Particle 2-Swarm Optimization for Robust Search

강 석 호* · 김 승*

《目 次》

요약

본 연구에서는 최근들어 관심을 받고 있는 메타 휴리스틱 방법론인 PSO 알고리즘의 조기수렴 문제를 해결하기 위해 새롭게 수정된 형태의 PSO 알고리즘을 제안한다. 기존의 PSO 알고리즘에서는 모든 군체(swarm)가 해당 반복단계까지의 군체 최적해 및 전체 최적해의 방향으로 위치를 이동하므로 조기수렴에 빠지기 쉬운 단점이 있었다. 본 연구에서 제안하는 P2SO(Particle 2-Swarm Optimization) 알고리즘은 군체를 positive와 negative의 두 종류로 나누어 positive 군체는 기존 PSO에서와 같은 방식으로 해의 위치를 이동하는 반면에 negative 군체는 해의 다양성을 유지하려는 방향으로 해의 이동을 유도하여 조기수렴을 막을 수 있도록 하였다. 제안된 방법론의 성능을 확인하기 위해 몇가지 비선형 함수들에 대해 실험을 진행하였으며, 실험결과 P2SO의 우수성을 확인하였다.

Ⅰ. Introduction

PSO(Particle Swarm Optimization)은 1995년 Kennedy와 Eberhart에 의해 처음 소개되었다 [1]. PSO는 1975년 Holland가 개발한 유전 알고리즘과 유사하게 집단을 토대로 한 최적화 기법으로 초기 해집단을 무작위로 생성하여 최적해를 탐색한다 [2]. 유전 알고리즘이 자연선택의 진화 메커니즘을 모방한 반면 PSO는 새떼와 물고기떼와 같은 생체군집의 사회적 행동양식을

* 서울대학교 산업공학과

바탕으로 하고 있다. PSO에서 개별해는 particle로, 해집단은 swarm으로 표현된다. 각 particle은 최적의 해를 얻기 위해 다차원 공간을 '날아다니며', 그들 자신과 그들 이웃의 경험에 대한 정보를 이용하여 최적의 위치로 이동해 간다.

기본적인 PSO 알고리즘은 해의 수렴과정의 후반부에 들어가면서 모든 particle이 국소 최적에 빠지게 되는 문제점이 여러 연구들을 통해 노출되고 있다. 이러한 조기 수렴문제를 해결하기 위해 Fuzzy PSO [3], Hybrid PSO [4], Intelligent PSO [5] 등의 변형된 PSO 기법이 소개되었다.

본 연구에서는 PSO의 조기수렴 문제를 해결하기 위해 새로운 P2SO 알고리즘을 제안한다.

Ⅱ. Basic PSO Algorithm

POS의 기본 아이디어는 새나 물고기 등이 천적에 대한 자기 방어의 목적으로 군체를 이루어 행동하거나, 또는 벌이나 개미 등이 음식물을 찾기 위해 떼를 이루어 행동하는 것과 같은 사회적 행동 양식을 모방하고 있다. 이렇게 군집을 이루게 되면 개별 개체들은 서로간의 협동을 통해 목적한 바를 효과적으로 이룰 수 있게 된다. PSO는 최적화 문제를 풀기 위한 하나의 도구로 이러한 생물체들의 사회적 행동 양식을 모방하고 있다.

PSO에서는 개별 잠재해를 입자(particle)로 나타낸다. 즉, 결정변수의 개수가 m이라면 PSO에서의 탐색공간은 m차원의 실수 공간이 되고 t시점에서의 j번째 입자, $x_j(t)$는 $x_j(t)=[x_{j1}(t), x_{j2}(t), \cdots, x_{jm}(t)]$의 실수벡터로 나타난다. Swarm은 이러한 입자들의 집합이다. n개의 입자를 운용한다면 t시점에서의 swarm, S(t)는 $S(t)=[x_1(t), x_2(t), \cdots, x_n(t)]^T$로 나타낼 수 있다.

개별 입자들은 '위치'와 '속도'라는 속성을 갖는다. 반복과정에서 입자들은 다차원 탐색공간을 옮겨다니며 적합도 함수에 의해 평가된 적합도 함수값을 가지고, 이를 바탕으로 보다 적합도 함수값이 높은 위치로 이동한다. 이때 입자들은 두 가지 최적 위치를 참고 삼아 다음위치로 이동하게 되는데 각 입자가 반복을 통해 발견한 최적해(pbest)와 모든 입자의 반복을 통해 발견된 전체 최적해(gbest)가 그것이다. t시간의 j번째 입자의 속도는 아래식에 의해 결정된다.

$$v_j(t+1) = w(t)\cdot v_j(t) + c_1\cdot r_1\cdot(pbest_j(t) - x_j(t)) + c_2\cdot r_2\cdot(gbest(t) - x_j(t)) \quad \cdots\cdots (1)$$

여기서, w(t)는 이전속도에 대한 관성하중, c_1, c_2는 가속상수, r_1, r_2는 (0,1)의 난수이다. 속도의 값이 너무 클 때는 입자가 최적해의 위치를 지나칠 수 있고, 속도가 너무 작을 때는 해공간을 충분히 탐색하지 못하는 경우가 생길수도 있으므로 적절한 값의 선택이 요구된다. 관성하중

w(t)는 현재시점의 속도에 대한 기존 속도의 영향을 조절하는 역할을 한다. 큰 w(t)는 전역탐색 능력을 강화시켜주고 작은 w(t)는 지역탐색능력을 강화시킨다. 따라서, 탐색 초기에는 큰 w(t)를 갖고 탐색후반에는 작은 w(t)값을 갖도록하는 방법을 사용한다. 경우에 따라서 w(t)를 식 2의 형태와 같은 지수적 감소 함수를 사용하기도 하고, 식 3과 같은 w_{max}와 w_{min}사이의 선형감소 함수를 사용할 수도 있다.

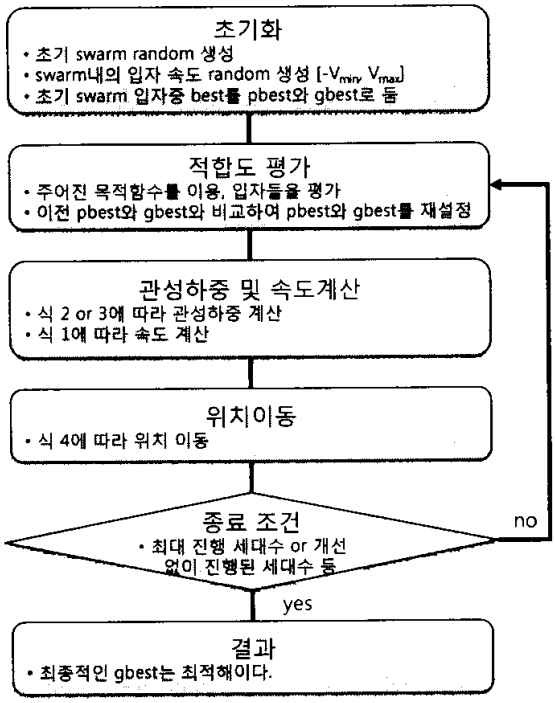
$$w(t) = aw(t-1),\ a\text{는 1에 가까운 감소상수} \quad \cdots\cdots (2)$$

$$w(t) = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times t,\ iter_{\max}\text{는 총 generation 수} \quad \cdots\cdots (3)$$

관성하중값과 속도값이 계산되면 입자는 아래 식 4를 사용하여 현재위치에서 다음세대의 위치로 이동한다.

$$x_j(t) = v_j(t) + x_j(t-1) \quad \cdots\cdots (4)$$

위 과정을 대략적인 순서도로 나타내면 아래 〈그림 1〉과 같다.



〈그림 1〉 Basic PSO의 순서도

Ⅲ. P2SO Algorithm

유전자 알고리즘 (GA: Genetic Algorithm)이나 Nearest Neighborhood Search (NNS), Simulated Annealing (SA), Tabu Search (TS) 등의 메타 휴리스틱들은 사용하는 연산이나 파라메터 값의 조정에 따라 국소최적에 빠지는 위험을 줄일 수 있다. 가령, GA와 같은 경우는 돌연변이율을 높인다던지 선별압력이 높지 않은 교배연산을 사용함으로써 모집단의 다양성을 키울 수 있고, 이를 통해 국소최적에 빠지는 위험을 줄일 수 있다. PSO의 경우는 해공간상에 초기 위치시키는 입자들의 고른 배치, 큰 관성상수 값, 가속상수에 부가되는 임의성, swarm의 크기 등을 통해 국소최적에 빠지는 위험을 줄일 수 있다. PSO는 해(입자)의 개선 방향이 조금씩 다른 해들을 집단 운용(swarm) 하므로 완전한 greedy hill-climbing 이라고 할 수는 없으며, 개별 입자들은 pbest와 gbest의 방향으로 해들을 개선시켜가되 난수를 통한 임의성을 가미하기는 하나, 한번 개략적인 해의 이동방향이 정해지게 되면 세대가 진행될수록 해집단의 다양성은 떨어지게 되고 국소최적에 빠질 위험성이 알고리즘 자체에 내제되어 있다고 할 수 있다.

본 연구에서는 기본적인 PSO의 조기수렴을 해결하기 위해 swarm을 positive와 negative의 2개로 나누고 positive swarm은 기본적인 PSO의 방향으로 해를 개선하되 negative swarm은 pbest와 gbest의 반대방향으로 해의 위치를 이동하도록 하는 bidirectional repositioning 전략을 사용하였다. Positive swarm에 속한 입자들은 기존 PSO에서의 입자들의 역할과 마찬가지로 세대를 거듭하며 pbest 및 gbest를 추종하여 해를 개선시켜나가는 역할을 한다. 이와 반대 방향으로 움직이는 negative swarm에 속한 입자들은 특정 시점(세대) 이전까지는 positive swarm과 동일한 방향 및 속도로 움직이다가 특정시점이 되면 pbest 및 gbest와는 반대방향으로 움직이기 시작해서 전체 swarm의 다양성을 높이는 역할을 하도록 하였다. 이 특정시점을 '폭파시점'이라고 하며 negative swarm이 gbest를 중심으로 다차원 공간상에서 방사상으로 고르게 퍼지게 하기 위해서는 해의 개선이 몇세대 동안 이루어지지 않을 때와 같이 어느 정도 국소최적에 이르렀다고 판단이 되는 시점을 폭파시점으로 삼는것이 적절하다.

폭파시점 이후에는 매 세대 진행과정에서 positive swarm과 negative swarm의 평가를 통해 negative swarm의 개체가 positive swarm의 개체보다 평가값이 높다면 두 개체의 위치를 교환하는 교환연산을 수행한다.

위 설명을 수식화하면 다음과 같다.

$$S(t) = PS(t) + NS(t)$$
$$= [x_1(t), x_2(t), \ldots, x_{\lceil n\times(1-\alpha)\rceil}(t), x_{\lceil n\times(1-\alpha)\rceil+1}(t), \ldots, x_n(t)]^T \quad \cdots\cdots (5)$$

P2SO에서는 위 식 5와 같이 전체 swarm을 positive swarm, PS와 negative swarm, NS로 구분한다. 전체 swarm의 수를 |S|=n 이라고 하면, PS의 개수는 $|PS| = \lceil n \times (1-\alpha) \rceil$이고 NS의 개수는 $|NS| = \lfloor n \times \alpha \rfloor$이 된다. 주어진 문제에 국소최적의 덫이 많다면 α값을 높여서 negative swarm의 크기를 키워야 할 것이다.

P2SO에서의 속도계산식은 다음 식 6과 같다.

$$v_j(t+1) = \begin{cases} w_1(t)\cdot v_j(t) + c_1\cdot r_1\cdot(pbest_j(t) - x_j(t)) & if\ j < \lceil n\times(1-\alpha)\rceil \\ \quad + c_2\cdot r_2\cdot(gbest(t) - x_j(t)), & \\ w_2(t)\cdot v_j(t) + coef\cdot c_3\cdot r_3\cdot(pbest_j(t) - x_j(t)) & otherwise \\ \quad + coef\cdot c_4\cdot r_4\cdot(gbest(t) - x_j(t)) & \end{cases}$$

$$coef = \begin{cases} 1 & if\ t < \text{explosion\_time} \\ -1 & otherwise \end{cases} \quad \cdots\cdots\cdots\cdots\cdots (6)$$

Ⅳ. Experiments

P2SO의 국소최적 탈출능력을 검증하기 위해 실험을 진행하였다. 실험에 사용된 함수는 아래 T1~T3의 3개 함수를 사용하였다. PSO와 P2SO의 비교를 위해 두 방법 모두 동일한 조건 - 초기해의 분포 : 변수 범위내에서 일양분포, 가속상수 : 2.0, 관성하중 : 1.0, 반복횟수 : 30회, swarm 크기 : 9 - 를 부여하였다.

T1. Bimodal function : $f(x) = \frac{(x_1-4)(x_1-6)(x_1-14)(x_1-20)}{100} + (x_2-20)^2,\ 0 \le x_1, x_2 \le 30$

(Global Min value : -13.6416 at $(x_1, x_2) = (17.6779, 20)$, 1 local min)

T2. De Jong's function : $f(x) = \sum_{i=1}^{2} x_i^2,\ -2 \le x_i \le 2$

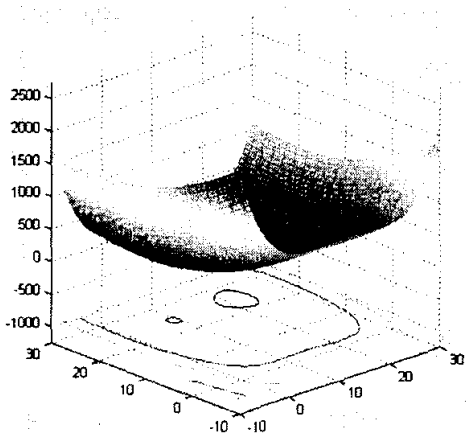
(Global Min value : 0 at $(x_1, x_2) = (0, 0)$, no local min)

T3. Six-hump camelback function :

$$f(x) = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1x_2 + 4(x_2^2 - 1)x_2,\ -2 \le x_1, x_2 \le 2$$

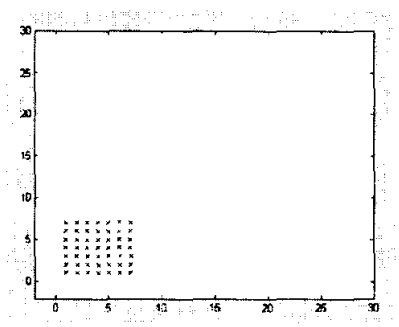
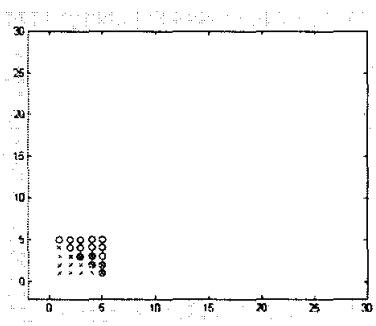
(2 Global Min value : -1.0316 at $(x_1, x_2) = (-0.0898, 0.7126)$, $(0.0898, -0.7126)$, 4 local min)

먼저 T1 함수는 최소화문제일 때 x=4.9125, y=20.0000에서 국소최적해=-1.3606값을 갖고, x=17.6779, y=20.0000에서 전체최적해=-13.6416을 갖는다. 목적함수식의 개략적인 형태는 〈그림 2〉와 같다.

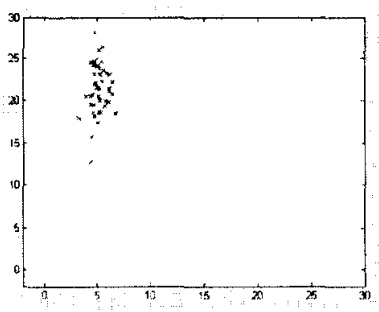
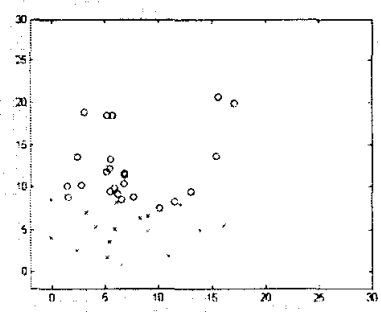
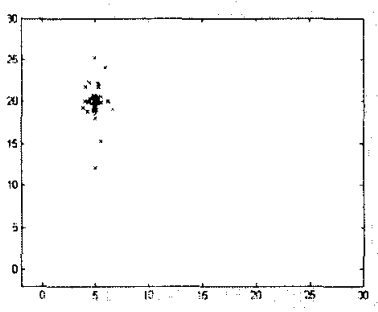
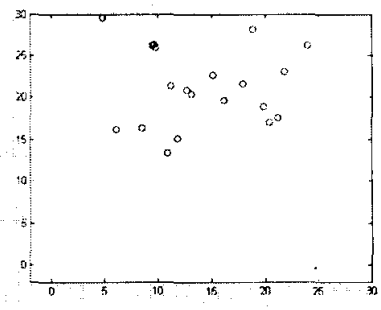
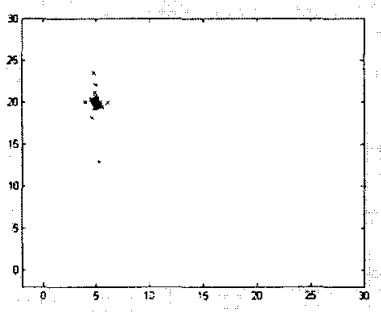
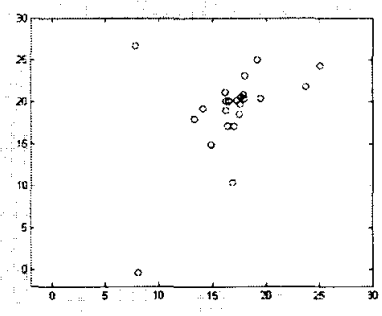
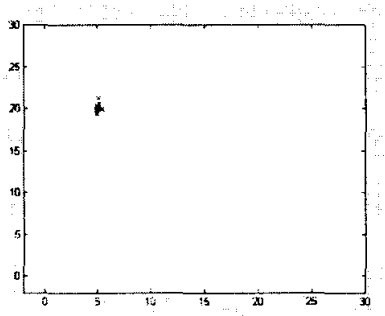
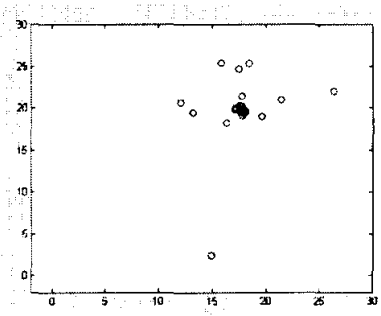


〈그림 2〉 T1 함수의 3차원 plot

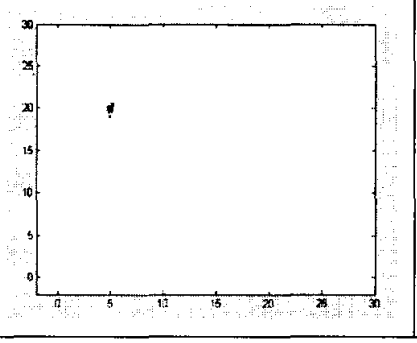
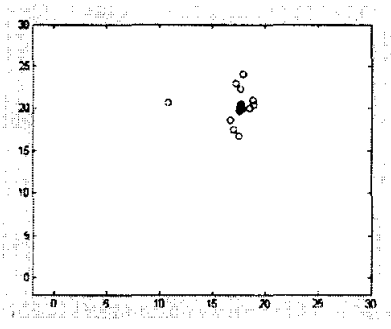
아래 〈그림 3〉은 T1함수 문제를 PSO와 P2SO의 두 알고리즘에 대해서 풀었을 때 각 세대별 swarm의 이동상황을 보여주고 있다. Swarm의 이동을 쉽게 파악하기 위해 초기 swarm의 위치를 (0,0) 근방으로 하고 swarm의 크기를 49로 하였다.

	Basic-PSO	P2SO
(iteration 0)		

〈그림 3〉 Basic-PSO와 P2SO의 해의 이동 비교

	Basic-PSO	P2SO
(iteration 6)		
(iteration 12)		
(iteration 18)		
(iteration 24)		

〈그림 3〉 Basic-PSO와 P2SO의 해의 이동 비교 (계속)

	Basic-PSO	P2SO
(iteration 30)		

〈그림 3〉 Basic-PSO와 P2SO의 해의 이동 비교 (계속)

위 〈그림 3〉에서 볼 수 있듯이 PSO는 전체 입자들이 국소최적해의 위치로 수렴되었으나, P2SO에서는 negative-swarm의 폭발 이후 전체최적해의 위치로 안정적으로 수렴하고 있다.

다음의 〈표 1〉은 T1~T3의 3개 함수에 대해서 초기해의 위치를 변수 범위내에서 균등하게 난수로 발생시켜 10회 반복 수행의 결과를 정리한 것이다.

〈표 1〉 Basic-PSO와 P2SO의 실험결과

Test function	Basic-PSO			P2SO		
	(x^*, y^*)		Obj^*	(x^*, y^*)		Obj^*
T1: Bimodal function	4.9126	19.9995	-1.3606	17.6770	20.0007	-13.6416
	17.6769	20.0002	-13.6416	17.6796	20.0277	-13.6408
	17.678	19.9999	-13.6416	17.6938	19.9920	-13.6411
	17.678	19.9998	-13.6416	17.6812	19.9971	-13.6416
	17.6783	19.9999	-13.6416	17.6809	20.0013	-13.6416
	4.9123	20.0000	-1.3606	17.6780	20.0009	-13.6416
	17.6781	20.0009	-13.6416	17.6798	20.0009	-13.6416
	17.6779	20.0019	-13.6416	17.6738	19.9947	-13.6415
	17.6774	19.9979	-13.6416	17.6825	19.9533	-13.6394
	4.9117	19.9983	-1.3606	17.6791	19.9964	-13.6416
T2: De Jong's function	-2.80E-06	-7.76E-06	6.80E-11	2.85E-05	2.15E-05	1.28E-09
	-3.35E-06	1.61E-05	2.71E-10	-1.37E-05	-1.84E-05	5.27E-10
	-9.93E-07	-1.44E-05	2.09E-10	-3.25E-05	1.53E-04	2.44E-08
	-7.71E-05	-3.50E-05	7.17E-09	7.43E-05	-1.30E-04	2.25E-08
	4.52E-06	-1.80E-05	3.43E-10	-4.42E-05	-5.11E-07	1.96E-09
	1.30E-05	3.19E-05	1.19E-09	3.47E-05	1.94E-04	3.90E-08
	1.78E-05	2.21E-06	3.21E-10	9.38E-07	9.60E-05	9.21E-09
	1.37E-05	7.64E-06	2.47E-10	3.23E-06	1.65E-06	1.31E-11
	-6.96E-05	3.58E-05	6.12E-09	-1.47E-05	-2.54E-05	8.63E-10
	2.01E-05	1.33E-05	5.83E-10	-4.06E-06	-5.32E-05	2.85E-09

〈표 1〉 Basic-PSO와 P2SO의 실험결과 (계속)

Test function	Basic-PSO			P2SO		
	(x^*, y^*)		Obj^*	(x^*, y^*)		Obj^*
T3: Six-hump camelback function	-1.7033	0.7957	-0.2155	-0.0898	0.7124	-1.0316
	0.0893	-0.7130	-1.0316	-0.0897	0.7127	-1.0316
	0.0898	-0.7127	-1.0316	-0.0898	0.7127	-1.0316
	0.0899	-0.7123	-1.0316	-0.0899	0.7127	-1.0316
	0.0897	-0.7127	-1.0316	-0.0899	0.7127	-1.0316
	-0.0898	0.7126	-1.0316	0.0893	-0.7125	-1.0316
	0.0898	-0.7124	-1.0316	-0.0899	0.7126	-1.0316
	0.0898	-0.7126	-1.0316	-0.0898	0.7123	-1.0316
	0.0939	-0.7128	-1.0316	-0.0898	0.7126	-1.0316
	-0.0898	0.7127	-1.0316	-0.0901	0.7125	-1.0316

〈표 1〉에서 확인할 수 있듯이 T2와 같은 단봉함수에서는 PSO나 P2SO나 모두 전체최적을 잘 찾아가고 있으나 T1과 같은 쌍봉함수의 경우 PSO는 총 10회 시행중 3번 국소최적에 빠지게 되었고, 전최최적과 국소최적 위치가 총 6개있는 T3 함수에 대해서도 1회 국소최적에 빠지고 있음을 알 수 있다. 이와 반대로 P2SO의 경우 어떠한 함수에 대해서도 모두 전체최적을 잘 찾고 있다.

V. Conclusions

본 연구에서는 해의 다양성을 유지하기 위한 방편으로 P2SO 알고리즘을 제안하였다. 기존의 PSO는 국소최적에 빠지기 쉬운 특성을 보이며 이러한 문제를 해결하기 위해 본 연구에서 제안하는 P2SO에서는 positive와 negative의 2-swarm을 운용하고, bidirectional repositioning 전략을 사용하였다. 또한, 국소최적을 탈출하고 보다 광범위한 탐색을 위해 '폭파시점'을 설정하여 폭파이후에는 negative-swarm이 방사상으로 고른 이동이 이루어질 수 있도록 하였다.

참 고 문 헌

1. Eberhart R, Kennedy J. A New Optimizer Using Particles Swarm Theory, Proc. Sixth International Symposium on Micro Machine and Human Science (Nagoya, Japan) IEEE Service Center, Piscataway, NJ:39-43, 1995
2. J. H. Holland, Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)

3. Shi YH, Eberhart RC, Fuzzy Adaptive Particle Swarm Optimization, Congress on Evolutionary Computation, 2001, pp.101-106
4. Lovbjerg M, Rasmussen TK, Krink T, Hybrid Particle Swarm Optimizer with Breeding and Subpopulation, Proceedings of Evolutionary Computation Conference, 2001
5. Ciuprina G, Ioan D, Munteanu I, Use of Intelligent-Particle Swarm Optimization in Electmmagnetics, IEEE Transactions on Magnetics, Vol.38, No.2, 2002, pp. 1037-1040