DOI: 10.3745/KIPSTB.2009.16-B.4.319

PSO의 다양한 영역 탐색과 지역적 미니멈 인식을 위한 전략

이 영 아[†] · 김 택 헌[†] · 양 성 봉[††]

요 약

PSO(Particle Swarm Optimization)는 군집(swarm)을 구성하는 단순한 개체들인 입자(particle)들이 각자의 경험을 공유하여 문제의 해답을 찾는 최적화 알고리즘으로 다양한 분야에서 응용되고 있다. PSO에 대한 연구는 최적화를 위해 군집이 적합한 영역으로 빠르게 수렴하도록 하는 파라미터 값의 선정, 토폴로지, 입자의 이동에서 주로 이루어지고 있다. 표준 PSO 알고리즘은 입자 자신과 최고의 이웃이 제공하는 정보만을 이용해서 이동하므로 다양한 영역을 탐색하지 못하고 지역적 최적점에 조기 수렴하는 경향이 있다. 본 논문에서는 군집이 다양한 영역을 탐색하기 위해, 각 입자는 더 나은 경험을 가진 이웃입자들의 정보를 상대적인 중요도에 따라서 참조하여 이동하도록 하였다. 다양한 영역의 탐색은 표준 PSO 알고리즘보다 지역적 최적화의 확률을 줄이고 탐색 속도를 가속화하며 탐색의 성공률을 높일 수 있다. 또한 군집이 지역적 미니멈으로부터 벗어나기 위한 검사 전략을 제안하여 탐색의 성공률을 높였다. 제안한 PSO 알고리즘을 평가하기 위하여, 벤치마크 함수들에 적용한 결과 최적화의 진행 속도 개선과 탐색 성공률의 향상이 있었다.

키워드 : 입자군집최적화, 상대적 가중치, 지역적 최적점

The Strategies for Exploring Various Regions and Recognizing Local Minimum of Particle Swarm Optimization

YoungAh Lee[†] · Tack-Hun Kim[†] · Sung-Bong Yang[††]

ABSTRACT

PSO(Particle Swarm Optimization) is an optimization algorithm in which simple particles search an optimal solution using shared information acquired through their own experiences. PSO applications are so numerous and diverse. Lots of researches have been made mainly on the parameter settings, topology, particle's movement in order to achieve fast convergence to proper regions of search space for optimization. In standard PSO, since each particle uses only information of its and best neighbor, swarm does not explore diverse regions and intended to premature to local optima. In this paper, we propose a new particle's movement strategy in order to explore diverse regions of search space. The strategy is that each particle moves according to relative weights of several better neighbors. The strategy of exploring diverse regions is effective and produces less local optimizations and accelerating of the optimization speed and higher success rates than standard PSO. Also, in order to raise success rates, we propose a strategy for checking whether swarm falls into local optimum. The new PSO algorithm with these two strategies shows the improvement in the search speed and success rate in the test of benchmark functions.

Keywords : PSO(Particle Swarm Optimization), Relative Weight, Local Optimum

1. 소 개

PSO 알고리즘은 1995년에 사회 심리학자 Kennedy와 전기 기술자 Eberhart에 의하여 제안된 최적화 알고리즘[1]으로 사회적 상호작용을 통하여 계산적인 지능(computational intelligence) 을 만드는 것이 목표이다. PSO 알고리즘이 처음 소개된 이후 표준 알고리즘에 많은 변화와 성능향상이 있었고 이미지/비디오 분석, 제어 문제, 스케쥴링, 통신망 디자인 및 최적화, 데이터 마이닝, 예측, 오류의 진단과 복구, 센서 네트워크, 게임과 같은 다양한 분야에서 응용되고 있다[2]. PSO 알고리즘에 대한 주요 연구는 최적화 성능에 영향을 미치는 파라미터, 토폴로지, 입자의 이동 등에 관한 것이다[3].

※ 이 논문은 교육인적자원부지원 연세대학교 BK21 지능형 모바일 서비스를 위한 차세대 단말 소프트웨어 사업단의 지원을 받아 연구되었음.
† 정 회 원 : 연세대학교 컴퓨터과학과 BK21 연구교수
†† 종신회원 : 연세대학교 컴퓨터과학과 교수

질 수 있다. 표준 PSO 알고리즘은 *gbest* 와 *lbest*라는 정보 공유 방식을 가지고 있다. *gbest*는 군집 전체가 이웃이고, 군집에서 최고 이웃을 선정한다. *gbest*는 군집이 빠르게 좋은 지역으로 수렴하지만 지역적 미니멈에 빠지기 쉽다. *lbest*는 입자들의 거리, 토폴로지에 따라서 또는 임의로 이웃들을 선정하고, 입자 자신의 정보와 가장 좋은 이웃의 정보로부터 영향을 받아 이동한다. *lbest*는 탐색 공간의 다양한 영역에서 수렴할 수 있고 *gbest*보다 지역적 최적화에 빠질 확률이 낮아진다.

표준 PSO 알고리즘은 지역적 최적화에 대한 처리 방안을 가지고 있지 않다. 표준 PSO 알고리즘은 *lbest*를 통해 지역적 최적화의 확률을 낮추고 있지만, 복잡한 문제에 적용하여 항상 좋은 성과를 얻지 못한다. 그 이유는 *lbest*도 입자 자신과 최고 이웃으로부터만 영향을 받으므로 최고 이웃이 지역적 최적점이라면 자신과 이웃들은 빠르게 최고 이웃으로 접근하고, 새롭게 이웃을 선정하면 다른 입자들도 영향을 받아 지역적 최적점 주위로 모이게 된다. 이러한 이동이 계속되면 군집은 지역적 최적점에 빠져서 탐색 시간이 증가하고 탐색 성공률이 낮아진다. PSO 알고리즘은 이러한 상황을 발견하고 수축되어 있는 군집이 다른 지역을 탐색할 수 있도록 팽창시켜야 한다.

본 논문에서는 PSO의 최적화 과정의 탐색 속도를 개선하고 전역적 미니멈(global minimum)을 찾는데 있어 성공률을 향상시키기 위해 다수의 이웃 입자들과의 상대적 가중치를 이용한 정보 공유 방법과 지역적 미니멈(local minimum)의 검사에 대한 두 전략을 제안한다. 제안 PSO는 입자 이동시에 입자 자신과 자신보다 좋은 경험을 한 여러 이웃 입자들의 정보를 확률적으로 받아들이므로 다양한 지역의 탐색으로 전역적 미니멈을 찾을 수 있다. 또한 제안 PSO는 지역적 미니멈을 검사하고 군집을 초기화하는 전략을 통해 탐색 속도와 성공률의 향상을 얻을 수 있다. 본 논문에서는 다양한 벤치마크 함수들[7,8]을 이용하여 제안 PSO와 표준 PSO 알고리즘(standard PSO)[6]을 비교하였다. 실험 결과에서 제안 PSO알고리즘이 표준 PSO알고리즘에 비하여 탐색 속도와 성공률에 큰 향상이 있었다. 본 논문의 구성은 다음과 같다. 2절에서는 PSO의 표준 알고리즘과 관련연구에 대하여 기술하고, 3절에서는 제안한 두 전략을 소개한다. 4절에서는 제안한 PSO 알고리즘을 다양한 벤치마크 함수에 적용한 실험 결과를 요약하고, 5절에서 결론을 맺는다.

2. PSO 알고리즘

PSO에서 군집(Swarm)의 각 입자(particle)는 작은 메모리를 가지고 있어서 해 공간의 탐색을 통해 구한 자신의 최적의 위치를 기억한다. 그리고 다수의 입자들이 서로 정보를 공유하면서 해를 찾는 사회적인 행동을 기반으로 한다. 유전자 알고리즘과 PSO 알고리즘은 군집을 바탕으로 최적화를 수행한다는 공통점이 있다. 그러나 유전자 알고리즘은 PSO 알고리즘과 달리 교차(crossover)와 돌연변이(mutation) 연산자를 가지고 있어서, 선택된 두 염색체의 일부를 서로 교환하고 새로운 형질을 가진 염색체를 생성하는 과정에서 해를 찾는다. PSO 알고리즘은 유전자 알고리즘과 같은 연산자를 가지고 있지 않고, 이웃들의 최적점 정보를 바탕으로 해에서 다른 해로 이동하며 최적의 해를 찾는다.

표준 PSO 알고리즘의 최적화 과정[1,3,6]은 다음과 같다. 단순한 개체들(entities)인 입자들은 주어진 문제의 다차원 탐색 공간에 놓여지고, 각 입자는 현재 위치를 적합치 함수(fitness function)를 이용하여 평가한다. 각 입자 i는 이웃들을 선정하고, 현재 자신이 알고 있는 최적 위치와 적합치 대한 정보($\overrightarrow{P_i}$) 및 최고 이웃이 알고 있는 최적의 위치와 적합치 정보 ($\overrightarrow{P_{lbest}}$)를 조합하여 탐색공간에서 이동하게 된다. 모든 입자들이 이동한 후, 군집의 에러($\overrightarrow{P_{gbest.fitness}}$)가 허용치 이하가 되거나 위치평가의 횟수가 최대 반복횟수가 될 때까지 군집의 이동은 반복된다. 결과적으로 군집 전체는 해결하려는 문제의 해의 주변으로 수축하게 된다.

PSO 알고리즘의 성능향상을 위한 주요 연구는 군집의 토폴로지, 입자의 이동과 알고리즘 파라미터들을 중심으로 이루어졌다[2]. 입자는 이동하기 위해 이웃의 정보를 받아들여 이동 방향과 이동량인 속도(velocity)를 계산하는데, 어떠한 이웃들로부터 어떻게 영향을 받는가에 따라서 속도가 달라져 탐색하는 공간도 다르게 된다. 표준 PSO 알고리즘은 군집의 이웃들 중에서 가장 좋은 적합치를 가진 입자의 정보만을 이용하였고, Mendes가 제안한 FIPS(Fully Informed Particle Swarm)[3]는 군집 전체가 이웃이 되고 모든 이웃들로부터 정보를 받아 이동한다. FIPS는 토폴로지에 크게 영향을 받지만 표준 알고리즘에 비하여 보다 적은 수의 반복으로 보다 나은 결과를 얻었다.

다음 (식 1)은 Shi 와 Eberhart(1998)가 입자의 이동을 보다 안정적으로 제어하기 위해서 제안한 속도를 업데이트 하는 식이다[2,3].

$$\vec{v_i} \leftarrow \omega\vec{v_i} + \vec{U}(0,\varphi 1) \times (\vec{P_i} - \vec{X_i}) + \vec{U}(0,\varphi 2) \times (\overrightarrow{P_{lbest}} - \vec{X_i}) \quad \text{(식 1)}$$

$\vec{v_i}$: 입자 i의 속도 벡터, $\vec{X_i}$: 입자 i의 현재 위치 벡터
$\vec{P_i}$: particle i 의 최적점, $\overrightarrow{P_{lbest}}$: 최고 이웃의 최적점
ω : 관성 가중치(inertia weight), $\varphi 1, \varphi 2$: 가속 계수(acceleration coefficients)

입자는 탐색 초기에는 탐색 공간을 크게 이동하다가 최적의 위치에 가깝다고 판단되면 속도를 줄여야 하는데, (식 1)의 파라미터 ω(inertia weight)가 이를 제어한다. 파라미터 ω는 초기에 큰 값으로부터 시작해서 점차 작은 값으로 줄여 나가면 입자는 탐험(exploration)에서 점차 세심한 탐색(exploitation)을 하게 된다.

Clerc과 Kennedy(2002)는 입자의 수렴을 보장하고 이동 속도를 제어하기 위해서 알고리즘의 파라미터들을 분석하여 다음 (식 2), (식 3), (식 4)의 파라미터 수축 방법(constriction method)[3-5]을 제안하였다

$$\vec{v_i} \leftarrow \chi\left(\vec{v_i} + \vec{U}(0,\varphi 1) \times (\vec{P_i} - \vec{X_i}) + \vec{U}(0,\varphi 2) \times (\overrightarrow{P_{lbest}} - \vec{X_i})\right) \text{(식 2)}$$

$$\overrightarrow{X_i} \leftarrow \overrightarrow{X_i} + \overrightarrow{v_i} \quad \text{(식 3)}$$

$$\chi = \frac{2}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}}, \ \varphi = \varphi 1 + \varphi 2 > 4 \quad \text{(식 4)}$$

(식 2)에서 파라미터 $\overrightarrow{U}(0,\varphi 1), \overrightarrow{U}(0,\varphi 2)$ 는 각각 $\overrightarrow{P_i}$와 $\overrightarrow{P_{lbest}}$에 대한 랜덤 세기로 입자 i가 이동시에 받게될 정보의 양을 결정한다. 입자의 행동은 가속 계수(acceleration coefficients)라고도 불리는 이 두 값의 변화에 따라서 속도가 매우 커져 불안정할 수 있다. 그러므로 입자의 속도를 제어하기 위해서 최대 속도 V_{max}를 정해야 하지만 문제에 적합한 V_{max}를 정하기는 어렵다. Clerc의 파라미터 수축 방법(constriction method)은 φ=4.1, φ1= φ2 로 정하였는데, 식 (2)와 식 (4)에 φ1, φ2을 대입하면 χ은 약 0.7298이 되고, 결과적으로 (식 2)의 두 $(\overrightarrow{P} - \overrightarrow{X})$에 최대 0.7298 X 2.05 ≒ 1.49618이 곱해진다. 위의 방법으로 임의로 최대 속도 V_{max}를 정할 필요가 없고, 입자는 수렴이 보장된다[3].

표준 PSO 알고리즘은 지역적 최적점에 빠르게 수렴하려는 경향이 있다. 입자들은 최적점을 향해 이동하는 다른 입자들의 영향을 받아 이동하므로 군집은 최적점 주위로 수축하게 된다. 만일 발견한 해가 전역적 최적점이라면 계속해서 최적점 주변을 탐색(exploitation)하면서 정밀하게 해를 찾아야 한다. 하지만 만일 발견한 해가 지역적 최적점이라면, 군집은 최적점 주위로 수축을 중단하고 다른 지역을 탐험(exploration)해야 한다. PSO 알고리즘이 멀티모달 함수와 같은 복잡한 문제를 최적화해야 한다면 다양한 영역의 탐험(exploration)이 더욱 중요하다. 이와 관련된 연구들은 다음과 같다. Loøvbjerg는 두 입자의 거리가 매우 가깝다면 위기값(critical value)을 증가시키고, 만일 위기값이 임계값(threshold)에 도달하면 다른 입자들에게 그 사실을 알리고 해당 입자를 다른 위치에 재배치하여 다양성을 얻으려 하였다.[3] Klinks는 공간적으로 확장하는 입자들을 개발하였다. 각 입자는 구에 의해 둘러싸여 있고 팽창하는 입자가 다른 입자와 충돌하면 튀어 오르게 하였다.[3] Xie는 지역적 최적점을 향해 조기 수렴하는 것을 막기 위해 군집에 부정적인 엔트로피를 사용하였다.[3]

3. 제안하는 PSO 알고리즘

본 논문에서는 PSO 알고리즘의 최적화 속도와 탐색 성공률을 개선하기 위한 두 전략을 제안한다. 첫번째 전략은 입자가 속도 업데이트시에 이웃들의 경험을 상대적 가중치에 따라 받아들여 상태공간의 다양한 영역을 탐색하기 위한 것이다. 표준 PSO 알고리즘(standard PSO 2006)[1,3,6]은 (식 1)과 (식 2)와 같이 이웃 입자 중에서 가장 낮은 적합치(fitness value)를 가진 입자 *lbest* 의 경험($\overrightarrow{P_{lbest}}$)을 이용했다. 표준 알고리즘에서는 이동하려는 입자 i가 지역적 미니멈에 가깝게 놓인 입자로부터 직접적으로 영향을 받으므로[3], 입자 i를 이웃으로 갖는 다른 입자들도 지역적 미니멈에 가까이 이동할 가능성이 높아진다. 입자들이 계속해서 이러한 이동을 반복적으로 하게 되면, 지역적 미니멈에 가까운 입자들의 개수가 증가하고 이들이 입자 *lbest*가 될 가능성이 높아진다. 결국 군집은 지역적 미니멈에 빠질 수 있다. 제안한 전략은 이동하려는 입자 i가 자신보다 적합치가 낮은 이웃 입자들의 탐색 정보를 모두 취합하는데, 각 이웃 입자의 적합치에 따른 상대적 가중치에 비례하여 받아들인다. 입자 i에 대한 이웃 입자 j의 상대적 가중치(relative weight, rw_{ij})는 (식 5)와 같이 구한다. 상대적 가중치의 합은 1이 만족되어야 한다.

$$rw_{ij} = \frac{1}{\sum_{k=1}^{n}\left(\frac{\vec{P}_{j.fitness}}{\vec{P}_{k.fitness}}\right)^2}, \qquad \sum_{j=1}^{n} rw_{ij} = 1 \quad \text{(식 5)}$$

$\vec{P}_{k.fitness}$: *입자 k의 최저 적합치,*
n : $\vec{P}_{i.fitness} \geq \vec{P}_{j.fitness}$ *를 만족하는 이웃의 개수*

(식 5)를 통해 계산되는 상대적 가중치는 이웃 입자 k들이 경험한 최저의 적합치 $\vec{P}_{k.fitness}$에 반비례한 값으로 (식 6)의 입자 속도 업데이트에 이용된다. (식 6)의 랜덤 가속 계수 $\vec{U}(0,\varphi 2)$는 이웃 입자들의 상대적 중요도를 변화시킬 수 있다. 상대적 가중치의 변화는 입자의 이동에 임의성을 부여해서 항상 좋은 입자들을 향해 이동할 경우의 수를 줄여 더욱 다양한 영역의 탐색이 가능하게 한다.

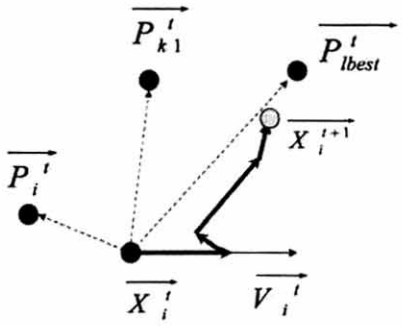
$$\vec{v}_i \leftarrow \omega\vec{v}_i + rw_{ii} \times \vec{U}(0,\varphi 1) \times (\vec{P}_i - \vec{x}_i) + \sum_{j=1}^{n}(rw_{ij} \times \vec{U}(0,\varphi 2) \times (\vec{P}_j - \vec{x}_i)), \ j = 1..n(i \neq j) \quad \text{(식 6)}$$

n : *입자 i의 이웃의 수*
rw_{ij} : *적합치 기준의 이웃 입자 j의 상대적 가중치*
rw_{ii}: *입자 i 자신이 가진 정보의 상대적 가중치,* $rw_{ii} = 1, i = 1 \sim S$
$\vec{P}_i$: *particle i 의 최적점,* $\vec{P}_j$: $\vec{P}_{i.fitness} \geq \vec{P}_{j.fitness}$ *를 만족하는 이웃*

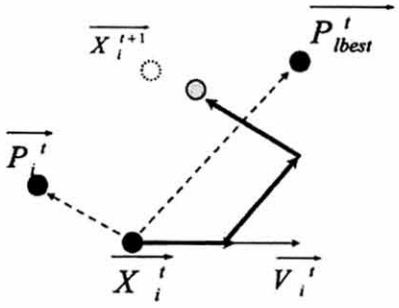
새로운 업데이트 식에 의해 입자 i는 입자 *lbest*의 경험($\overrightarrow{P_{lbest}}$)뿐만 아니라 다른 이웃들의 경험도 공유하여 이동 방향과 이동량을 결정하게 되므로, (그림 1)과 같이 표준 알고리즘의 입자와는 다른 경로로 탐색을 하게 된다. (그림 1)에서, t시점에서의 입자 i는 위치 $\overrightarrow{x_i^t}$에서 자신의 속도 $\overrightarrow{v_i^t}$와 최고 이웃의 최적의 위치($\overrightarrow{P_{lbest}^t}$), 이웃들이 알고 있는 최적의 위치 $\overrightarrow{P_k^t}$를 바탕으로 새로운 위치 $\overrightarrow{x_i^{t+1}}$로 이동한다.

두번째 전략은 군집이 지역적 미니멈에 빠졌는가를 검사하기 위한 임계값(threshold)을 지정하는 방법이다. 전역적 미니멈은 알려져 있거나 임의로 원하는 값을 정할 수 있으나 지역적 미니멈은 일반적으로 알려지지 않기 때문에 지역적 미니멈에 대한 임계값을 정하기 어렵다.

군집이 지역적 미니멈에 빠지게 되면 이후 이동을 해도 지역적 미니멈 주위를 맴돌며 군집의 크기는 계속해서 수축하게 된다. 본 논문에서는 군집이 매우 수축되어 있지만 계속해서 전역적 최적점을 찾지 못한다면 지역적 미니멈에 빠



(그림 1) (1) 제안한 PSO의 위치 업데이트



(2) 표준 PSO의 위치 업데이트

졌다고 판단한다. 그리고 군집의 수축 상태를 의미하는 수치로서 군집의 현재 최저 적합치(error, $\overrightarrow{P^t_{gbest.fitness}}$)와 군집에서 임의로 선택한 입자와 군집의 전역적 최적점($\overrightarrow{P^t_{gbest}}$)사이의 거리(diff)를 이용한다. 군집의 현재 최저 적합치인 에러(error)는 최적화가 진행되면서 지역적 미니멈 또는 전역적 미니멈에 가깝게 감소하는 값이다. 또한 거리(diff)도 군집이 수축하면서 점점 감소한다. 이 두 값은 군집의 최적화 정도를 의미하고 정확한 임계값을 대신하여 검사에 이용될 수 있다. 만일 군집이 전역적 최적점($\overrightarrow{P^t_{gbest}}$)이 중심이고 두 척도가 반지름인 구 안에 존재한다면 지역적 미니멈에 빠졌을 가능성이 높다고 보고, 군집을 초기화하여 지역적 미니멈으로부터 벗어나게 한다. 다음은 군집의 모든 입자들이 이동하였고 그 결과 에러가 감소하지 않았을 경우에 이루어지는 검사 과정과 사후 처리의 의사 코드이다.

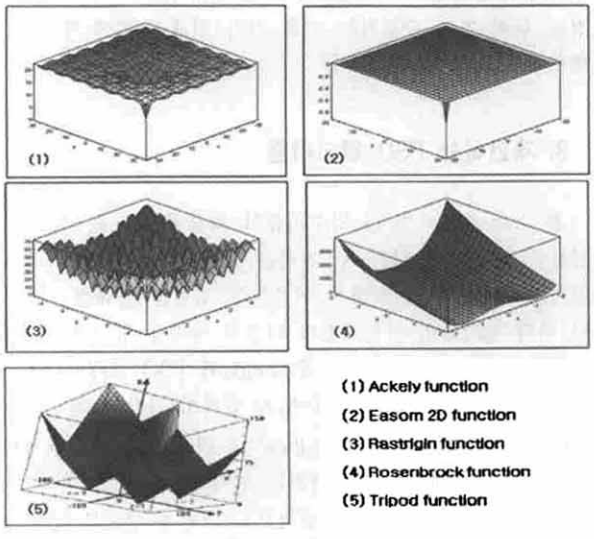
(지역적 미니멈의 검사와 사후처리)
step 1: 초기화
S = 군집의 크기; count = 0; local_minimum_flag = 0;
step 2: 군집의 너비를 정한다. 너비 'range'의 값은 t시점의 전역적 최적점의 적합치(error, $\overrightarrow{P^t_{gbest.fitness}}$)이거나 임의로 선택된 입자와 전역적 최적점(global optimum)인 $\overrightarrow{P^t_{gbest}}$ 사이의 거리 (diff)이다.
range = error; 또는 range = diff;
step 3: 각 입자의 위치 $\overrightarrow{x_i}$에 대하여 다음 조건을 검사하고, 만족하면 count를 1 증가 한다.
for (i = 0; i < S; i++)
if (0 ≤ ABS($\overrightarrow{P^t_{gbest}}$ - $\overrightarrow{x_i}$) ≤ range) count++;
step 4: if (count = S) local_minimum_flag = 1;
step 5: local minimum*에 빠졌다면 모든 입자들은 임의의 위치로 이동하고 다시 탐색을 시작한다.*

4. 실험과 분석

본 실험에서는 제안한 PSO 알고리즘의 성능을 평가하기 위해 (그림 2)와 <표 1>의 벤치마크 함수인 Ackley 함수, Easom 2D 함수, Rastrigin 함수 Rosenbrock(Banana) 함수 및 Tripod 함수[5]를 이용하여 전역적 미니멈을 찾는 문제에 적용하였다[7,8]. 4차원 이상의 Rosenbrock 함수는 적어도 하나이상의 지역적 미니멈을 갖는 멀티모달 함수이고, Easom 2D 함수는 유니모달 함수로 지역적 미니멈이 없고 하나의 전역적 미니멈을 갖는다. Ackley와 Rastrigin 함수는 하나의 전역적 미니멈과 다수의 지역적 미니멈을 가지는 멀티모달 함수이다. Tripod 함수는 두개의 깊은 지역적 미니멈과 하나의 전역적 미니멈을 가지고 있다.

본 실험에서는 군집의 크기(swarm size)와 이웃 입자의 수는 각각 12와 3으로 정하였다. (식 6)에서의 가속 계수 φ1,φ2는 표준 PSO와 같이 0.5+log(2) ≈ 1.1931로 하거나, 보다 다양한 영역을 탐색해야 하는 경우에 입자의 움직임에 큰 임의성을 주기 위해서 φ1는 2.0과 φ2는 2.1, 또는 각각 2.05와 2.05로 하였다. 위의 군집의 크기, 이웃 입자의 수, 가속 계수의 수치는 실험 결과와 파라미터 수축방법(constriction method)[3-5]을 바탕으로 정하였다.

본 실험에서 알고리즘의 탐색 속도 향상은 평가횟수를 이용하여 평가한다. 평가횟수는 군집의 입자들이 적합치 함수(fitness)를 평가한 총 횟수로서 입자들이 최적해를 찾기 위해 탐색 공간을 이동한 횟수를 의미한다. 제안 알고리즘은 입자들이 적합치가 낮은 영역뿐만 아니라 보다 넓은 영역을 탐색하므로 지역적 미니멈에 빠질 확률을 줄이고, 지역적



(그림 2) 실험에 사용한 벤치마크 함수들

〈표 1〉 실험에 사용된 2차원 벤치 마크 함수들과 파라미터들

함수	식	테스트 범위, 글로벌 미니멈
Ackley	$f(x) = -a \cdot \exp\left(-b \cdot \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(cx_i)\right) + a + \exp(1)$	$a = 20, b = 0.2,$ $c = 2\pi$ $-32.768 \le x_i \le 32.768$
		$f(0,0) = 0$
Easom 2D	$f(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2 - (x_2-\pi)^2)$	$-100 \le x_i \le 100$
		$f(\pi, \pi) = -1$
Rastrigin	$f(x) = 10n + \sum_{i=1}^{2}[x_i^2 - 10\cos(2\pi x_i)]$	$-5.12 \le x_i \le 5.12$
		$f(0,0) = 0$
Rosenbrock	$f(x) = \sum_{i=1}^{2}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	$-2.048 \le x_i \le 2.048$
		$f(1,1) = 0$
Tripod	if $(x_2 < 0)$ $f(x_1, x_2) = \lvert x_1\rvert + \lvert x_2 + 50\rvert$ else if$(x_1 < 0)$ $f(x_1 + x_2) = 1 + \lvert x_1 + 50\rvert + \lvert x_2 - 50\rvert$ else $f(x_1 + x_2) = 2 + \lvert x_1 - 50\rvert + \lvert x_2 - 50\rvert$	$-100 \le x_i \le 100$
		$f(0, -50) = 0$

〈표 2〉 표준 PSO 알고리즘과 제안 알고리즘의 실행시간과 평가횟수의 비교

알고리즘 / 함수	표준 PSO 알고리즘		제안 알고리즘		허용에러
	실행시간(초)	평가횟수	실행시간(초)	평가횟수	
Ackley	0.0343	889	0.0062	799	10^{-3}
Easom 2D	0.3453	8,614	0.0047	1,186	10^{-3}
Rastrigin	0.0312	622	0.0031	1,057	10^{-3}
Rosenbrock	0.0344	1,042	0.0015	839	10^{-3}
Tripod	1.4685	28,954	0.0311	4,648	10^{-2}

미니멈에 빠졌어도 제안한 두번째 전략에 의해 지역적 미니멈에 계속해서 머무르지 않고 다른 곳을 탐색할 수 있다. 그러므로 제안 알고리즘은 표준 PSO 알고리즘에 비하여 평가 횟수가 작다. 제안 알고리즘은 상대적 가중치 계산과 지역적 미니멈 검사의 추가로 평가횟수당 소모시간은 증가하지만 평가횟수가 감소하므로 알고리즘 실행시간 또한 표준 PSO 알고리즘과 비교하여 작다. <표 2>는 표준 PSO알고리즘과 제안 알고리즘이 전역적 미니멈을 발견하기까지의 실행시간과 평가횟수를 비교한 표이다. <표 2>의 실행시간과 평가횟수는 각 함수를 반복 실행하여 나온 총 실행시간과 총 평가횟수를 반복 횟수로 나눈 값이다. 실험에서 군집의 크기는 12이고 이웃 입자의 개수는 3으로 정하였다. <표 2>의 실험 결과를 바탕으로 제안 알고리즘의 탐색 속도 향상의 평가 척도로서 평가 횟수를 사용한다.

제안 알고리즘은 (step 2)~(step 3)의 지역적 미니멈 검사에서 검사척도로서 $\overrightarrow{P^t_{gbest.fitness}}$ (error) 또는 임의로 선택된 입자와 $\overrightarrow{P^t_{gbest}}$의 거리(diff)를 사용하였다. <표 3>의 각 열은 두 척도를 각각Tripod 함수에 적용하여 발견한 지역적 미니멈의 두 예를 나타낸다. 그리고 <표 3>은 지역적 미니멈에 빠진 상황에서의 평가횟수, $\overrightarrow{P^t_{gbest}}$위치와 적합치(error), 군집에 속한 각 입자(s0~s11)의 위치를 보여준다. Tripod 함수의 전역적 미니멈 위치는 (0, -50)이고 이 위치의 적합치는 0이다. 두 지역적 미니마의 위치와 적합치는 (-50, 50)과 1이고 (50, 50) 과 2이다.

<표 3>와 같이 군집이 지역적 미니멈에 빠지게 되면, $\overrightarrow{P^t_{gbest}}$*의 위치* 는 지역적 미니멈에 매우 가깝고 군집은 $\overrightarrow{P^t_{gbest}}$*의 위치*를 중심으로 수축한 상태이므로 $\overrightarrow{P^t_{gbest}}$*의 위치*와 모든 입자들(s0~s11) 사이의 거리는 매우 가깝다. 그리고 입자들(s0~s11)의 적합치(fitness)는 지역적 미니멈의 적합치와 큰 차이가 없어진다. 이 상태에서 군집이 다시 이동을 한다면 $\overrightarrow{P^t_{gbest}}$*의 위치* 주위로 더욱 수축하고 계속 지역적 미니멈에 빠진 상태가 된다.

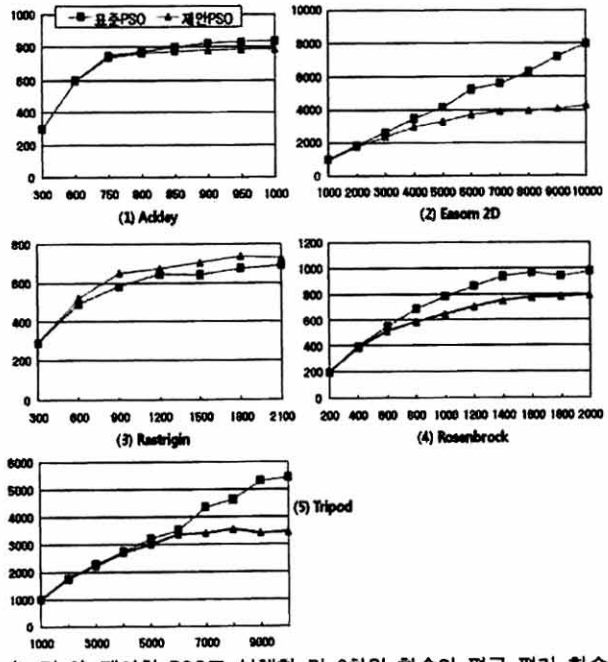
두 검사 척도에 대한 비교 실험에서, 멀티모달 함수인 Rastrigin 함수와 Tripod 함수는 두 척도에 대하여 비슷한 평가 횟수와 성공률(95%이상)을 보였고, Ackley, Easom 2D, Rosenbrock 함수는 $\overrightarrow{P^t_{gbest}}$의 적합치(error)가 더 나은 결과를 보였다.

(그림 3)은 표준 PSO 알고리즘과 제안한 PSO를 500회 실행하여 얻은 각 함수의 평균 평가 횟수를 비교한 것이다. 평가 횟수는 군집의 입자들이 전역적 미니멈을 찾기 위해 이동한 횟수로 알고리즘의 탐색 속도를 평가하는 척도로 사용된다. 제안한 PSO알고리즘은 Rastrigin 함수를 제외한 Ackley, Easom 2D, Rosenbrock, Tripod 함수의 실험에서 표준 PSO와 비교하여 평균 평가 횟수가 감소하였다. 이 결과는 표준 알고리즘에 비하여 적은 수의 이동으로 전역적 미니멈에 도달하였음을 의미한다. 본 실험에서, Rastrigin함수는 평가 횟수와 성공률을 높이기 위해 다른 함수와는 달리 (식 6)의 가속 계수 $\varphi 1, \varphi 2$를 차례로 2.0, 2.1로 정하여서 이동 방향과 이동거리에 임의성을 크게 주었다.

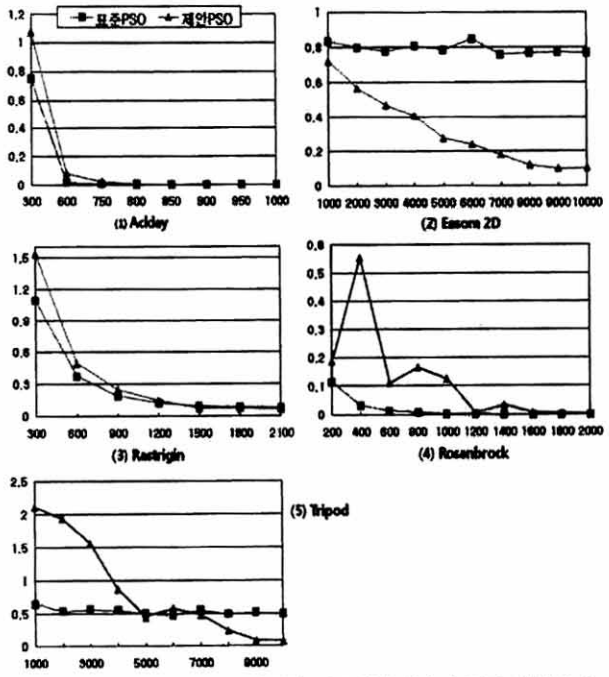
(그림 4)는 제안한 PSO알고리즘과 표준 PSO 알고리즘의

〈표 3〉 Tripod함수의 지역적 미니멈의 발견 예. (f는 적합치 함수이다.)

$\overrightarrow{P^t_{gbest.fitness}}$ (error를 의미함)	임의로 선택된 입자와 $\overrightarrow{P^t_{gbest}}$의 거리 (diff를 의미함)
평가횟수: 972, error = 2.000158 $\overrightarrow{P^t_{gbest}}$*의 위치 = (49.999951, 49.999890)*	*평가횟수=8,964, error = 2.000107* $\overrightarrow{P^t_{gbest}}$*의 위치 = (50.000082, 49.999975)* *diff = (0.180943, 0.013787)*
s0 : 50.000684 49.999913 fitness : 2.000771	s0 : 49.999561 50.000800 fitness : 2.001239
s1 : 49.999951 50.000511 fitness : 2.000559	s1 : 50.181025 49.986188 fitness : 2.194837
s2 : 48.938410 49.999228 fitness : 3.062362	s2 : 50.000395 50.003336 fitness : 2.003731
s3 : 50.000886 49.998582 fitness : 2.002304	s3 : 49.998244 49.999573 fitness : 2.002184
s4 : 49.961805 50.153131 fitness : 2.191326	s4 : 50.002145 49.999543 fitness : 2.002602
s5 : 50.002845 49.999519 fitness : 2.003326	s5 : 49.949127 50.000473 fitness : 2.051346
s6 : 50.002014 49.998882 fitness : 2.003132	s6 : 50.005400 49.999407 fitness : 2.005993
s7 : 49.969568 49.999623 fitness : 2.030808	s7 : 50.001890 50.000863 fitness : 2.002753
s8 : 50.000106 49.999060 fitness : 2.001046	s8 : 50.000359 49.994055 fitness : 2.006304
s9 : 50.002328 49.999571 fitness : 2.002757	s9 : 49.958022 50.004925 fitness : 2.046902
s10 : 49.975648 50.000938 fitness : 2.025290	s10 : 50.000361 50.001038 fitness : 2.001398
s11 : 49.543104 50.041249 fitness : 2.498145	s11 : 50.002385 49.999723 fitness : 2.002662
평가횟수= 1,440, error = 1.044126 $\overrightarrow{P^t_{gbest}}$*의 위치 = (-50.035482, 49.991356)*	*평가횟수=11,832, error = 1.552109* $\overrightarrow{P^t_{gbest}}$*의 위치 = (-49.685654, 50.237763)* *diff = (4.258192, 6.137622)*
s0 : -49.923149 49.995456 fitness : 1.081395	s0 : -50.569658 50.755297 fitness : 2.324955
s1 : -49.760446 49.955218 fitness : 1.284336	s1 : -46.766793 50.477193 fitness : 4.710400
s2 : -50.003338 50.252784 fitness : 1.256122	s2 : -50.903627 52.048017 fitness : 3.951644
s3 : -49.753793 50.793959 fitness : 2.040166	s3 : -47.389993 51.155267 fitness : 4.765274
s4 : -50.078424 50.117885 fitness : 1.196309	s4 : -52.026610 49.213315 fitness : 3.813295
s5 : -49.714191 50.408892 fitness : 1.694701	s5 : -51.510682 49.019509 fitness : 3.491173
s6 : -49.738914 50.262246 fitness : 1.523332	s6 : -53.203404 52.749236 fitness : 6.952639
s7 : -49.937439 49.897347 fitness : 1.165214	s7 : -53.871435 49.778159 fitness : 5.093276
s8 : -50.354504 49.996360 fitness : 1.358144	s8 : -50.788169 51.052448 fitness : 2.840617
s9 : -50.585400 50.016974 fitness : 1.602374	s9 : -50.146448 49.303204 fitness : 1.843244
s10 : -49.772378 50.081877 fitness : 1.309499	s10 : -53.943845 56.375385 fitness : 11.319231
s11 : -49.804341 50.250449 fitness : 1.446108	s11 : -50.328344 54.588417 fitness : 5.916761



(그림 3) 제안한 PSO로 실행한 각 2차원 함수의 평균 평가 횟수 (x축: 최대 평가횟수, y축: 전역적 미니멈 발견 평균 평가 횟수)

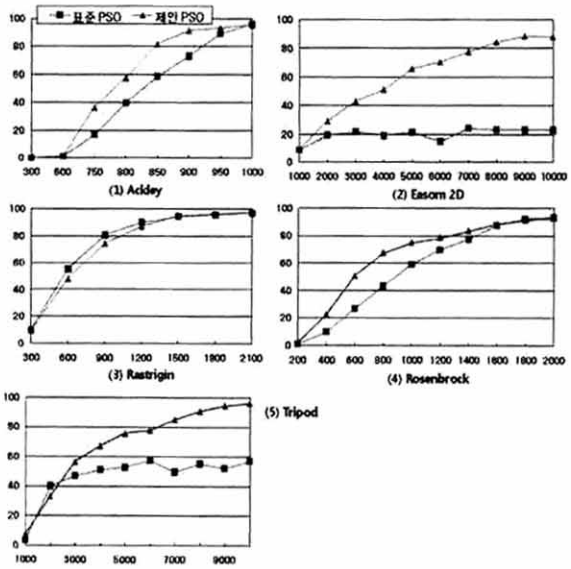


(그림 4) 제안한 PSO에서 실행한 각 2차원 함수의 평균 에러 (x축: 최대 평가횟수, y축: 전역적 미니멈 발견 평균 에러)

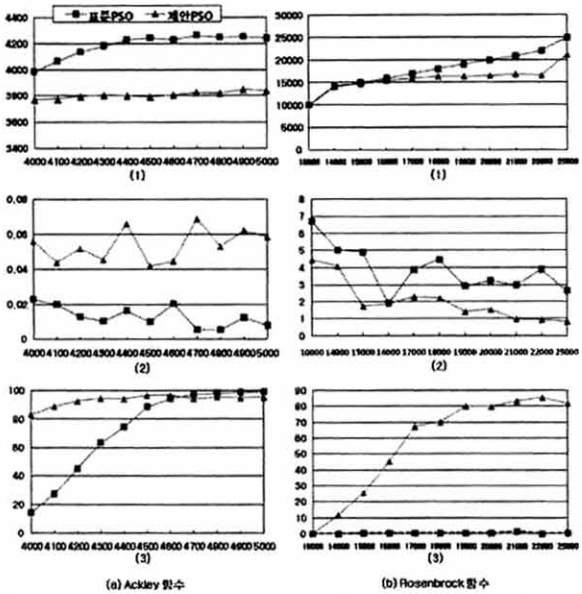
각 함수에 대한 평균 에러를 비교한 그래프들이다. 유니 모달 함수인 Easom 2D 함수는 탐색 초기부터 빠르게 에러가 감소한다. 그 밖의 함수들은 탐색 초기에는 제안한 PSO알고리즘의 에러가 표준 알고리즘에 비하여 크지만 탐색 중반 이후로는 표준 알고리즘의 에러 이하로 감소한다. 이러한 양상의 원인은 다음과 같다. 표준 알고리즘은 이웃들 중에서 최소 적합치를 가진 이웃의 정보 $\overrightarrow{p_{gbest}^{t}}$만을 참조해서 직접적으로 최소 적합치를 가진 이웃을 향해 이동하므로 탐색 초기부터 적합치 즉 에러가 크게 감소할 수 있지만 점차 부분적인 지역 탐색을 하므로 에러의 감소가 둔화된다. 이에 반해, 제안한 PSO알고리즘은 적합치가 낮은 다수의 이웃들의 정보를 바탕으로 이동을 하므로 최소 적합치의 이웃에 간접적으로 이동을 하게 되어 탐색 중반까지는 에러가 크지만, 보다 넓은 탐색 공간을 탐색하여 적합치가 낮은 곳을 찾게 되므로 점차 에러는 감소하게 된다.

(그림 5)는 두 알고리즘을 각 함수에 500번 적용한 후 계산된 각 함수의 평균 성공률을 보여준다. Ackley, Easom 2D, Rosenbrock 및 Tripod 함수는 표준 알고리즘과 비교하여 성공률이 빠르게 증가한다. Rastrigin함수는 평가횟수 1500회 이상부터 표준 PSO의 성공률과 비교하여 작은 향상이 있었다.

(그림 6)은 10차원 Ackley함수와 10차원 Rosenbrock 함수를 제안 PSO 알고리즘에 적용한 실험 결과이다. 실험에서 두 함수 모두 탐색 초기부터 적은 횟수의 이동으로 성공적으로 전역적 미니멈을 발견하는 것을 보였다. 10차원 Ackley 함수의 탐색 공간은 광대하고 많은 지역적 미니멈이 존재하여 매우 복잡한데, 이러한 공간에서 여러 이웃들의 정보를 바탕으로 이동하면 허용 에러 이하는 만족하지만 큰 적합치를 갖는 전역적 미니멈을 발견하게 되어 그래프와 같이 평균 에러가 크다.



(그림 5) 제안한 PSO에서 실행한 각 2차원 함수의 평균 성공률 (x축: 최대 평가 횟수, y축: 전역적 미니멈 발견 평균 성공률)



(그림 6) 10차원 함수의 실험 결과 (1) 평균 평가 횟수 (2) 평균 에러 (3) 평균 성공률

5. 결론과 향후 연구 과제

본 논문에서는 표준 PSO 알고리즘의 최적화 속도 개선과 성공률 향상을 위한 두 전략을 제안하였다. 제안한 알고리즘은 입자가 자신보다 적합치가 낮은 다수의 이웃들의 정보를 공유하고 이웃들의 상대적 중요도를 바탕으로 정보를 받아들여 이동을 하므로 적합치가 가장 낮은 이웃을 향해 직접적으로 이동을 하지 않고 보다 넓게 탐색을 하게 된다. 또한 제안한 알고리즘에서는 최적화의 진행 상태를 의미하는 수치인 군집의 너비를 이용하여 군집이 지역적 미니멈에서 빠져 나오지 못하는 상황인가를 검사하는 전략을 사용하였다. 이 두 전략을 다양한 척도 함수에 적용한 실험 결과, 표준 알고리즘에 비하여 적은 수의 이동을 통해서 성공률이 빠르게 증가함을 알 수 있었다.

향후에는 탐색 초기에는 다수의 이웃들의 정보를 받아들여 다양한 공간을 탐험하도록 하고 점차 입자 자신과 군집의 최적 정보의 중요도를 높이기 위한 관성 가중치(inertia weight)와 가속 계수의 조정에 대하여 연구할 계획이다. 또한 제안 PSO 알고리즘과 신경망 알고리즘의 비교를 하겠다. 향후 연구를 통해, 군집이 전역 미니멈에 가까워질수록 탐색 속도의 가속화를 얻을 수 있으리라 기대한다.

참 고 문 헌

[1] Kennedy J, Eberhart R.C., "Particle Swarm Optimization", Proceedings of IEEE International Conference on Neural Networks, p.1942-1948, 1995.

[2] Shi Y.H, Eberhart R.C., "A Modified Particle Swarm Optimizer", Proceedings of IEEE International Conference on Evolutionary Computation, 1998.

[3] Riccardo Poli, James Kennedy, Tim Blackwell, "Particle swarm optimization: An overview", Swarm Intell(2007) 1:p.33-57 DOI 10.1007/s11721-007-0002-0.
[4] ZHANG Li-ping, YU Huan-jun, HU Shang-xu, "Optimal choice of parameters for particle swarm optimization", Journal of Zhejiang University SCIENCE ISSN 1009-3095, 2005.
[5] Clerc M, Kennedy J, "The particle swarm explosion, stability and convergence in a multidimensional complex space", IEEE transaction on Evaolutionary Computation, 6(1):p.58-73, 2002.
[6] http://www.particleswarm.info/Programs.html
[7] Marcin Molga, Czeslaw Smutnicki, "Test functions for optimization needs", 2005.
[8] Neclai Andrei, "An Unconstrained Optimization Test Functions Collection", Advanced Modeling and Optimization, Vol.10, No.1, 2008.

이 영 아

e-mail : leeyaa10@yahoo.co.kr
2004년 경희대학교 컴퓨터공학과(공학박사)
2008년~2009년 연세대학교 컴퓨터과학과 BK21 연구교수
관심분야 : Reinforcement Learning, Optimization Algorithm

김 택 헌

e-mail : kimthun@cs.yonsei.ac.kr
2005년 연세대학교 컴퓨터과학(공학박사)
1996년~2000년 삼성SDS 정보기술연구소 연구원
2005년~2006년 연세대학교 컴퓨터과학과 BK21 박사후연구원
2006년~현 재 연세대학교 컴퓨터과학과 BK21 연구교수
관심분야 : Information Searching & Retrieval, Mobile Computing, Personalization

양 성 봉

e-mail : yang@cs.yonsei.ac.kr
1992년 University of Oklahoma 컴퓨터과학(공학박사)
1993년~1994년 전주대학교 전자계산학과 전임강사
1994년~현 재 연세대학교 컴퓨터과학과 교수
관심분야 : Information Searching & Retrieval, Mobile Computing, 3D-Graphics, Computer Theory