.2 최대 속도($V_{\max}$)

최대 속도는 입자의 속도가 일정 속도를 초과하지 않도록 제한을 하는 것을 말한다. 이런 제약은 $V_{\max}$라 불리며, 입자의 가속 값이 통제 불능이거나 탐색공간의 크기를 벗어나지 않도록 한다. 가속 값은 [$-V_{\max}$, $V_{\max}$]사이의 값을 갖으며, 위 범위를 벗어난 경우 $V_{\max}$ 값으로 제한된다. $V_{\max}$의 값이 너무 작은 경우 입자가 탐색하는 간격은 매우 작아 알고리즘이 지역 최적해에 빠지게 된다. 따라서 최적의 해를 찾기 위해서는 반복횟수를 증가해야 할 것이다[1].

Ⅲ. 두 번째 버전의 수정된 이진 입자군집최적화

이 장에서는 BPSO의 속도 갱신에 유전알고리즘의 유전자형-표현형 개념을 적용한 MBPSO를 설명하고 MBPSO의 성능을 향상 시킨 두 번째 버전의 수정된 이진 입자군집최적화인 MBPSO2를 소개한다. 먼저 유전자형-표현형 개념에 대해 설명하고 MBPSO를 설명한 후 MBPSO2를 소개한다. 마지막으로 미성숙 수렴(premature convergence)을 방지하기 위한 돌연변이 기법을 소개한다.

3.1 유전자형과 표현형

유전자형은 각각의 유전자로부터 얻은 유전 정보로써, 유전자가 어떻게 발현 할지 여부를 결정한다. 표현형은 개인의 유전자로부터 발현되며 신체적 특징이나 성격과 같은 외부적으로 표현된 것을 의미한다. 표현형의 일부분은 유전자로부터 발현되지만 객체의 주변 환경 및 삶의 방향으로부터 영향을 받는다.

유전 알고리즘에서 유전자형과 표현형은 이미 많은 부분에 코딩된 방법이다. 예를 들면, 다음과 같은 랜덤키 코딩이 있다. 랜덤키 표현은 잠재해가 난수로 표현되고, 이 값들은 객체를 해석하는데 키로 사용된다. 예를들어 (0.42, 0.14, 0.68, 0.11, 0.98)로 표현된 공정 순서를 정하기 위한 해가 있다고 하자. 이를 해석하여 실제 공정 순서를 표현하는 방법이 오름차순 정리라고 하면 실제 공정순서는 (4->2->1->3->5)의 순으로 나열된다 (0.11->0.14->0.42->0.98). 여기서 나열된 공정순서는 표현형이고 난수는 유전자형이다.

3.2 MBPSO

PSO와 BPSO의 큰 차이점은 위치 갱신 함수이다. 특히, BPSO의 위치 갱신의 경우 식(3)과 같이 위치정보를 전혀 사용 하지 않는다. 다시 말해서 BPSO의 다음 위치는 현재 위치와 관계없이 오직 이전 속도($v_{i,j}(t)$)

에 의해 결정 된다. 이 문제를 개선한 것이 속도를 또 하나의 탐색 범위로 간주하고 속도와 위치를 유전자형과 표현형으로 변경한 MBPSO이다[1]. MBPSO는 PSO와 BPSO의 식(1), (3), (4)를 아래와 같이 변경한다.

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1R_1(x_{pPbest,i,j} - x_{p,i,j}(t) + c_2R_2(x_{pGbest,i,j} - x_{p,i,j}(t)) \quad (6)$$

$$x_{g,i,j}(t+1) = x_{g,i,j}(t) + v_{i,j}(t+1) \quad (7)$$

$$x_{p,i,j}(t+1) = \begin{cases} 0 \text{ if } rand() \geq S(x_{g,i,j}(t+1)) \\ 1 \text{ if } rand() < S(x_{g,i,j}(t+1)) \end{cases} \quad (8)$$

위의 식(8)에서 $S(\bullet)$의 정의는 아래와 같다.

$$S(x_{g,i,j}(t+1)) = \frac{1}{1+e^{-x_{g,i,j}(t+1)}} \quad (9)$$

식(8)은 표현형으로 변경된 BPSO의 위치 갱신 함수로 시그모이드 함수에 x_g값을 대입한다. x_p는 식(8)에 의해 0과 1의 이진값을 갖는다. 식(7)의 x_g는 유전자형으로 현재 값과 갱신된 속도의 합으로 갱신된다. 식(6)은 속도 갱신함수로 0과 1의 값만 가지는 표현형으로 (x_{pGbest}, x_p) 및 (x_{pPbest}, x_p)의 차를 계산하여 0, -1 또는 1의 3가지 경우 값 만이 계산결과로 나올 수 있다. 이로 인해 $v_{i,j}$의 계산결과는 PSO에 비해 탐색범위가 매우 협소함을 예상할 수 있다.

3.3 MBPSO2

MBPSO의 속도 갱신 식(6)에서 우수해와 현재해와의 차이 값이 (0, 1, -1)의 3가지 경우만 나옴으로 인해 다양한 탐색을 할 수 없는 사실을 확인할 수 있었다. 이러한 단점을 보안하기 위해 우수해와 현재해와의 차이를 계산하는 방식에 있어서, 이진 값만 가지는 표현형을 사용하는 것을 전체 실수 값을 가지는 유전자형으로 변경하여 MBPSO의 두 번째 버전(MBPSO2)을 제안한다. 식(10)은 MBPSO2의 속도 갱신 식을 보여준다.

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1R_1(x_{gGbest,i,j} - x_{g,i,j}(t) + c_2R_2(x_{gPbest,i,j} - x_{g,i,j}(t)) \quad (10)$$

x_p는 $S(x_g)$를 통해 생성된 0과 1사이의 난 수 값이 0에 가까우면 0이 될 확률이 높고, 1에 가까우면 1이 될 확률이 높다. 따라서 실제 적용되는 표현형 x_p 값 0과 1은 유전형 x_g의 실수 값이 확률적으로 음의 방향으로 매우 큰 것과 양의 방향으로 매우 큰 것으로 각각 매칭시킬 수 있다. MBPSO2를 통해 표현형 x_p를 유전형 x_g로 변경하는 것은 확률적인 오차를 야기할 수는 있으나 0, 1의 탐색에서 실수 전체의 탐색 공간으로 확장한다는 측면에서 보다 더 표준 PSO의 개념에 적합하다 볼 수 있다.

3.4 돌연변이

돌연변이란 유전알고리즘에서 세대가 반복되더라도 해의 다양성을 유지하기 위해 정해진 확률에 의해 유전자가 변하도록 하는 연산을 말한다. 이진 표현에서 예를 들면, 일정한 확률에 의해 0인 유전자의 값을 1로 또는 1의 유전자 값을 0으로 변경하는 것을 말한다.

BPSO에서는 최적의 해를 찾기 위해 탐색할 때, 속도의 값은 목표 위치가 1인 경우는 $V_{\max}$, 목표치가 0인 경우는 $-V_{\max}$ 값을 향해 가는 경향이 있다. 속도가 $V_{\max}$ 또는 $-V_{\max}$ 에 도달한 경우는 0 또는 1의 값에서 빠져나오기가 힘들며, 만약 이 값이 미성숙 수렴에 의한 잘못된 값으로 수렴한 것이라면 미성숙에서 빠져나오게 할 수 있는 연산이 필요하다. 이 문제를 해결하기 위해, 우리는 유전알고리즘의 돌연변이 연산과 유사하게 작동할 식(11)을 BPSO의 속도 갱신함수 식 (1)과 위치 갱신함수 (2)사이에 추가하였다.

$$\begin{aligned} &\text{for}(i=1; i<n; i=i+1\ )\{ \\ &\text{if}(rand()\ < r_{mu}) \\ &\text{then } v_{i,j_r}(t+1)\ = -v_{i,j_r}(t+1)\} \end{aligned} \quad (11)$$

위의 식에서 r_{mu}는 0과 1사이의 난수로서 제안한 돌연변이 기능을 수행할 확률을 나타내고, j_r은 랜덤하게

선택된 돌연변이 연산을 수행할 유전자를 의미한다. BPSO에서 식(11)의 연산은 유전알고리즘의 이진변이 확률과 유사하다[7]. 즉, 일정한 확률에 대하여, 유전자는 0에서 1로 또는 1에서 0으로 변경된다.

한편, MBPSO2 알고리즘에서 각각의 X_{gmax} 및 x_{g,i,j_r}는 BPSO의 $V_{\max}$ 및 v_{ijr}과 같다. MBPSO2의 변이를 위해 아래와 같은 공식을 식(7)과 (8)사이에 추가했다.

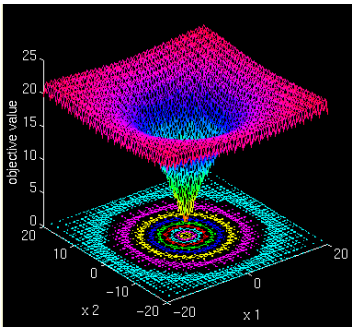
$$\begin{aligned} &\text{for}(i=1; i<n; i=i+1)\{ \\ &\text{if}(rand() < r_{mu}) \\ &\text{then } x_{g,i,j_r}(t+1) = -x_{g,i,j_r}(t+1) \ \} \end{aligned} \tag{11}$$

Ⅳ. 실험

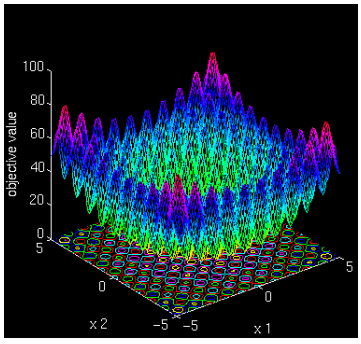
이 실험에서는 벤치마크함수인 De Jong의 10가지 함수에 MBPSO2와 MBPSO를 적용하여 얻은 결과를 비교하고 분석한다. 실험환경은 AMD FX-815 3.61Ghz 성능의 옥타코어 CPU와 16GB메모리, C++를 사용하였다.

4.1 평가 함수

알고리즘 성능 측정을 위한 평가함수들은 최적해를 쉽게 찾을 수 없도록 무수히 많은 지역 최적해를 가지고, 단 하나의 전역 최적해을 가진다. 알고리즘의 성능은 평가 함수의 지역 최적해에 머무르지 않고 전역 최적해를 빠르게 찾는 것으로 평가된다.



(a) Ackley's Path 함수(f_8) 그래프



(b) Rastrigin's 함수(f_{10}) 그래프

그림 1. 알고리즘 측정을 위한 평가함수[9][10]

[그림 1]은 f_8과 f_{10} 함수의 그래프로 무수히 많은 지역 최적해와 단 하나의 전역 최적해를 갖는 예이다. 1에서 10까지의 함수 모두 단 하나의 전역 최적해를 갖으며, 최적해의 값은 최소값인 0으로 한다. 보다 자세한 함수의 모양 및 그래프는 참조문헌 [10]에 자세히 기술되어있다.

아래는 본 실험에서 사용한 10가지 평가함수의 계산식이다.

$$f_1(x) = \sum_{i=1}^{n} x_i^2$$

$$f_2(x) = 100(x_1^2 - x_2^2)^2 + (1-x_1)^2$$

$$f_3(x) = 6 \bullet \sum_{i=0}^{5} int(x_i)$$

$$f_4(x) = \sum_{i=1}^{n} i \bullet x_i^4 + N(0,1)$$

$$\frac{1}{f_5(x)} = \frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}$$

$$f_6(x) = 0.5 + \frac{\sin^2 \sqrt{x^2+y^2} - 0.5}{(1.0 + 0.001(x^2+y^2))^2}$$

$$f_7(x) = \frac{1}{4000} \sum_{i=1}^{n} (x_i - 100)^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i - 100}{\sqrt{i}}\right) + 1$$

$$f_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$$

$$f_9(x)=\sum_{i=1}^{n}100((x_{i+1}-x_i^2)^2+(x_i-1)^2)$$

$$f_{10}(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$$

위에 $int(x_i)$숫자의 끝이고, $N(0,1)$는 가우시안 노이즈이며, $a_{ij}=\begin{pmatrix}-32&-16&0&16&32&-32\\-32&-32&-32&-32&-32&-16\end{pmatrix}\cdots$ 이다.

모든 평가함수는 최소화 문제이며 [표 1]에 각 함수의 특징을 정리하였다. [표 2]에서 Dim은 실수 공간상의 탐색 공간의 차원, B.Dim은 2진수 공간상의 탐색 공간의 차원, Opt는 함수의 최적해를 나타낸다. f_5(전역 최소값 = 1)를 제외한 모든 평가함수 들은 전역 최소값 0.0으로 설정되어 있다.

4.2 변수 설정

PSO 알고리즘에서 변수 설정은 최적의 해를 찾기 위해 매우 중요한 역할을 한다. 첫째, 랜덤 변수 발생에 관한 것, C언어에서 발생하는 $rand()$ 함수는 좋은 랜덤 수를 발생하지 않는다[5]. 이러한 이유로 우리는 *KISS* $rand()$ 함수를 사용하였다. *KISS* $rand()$는 일반적인 $rand()$보다 랜덤 수 발생에 우수한 특징을 갖고 있다. 갱신 함수의 계수에 관해서(즉,w, c_1, c_2), Clerc[5]가 제안한 ϕ=2.07을 사용하고, $c_1, c_2 = 0.689343$, $w = 1.42694$로 하였다.

표 1. 평가함수 상세정보[1]

F	명칭	Dim	탐색 공간	B.dim	Opt.
f_1	Spherical	3	(−5.12, 5.12)	30	0
f_2	2D Rosenbrock	2	(−2,048, 2.048)	24	0
f_3	Step Function	5	(−5.12, 5.12)	50	0
f_4	Quartic	30	(−1.28, 1.28)	240	0
f_5	Foxholes	2	(−65536.0, 65536.0)	34	≈1
f_6	Schaffer' s	2	(−100.0, 100.0)	22	0
f_7	Griewank	30	(−300.0, 300.0)	390	0
f_8	Ackley	30	(−30.0, 30.0)	300	0
f_9	Rosenbrock	30	(−2.048, 2.048)	360	0
f_{10}	Rastrigin	30	(−5.12, 5.12)	300	0

4.3 실험 결과

[표 2]과 [표 3]는 돌연변이 연산을 사용하지 않을 때와 사용할 때의 MBPSO와 MBPSO2 성능 비교 결과를 각각 보여주고 있다. 실험에 사용된 알고리즘 환경으로 해집단의 수는 40, 반복횟수는 1000을 사용하였다. 결과 값은 30번씩 실행된 값을 평균하여 얻은 값을 기록하였다.

변이 확률은 0.1에서 0.9까지 변화하며 실험한 결과 중 성능이 가장 우수할 때를 나타내었다. 변이확률 항목에서 'All'은 모든 변이확률에 대해 우수한 성능을 나타내며 '0'은 변이확률이 없는 경우 가장 우수한 결과를 보여 준 것을 뜻한다. MBPSO와 MBPSO2 모두 돌연변이가 없는 실험보다 돌연변이가 있는 실험에서 더 우수한 성능을 보였다. 표에서 볼드 표시된 값은 MBPSO2가 MBPSO보다 성능이 우수한 경우를 나타내었다. 10개의 평가 함수 중에서 탐색 공간이 넓고 지역 최적해가 많은 f_4, f_6, f_7, f_8, f_9, f_{10} 함수에서 MBPSO보다 우수한 반면, 비교적 단순하고 지역 최적해가 적은 f_1, f_2, f_3 함수에서는 탐색공간이 적은 MBPSO가 우수하거나 비슷한 결과를 보였다.

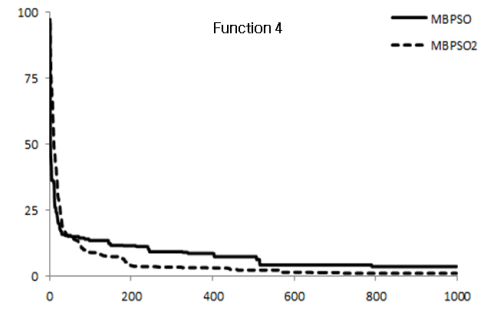
표 2. 비교 결과(돌연변이 없음)

F	MBPSO2	MBPSO
f_1	0.00038±0.00027	0.00015±0.00001
f_2	0.0000835±0.00004	0.00041±0.00007
f_3	0.0±0.0	0.0±0.0
f_4	1.0952596±0.12061	2.66643±0.36769
f_5	1.4778555±2.31615	1.02889±0.07352
f_6	0.0048±0.00981	0.0073±0.00042
f_7	8.074078±4.39371	8.50211±4.80884
f_8	2.031223±0.11322	3.63762±0.62202
f_9	26721.9±1.3	26769.2±0.0
f_{10}	39.47±5.90821	49.82±7.35563

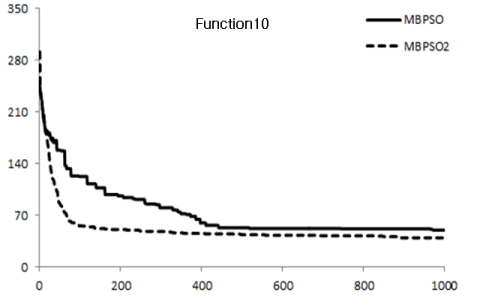
표 3. 비교 결과(돌연변이 있음)

F	변이 확률	MBPSO2	변이 확률	MBPSO
f_1	0.1	0.00028±0.0004	0.3	0.00013±0.000014
f_2	0.2	0.0000028±0.0002	0.7	0.00009±0.000002
f_3	0.1	0.0±0.0	All	0.0±0.0
f_4	0	1.0952596±0.12	0.7	2.7598±5.29545
f_5	0.4	0.991±0.000036	0.7	0.998004±0.0
f_6	0.1	0.0058±0.0097	0	0.007303±0.000422
f_7	0	8.074078±4.39371	0.4	6.128562±3.650294
f_8	0	2.031223±0.113	0	3.637617±0.622022
f_9	0.1	26669.3±0.615	0	26769.1±0.0
f_{10}	0	39.47±5.90821	0	49.82±7.355633

[그림 2]는 MBPSO와 MBPSO2의 수렴 특성을 비교하여 보여주고 있다. 10가지 평가함수 중 f_4, f_{10}의 결과 그래프이다. 그래프에서 볼 수 있드시 MBPSO2가 MBPSO의 결과보다 우수함과 동시에 빠른 수렴성을 보여주고 있음을 확인할 수 있다.



(a) function 4의 MBPSO와 MBPSO2 비교



(b) function 10의 MBPSO와 MBPSO2 비교

그림 2. MBPSO 및 MBPSO2 수렴 특성 비교

V. 결론

본 논문에서는 MBPSO의 속도갱신방법에서 이진 탐색공간의 표현형 정보를 사용하는 것을 실수 탐색공간의 유전자형 정보를 사용하는 것으로 수정한 MBPSO2를 소개하였다. 10개의 벤치마크 함수에 실험한 결과, 제안한 MBPSO2의 성능이 기존의 MBPSO보다 향상된 것을 보여주었다. 특히, MBPSO2는 지역 지역최적해가 많은 문제에서 더욱 강점을 보였는데, 이는 제안한 MBPSO2가 탐색 공간에서 해의 다양성을 유지하여 미성숙 수렴의 확률을 감소시켜주기 때문으로 분석된다. 해의 다양성을 보완하여 우수한 결과 얻은 것은 PSO가 속한 그룹인 군집 지능(Swarm Intelligence) 기법 중에 하나인 개미집단최적화(Ant Colony Optimization)에서도 관찰된 내용이다[11].

참 고 문 헌

[1] S. W. Lee, S. M. Soak, and S. H. Oh, Witold Pedrycz, and M. G. Jeon, “Modified binary particle swarm optimization,” Progress in Natural Science, Vol.18, No.9, pp.1161-1166, 2008.

[2] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” Proceedings of IEEE international conference on neural networks, Vol.4, pp.1942-1948, 1995.

[3] S. W. Lee, “Particle Swarm Optimization for Scheduling and Permutation Problems using Random Key Representation,” The Korea Contents Society, pp.331-332, 2010.

[4] R. C. Eberhart, “A discrete binary version of the particle swarm algorithm,” Proceedings of 1997 conference systems man cybernetics, pp.4104-4108, 1997.

[5] M. Clerc and J. Kenney, “The particle swarm-explosion, stability, and convergence

in a multidimensional complex space," IEEE Trans Evol Comput 2002, Vol.6, pp.58-73, 2002.

[6] M. Clerc, *Particle swarm optimization*, ISTE Pub, 2006.

[7] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," Proc 2002 Congress Evol Comput 2002, Vol.2, pp.1671-1676, 2002.

[8] J. C. Bean, "Genetics and random keys for sequencing and optimization," ORSA J Comput, Vol.6, pp.154-160, 1994.

[9] J. H. Holland, *Adaptation in natural and artificial system*, USA: Uni-versity of Michigan Press, 1975.

[10] http://www.denizyuret.com/pub/aitr1569/node19.html

[11] 이승관, 최진혁, "개미 집단 최적화에서 강화와 다양화의 조화", 한국콘텐츠학회논문지, 제11권, 제3호, pp.100-107, 2011.

저 자 소 개

임 승 균(Seungkyun Lim) 준회원

- 2013년 2월 : 목원대학교 정보통신공학과(공학사)
- 2013년 3월 ~ 현재 : 목원대학교 IT공학과 석사과정

<관심분야> : 최적화, 병렬처리, 전파 및 안테나

이 상 욱(Sangwook Lee) 정회원

- 2000년 2월 : 한국과학기술원 기계공학과(공학사)
- 2002년 2월 : 광주과학기술원 기전공학과(공학석사)
- 2007년 8월 : 광주과학기술원 정보기전공학부(공학박사)
- 2007년 9월 ~ 2008년 9월 : 조지아공과대학교 전산학과 박사후연구원
- 2008년 11월 ~ 2009년 2월 : 삼성전자 통신연구소 책임연구원
- 2009년 3월 ~ 현재 : 목원대학교 정보통신공학과 조교수

<관심분야> : 휴리스틱 알고리즘, 최적화, 병렬처리, 빅데이터