韓國컴퓨터情報學會 論文誌
第16卷 第12號, 2011. 12.

2011-16-12-1-3

등급기준 교란을 통한 단순 박테리아협동 최적화의 성능향상

정 성 훈*

Performance Improvement of Simple Bacteria Cooperative Optimization through Rank-based Perturbation

Sung Hoon Jung*

요 약

최적화 알고리즘의 하나로 제안한 단순 박테리아협동 최적화는 비교적 좋은 성능을 보였으나 개체가 한 번에 한 스텝씩 움직이는 것으로 말미암아 성능에 한계가 발생하였다. 이러한 문제를 해결하고자 개체에 등급을 매기고 등급별로 개체의 속력을 할당하는 방법을 제안하여 어느 정도의 성능향상을 보았다. 본 논문에서는 개체에 속력을 할당하는 방법에 추가적으로 성능향상을 위하여 기존의 진화적 최적화 알고리즘들이 많이 사용한 돌연변이를 새로 추가한 알고리즘을 제안한다. 새로 추가한 돌연변이에서는 적합도가 좋지 않은 일정 퍼센트의 개체를 해당 개체의 등급에 비례하는 영역내로 돌연변이를 일으킨다. 즉, 적합도가 낮아 등급이 낮으면 더 큰 표준편차의 가우시안 잡음을 섞어서 돌연변이를 발생한다. 결국 낮은 등급을 갖는 개체들은 부모로부터 멀리 떨어질 확률이 증가하게 된다. 이렇게 함으로서 개체가 지역 최적해 영역에 빠질 가능성을 줄이고 지역 최적해 영역에 빠져도 빠르게 나올 수 있는 가능성이 높아진다. 네 개의 함수 최적화 문제에 적용해본 결과 개체 속력과 돌연변이를 함께 적용했을 경우에 성능이 향상되는 것을 보았다. 다만, 아주 복잡도가 높은 함수에서는 반드시 좋아지지 만은 않았는데, 추후 이를 해결하기 위한 다른 방법을 고안해야할 것으로 판단된다.

▸ Keyword : 최적화, 박테리아협동 최적화, 등급기준 교란, 함수최적화

Abstract

The simple bacteria cooperative optimization (sBCO) algorithm that we developed as one of optimization algorithms has shown relatively good performances, but their performances were limited by step-by-step movement of individuals at a time. In order to solve this problem, we

• 제1저자 : 정성훈 교신저자 : 정성훈
• 투고일 : 2011. 07. 15, 심사일 : 2011. 08. 04, 게재확정일 : 2011. 09. 27.
* 한성대학교 정보통신공학과 (Department of Information and Communications Engineering, Hansung University)
※ 본 논문의 초기결과는 2011 International Conference on Management and Artificial Intelligence에서 발표하였음. 본 연구는 한성대학교 교내연구비 지원과제 임.

proposed a new method that assigned a speed to each individual according to its rank and it was confirmed that it improved the performances of sBCO in some degree. In addition to the assigning of speed to the individuals, we employed a new mutation operation that most existing evolutionary algorithms used in order to enhance the performances of sBCO in this paper. A specific percent of bad individuals are mutated within an area that is proportion to the rank of the individual in the mutation operation. That is, Gaussian noise of large standard deviation is added as the fitness of individuals is low. From this, the probability that the individuals with lower ranks can be located far from its parent will be increased. This causes that the probability of falling into local optimum areas is decreased and the probability of fast escaping the local optimum areas is increased. From experimental results with four function optimization problems, we showed that the performances of sBCO with mutation operation and individual speed were increased. If the optimization function is quite complex, however, the performances are not always better. We should devise a new method for solving this problem as a further work.

▸ Keyword : Optimization, Bacteria cooperative optimization, Rank-based perturbation, Function optimization

I. 서론

최근 들어 개미군집 최적화(ACO: Ant Colony Optimization), 인공면역 시스템(AIS: Artificial Immune System), 입자군집 최적화(PSO: Particle Swarm Optimization)등 생물에서 영감을 받아 개발한 알고리즘이 공학적 최적화 문제에 응용되어 성공적으로 사용되고 있다 [1-3,8,9]. 우리는 또 다른 생물인 박테리아가 먹이를 찾아가는 주화성(chemotaxis)에서 영감을 받아 새로운 최적화 알고리즘인 박테리아협동 최적화(BCO: Bacteria Cooperative Optimization) 알고리즘을 제안하였다 [4,6]. 박테리아의 일종인 대장균(E. Coli)이 먹이를 찾아가는 행동양식을 모방하여 인공대장균(AE: Artificial E. Coli)을 만들고 인공대장균이 먹이의 분포가 높은 쪽으로 이동하는 주화성을 이용하여 최적화 문제를 해결 한다 [7].

제안한 알고리즘은 기존의 단순유전자알고리즘과 비교하여 비교적 좋은 성능을 보였으나 성능이 좋지 않은 경우도 발생하였다. 성능이 좋지 않은 경우의 대부분은 개체(박테리아협동 최적화에서는 인공대장균)수가 작은 경우이다. 기본적으로 박테리아협동 최적화 방법은 먹이농도의 기울기를 이용한 방법이기 때문에 지역 최적해 영역에 빠질 가능성이 높으며 일단 지역 최적해 영역에 빠지면 이 영역을 벗어나기가 어렵다. 이러한 이유로 개체수가 작은 경우 대부분의 개체가 지역 최적해 영역에 빠지면 벗어나지 못하고 성능이 급격히 저하된다. 개체수가 많은 경우에는 개체가 전역 최적해 영역에도 빠질 확률이 높기 때문에 큰 문제가 되지 않는다. 그러나 개체수를 늘리면 계산비용이 증가하기 때문에 단순히 개체수를 늘리는 것이 해결방법은 아니다.

우리는 이러한 문제를 인식하고 인공대장균 중에서 적합도가 낮은 (즉, 먹이농도가 낮은 곳에 있는) 특정 퍼센트의 개체를 제거하고 해당 개수만큼 무작위적인 위치에 새로운 개체를 생성하는 방법을 도입하였다 [5]. 비록 이 방법이 새로 만들어지는 개체에 의해서 지역 최적화 영역에서 벗어날 수 있어서 어느 정도의 성능향상을 가져왔으나 새로 만들어지는 개체가 기존의 개체가 찾은 어떤 정보도 갖지 못하고 전체 영역에 무작위로 만들어짐으로 인하여 효과적인 해결방법은 아니었다. 우리는 이런 문제를 인식하고 기존의 알고리즘에서 인공대장균이 한 번에 한 칸씩 이동하던 것을 각 개체의 등급에 따라서 이동하는 속력 개념을 적용한 알고리즘을 제안하였다 [6]. 즉 적합도가 제일 좋은 1등은 한 번에 한 칸을 3등은 한 번에 3칸을 이동한다. 이렇게 하면 적합도가 좋은 인공대장균은 주변에 전역 최적해가 있을 가능성을 감안하여 세부적으로 탐색하고 적합도가 낮은 인공대장균은 한 번에 여러 칸을 이동함으로 지역 최적해 영역을 벗어나 전역 최적해로 이동할 수 있을 것이다. 이 방법으로 기존에 방법에 비하여 상당히 좋은 성능을 얻을 수 있었다.

본 논문에서는 성능을 더욱 향상시키기 위한 방법으로 기존의 유전자알고리즘에서 많이 사용하는 돌연변이(mutation) 연산자를 새로 도입하였다. 즉 적합도가 낮은 일정 퍼센트의 개체에 등급별로 돌연변이를 가하여 이전 위치에서 다른 곳으로 점프를 가능하게 하는 것이다. 이렇게 함으로써 지역 최적

해 영역에서 쉽게 벗어날 수 있는 방법을 제공한다. 이 방법은 적합도가 낮은 일정 퍼센트의 개체를 제거하고 전체 영역에 걸쳐서 무작위로 생성하는 방법에 비하여 적합도 별로 돌연변이 하는 영역이 제한되므로 보다 효율적인 결과를 가져온다. 즉 나쁜 개체 중에서도 비교적 좋은 개체는 이전 위치 근처로 돌연변이하며 아주 나쁜 개체는 이전 위치에서 멀리 떨어진 곳으로 돌연변이하게 된다. 이렇게 함으로써 나쁜 개체들이 지역 최적해 영역에서 벗어나 전역 최적해 영역으로 접근할 수 있는 기회를 갖게 된다.

네 개의 함수 최적화문제를 이용하여 제안한 알고리즘의 성능을 평가한 결과 성능향상이 있음을 보았으나 기존의 등급별 속력을 적용한 방법보다는 크게 좋아지지 않았다. 다만 등급별 속력과 돌연변이를 함께 적용한 경우에는 속력만을 적용한 것보다 많은 경우에 성능이 좋아지는 것을 확인할 수 있었다. 본 논문은 다음과 같이 구성되었다. 2절에는 단순 박테리아협동 최적화 알고리즘을 소개한다. 3절에서는 속력과 돌연변이를 모두 적용한 등급기준 교란을 갖는 단순 박테리아협동 최적화 알고리즘을 설명한다. 4절에서는 네 가지 함수최적화 문제를 적용하여 기존의 방법과 제안한 방법의 여러 가지 실험을 한 결과를 제시하며 5절의 결론으로 끝을 맺는다.

II. 단순 박테리아협동 최적화

우리는 박테리아의 일종인 대장균의 행동양식을 모방하여 단순 박테리아협동 최적화 알고리즘을 제안하였다 [4]. 논문[7]에서 기존의 생물학적 연구를 바탕으로 먹이 분포에 따른 대장균의 움직임을 모델링하는 연구를 수행하였는데 이 논문에서 정립한 대장균의 행동양식을 적용하여 최적화 알고리즘을 개발하였다. 그 이유는 대장균이 수천만 년을 거쳐 진화해 오면서 먹이를 찾아가는 행동양식이 최적화 되었을 것이라는 전제를 기반으로 한다. 그러므로 대장균이 먹이를 찾아가는 방법을 모델링하고 이를 규칙화해서 인공대장균에 적용함으로서 공학적인 최적화 문제에 적용할 수 있는 알고리즘을 개발하였다.

우리는 논문[7]에서 대장균의 행동양식을 크게 행위규칙(Behavior rules)과 결정규칙(Decision rules)의 두 종류로 모델링하였다. 이 두 규칙은 단순 박테리아협동 최적화의 핵심요소이다. 행위규칙은 두개의 세부규칙으로 이루어졌는데 다음과 같다. (B1) 인공대장균은 매 B_n 스텝 이동마다 계속 직진(run)할지 아니면 방향을 전환할지(tumble)를 결정한다. (B2) 만약 직진 스텝(run counts)이 $B_m (> B_n)$이 되면 인공대장균은 무조건 방향전환을 한다.

첫 번째 행위규칙 B1은 먹이농도의 전체적인 분포와 기울기과 관계가 있다. 만약 먹이농도의 변화가 작다면 (즉 먹이농도의 기울기가 작다면) 자주 행위를 결정할 필요가 없기 때문에 큰 B_n을 선택한다. 그러나 먹이농도의 기울기는 위치에 따라 다르기 때문에 신중히 선택해야하며 특히 전역 최적해 근방에서의 기울기를 고려하여 선택해야한다. B_n이 작으면 직진할지 방향전환할지를 자주 결정하기 때문에 언덕 오르기(hill climbing) 방법과 같이 지역 최적해 영역에 빠질 가능성이 커진다. B_n이 크면 지역 최적해 영역에 빠질 가능성은 줄어드나 전역 최적해로 세부적인 접근이 어렵다는 단점이 있다. 그러나 여기에서 B_n이 작다거나 크다거나 하는 것은 절대적인 것이 아니라 먹이농도의 전역 및 지역해의 기울기에 의존한다. 즉 기울기가 작다면 B_n이 작은지 큰지를 나누는 경계 값이 커져야 하며 그렇지 않으면 경계 값이 작아야한다.

두 번째 행위규칙 B2는 두 가지 효과를 갖는다. 첫 번째로 현재의 방향이 먹이농도가 증가하는 좋은 방향이라고 하더라도 먹이농도가 더 빨리 증가하는 더 좋은 방향이 있는지를 탐색할 수 있다. 물론 방향전환을 하여 현재 방향보다 먹이농도가 증가하는 것이 더 작아질 가능성도 있기 때문에 반드시 좋아지는 것은 아니다. 두 번째 효과로는 지역 최적해 영역에 빠지는 가능성을 줄여준다는 것이다. 현재의 방향이 먹이농도가 증가하는 방향이라고 무조건 직진한다면 이는 지역 최적해 영역에 빠질 가능성을 높여준다. 그러나 B2로 지역 최적해 영역에 빠지는 것을 완전히 막지는 못하기 때문에 지역 최적해 영역에 빠지는 것을 막는 부가적인 방법이 필요하다.

결정규칙도 두개의 세부규칙으로 구성되는데 두개의 세부규칙은 다음과 같다. (D1) 인공대장균은 D_n 스텝에서의 먹이농도를 평균하여 현재의 먹이농도를 계산하며 $D_m (> D_n)$ 스텝에서의 먹이농도를 평균하여 이전의 먹이농도를 계산한다. (D2) 만약 현재의 먹이농도가 이전의 먹이농도보다 크면 직진으로 행위를 결정하며 그렇지 않으면 방향전환으로 행위를 결정한다. 첫 번째 결정규칙은 인공대장균이 먹이농도 변화를 얼마나 민감하게 체크할지를 결정한다. 만약 D_n이 1이고 D_m이 2이면 현재 위치의 먹이농도가 현재 먹이농도가 되며 현재 위치의 농도와 이전 위치의 농도를 평균한 것이 이전 먹이농도가 되어 먹이농도의 변화를 매우 민감하게 체크하는 결과를 가져온다. 반면에 D_n과 D_m이 크면 먹이농도의 변화를 민감하지 않게 체크하게 된다. 만약 전역 최적해가 좁은 영역에 걸쳐 있다면 작은 D_n과 D_m이 좋을 것이다. 그러나 이 영역이 지역 최적해 영역일 경우에는 작은 D_n과 D_m이 오히려 지역

최적해 영역을 벗어나는데 방해가 된다. 그러므로 적절한 D_n과 D_m을 찾는 것은 함수에 따라 다르며 매우 어려운 일이다. 행위규칙 두개와 결정규칙 두개를 핵심 동작으로 하여 단순 박테리아협동 최적화 알고리즘을 논문 [4]에서 개발하였다.

III. 등급기준 교란을 갖는 단순 박테리아협동 최적화

단순 박테리아협동 최적화가 어느 정도의 성능을 보였지만 근본적인 성능의 한계를 보였다. 그 이유는 개체가 한 번에 한 스텝만 움직이기 때문에 전역 최적해 영역에 접근하는데 시간이 오래 필요했으며 더 큰 문제는 지역 최적해 영역에 빠진 경우 쉽게 지역 최적해 영역을 벗어날 수 없었다. 왜냐하면 단순 박테리아협동 최적화의 경우 기본적으로 먹이농도의 경사를 이용하는 방법이기 때문이다. 지역 최적해 영역을 벗어나려면 먹이농도가 감소하여도 어느 정도 이동하거나 지역 최적해 영역을 벗어나기 위한 다른 방법이 있어야한다. 행위규칙에 의하여 어느 정도 좁은 영역의 지역 최적해 영역은 벗어날 수 있으나 넓은 영역에 걸쳐있는 지역 최적해 영역은 벗어나기가 어렵다.

이러한 이유로 인공대장균은 오랜 기간 지역 최적해 영역에 머물게 되고 성능은 떨어질 수밖에 없다. 특히 인공대장균의 개체수가 작은 경우에 이러한 경향은 더 크다. 이 문제를 해결하기 위한 하나의 방법으로 우리는 적합도가 낮은 개체 중 일정 퍼센트의 개체를 무작위적인 위치에 생성되는 새로운 개체로 대체하는 방법을 제안하였다 [5]. 이 방법은 어느 정도 성능을 향상시켰으나 새로 생성되는 개체가 전 영역에 걸쳐 만들어지기 때문에 기존에 찾은 정보를 이용할 수 없는 비효율성을 갖고 있었다.

보다 더 효율적으로 이 문제를 해결하기 위한 새로운 방법으로 우리는 속력개념을 두어 개체가 한 번에 한 스텝씩 이동하는 것이 아니라 등급에 따라서 한 번에 여러 스텝을 이동하는 방법을 제안하였다 [6]. 즉 등급이 높은 좋은 개체는 이동 스텝을 작게 하고 등급이 낮은 나쁜 개체는 이동 스텝을 많게 하는 것이다. 이렇게 함으로서 등급이 좋은 개체는 주변의 영역을 탐색하고 등급이 나쁜 개체는 한 번에 많은 스텝을 이동해 지역 최적해 영역을 벗어날 수 있게 하였다. 이 방법은 기존의 방법들 보다 성능이 월등이 향상됨을 보였다. 특히 등급별로 이동하는 스텝을 달리하기 때문에 전체 영역에 걸쳐서 무작위로 새로운 개체를 만드는 방법에 비하여 효율적이어서 성능이 향상될 수 있었다.

본 논문에서는 등급별 이동스텝을 다르게 하는 속력방법에 추가하여 등급별로 돌연변이를 발생시키는 방법을 제안한다. 즉 등급이 좋지 않은 일정 퍼센트의 인공대장균에 돌연변이를 발생시킨다. 돌연변이는 해당 개체의 등급에 비례하는 가우시안 교란(perturbation)을 섞어서 발생시킨다. 즉 등급이 낮은 경우 더 많은 교란을 섞어주어서 해당 위치에서 더 많이 벗어난 곳으로 돌연변이 되게 한다. 이렇게 함으로서 등급별로 개체이동을 조정하는 속력방법에 추가하여 등급별로 돌연변이를 시키는 등급기준 교란을 갖는 단순 박테리아협동 최적화 방법을 제시하였다.

알고리즘 1은 등급기준 교란을 갖는 단순 박테리아협동 최적화 알고리즘이다. 단순 박테리아협동 알고리즘은 초기화 E(t), 이동 E(t), 측정 E(t), 결정 E(t) 네 개의 동작으로 이루어져있다. 초기화 E(t)는 인공대장균의 위치, 이동방향 등 각종 파라미터의 초기 값들을 설정한다. 이동 E(t)는 이동 스텝 수에 따라서 이동방향으로 이동한다. 위에서 언급한 것처럼 속력버전에서는 등급에 의해 결정된 이동 스텝 수에 따라서 이동한다. 즉 1등은 한 번에 한 스텝 움직이며 n등은 n 스텝 움직인다. 이를 위해 제안된 알고리즘에서는 각 인공대장균의 등급을 매기는 등급 E(t)이 추가되었다. 측정 E(t)는 각 인공대장균의 위치에서 먹이농도를 측정하며 결정규칙에 따라서 이전 먹이농도와 현재 먹이농도를 계산한다. 결정 E(t)는 행위규칙 B1 과 B2 그리고 결정규칙 D2에 따라서 직진할지 방향전환할지를 결정하는 과정으로서 가장 중요한 동작이다. 돌연변이를 위하여 결정 E(t) 뒤에 돌연변이 E(t)를 추가하였다. 위에서 설명한 것처럼 이 과정에서는 적합도가 나쁜 일정 퍼센트의 인공대장균을 등급에 비례하여 돌연변이 시킨다.

등급에 따른 속력과 등급에 따른 돌연변이는 인공대장균이 전역최적해로 이동하는데 도움을 주기보다는 지역 최적해 영역에 빠진 것을 벗어나는데 도움을 준다. 그러므로 단순 박테리아협동 최적화의 문제점을 완화하는데 큰 기여를 할 수 있다.

알고리즘 1. 개체속력을 적용한 박테리아 협동 최적화

// t : 이산시간 //
// r_c : 직진 이동개수 //
// B_n : 인공대장균이 행동을 결정할 스텝 수 //
// B_m : 인공대장균이 무조건 방향 전환할 스텝 수 //
// D_n, : 현재의 먹이농도를 측정하기위한 스텝 수//
// D_m : 과거의 먹이농도를 측정하기위한 스텝 수 //
// ρ_{D_n}, : 현재의 먹이농도 //
// ρ_{D_m} : 과거의 먹이농도 //
// s : 한 번에 이동하는 거리 (속력적용) //
// $E(t)$: 시간 t에서의 인공대장균 집합 //

$t=0$
초기화 $E(t)$
    탐색 공간상에 일정한 분포로 인공대장균을 무작위로 생성
    생성된 인공대장균의 방향을 무작위로 결정
    생성된 인공대장균의 행동을 직진으로 결정
    모든 인공대장균의 직진 이동개수 r_c, 현재 먹이농도 ρ_{D_n}, 과거 먹이농도 ρ_{D_m}를 0으로
    현재의 위치에서 먹이농도를 측정하여 저장
while (not 종료조건)
do
    $t=t+1$
    등급 $E(t)$
        모든 인공대장균의 등급을 매김
        등급을 각 인공대장균의 속력으로 설정
    이동 $E(t)$
        설정된 방향으로 s 만큼 직진이동
        직진 이동 수 r_c를 증가
    측정 $E(t)$
        현재의 위치에서 먹이농도의 농도를 측정하여 저장
        ρ_{D_n} 과 ρ_{D_m}를 계산 ▷ 결정규칙 (D1)
    결정 $E(t)$
        if (r_c 나머지 B_n) $=0$ then ▷ 행동규칙 (B1)
            if $\rho_{D_n} > \rho_{D_m}$ then ▷ 결정규칙 (D2)
                직진 설정
            else
                방향전환 설정
            end if
        end if
        if $r_c = B_m$ then ▷ 행동규칙 (B2)
            방향전환 설정
        end if
        if 방향전환 then
            현재방향과 반대방향을 제외하고 임의의 방향으로 방향 설정
            직진설정
            직진 이동개수 설정, $r_c=0$
        end if
    돌연변이 $E(t)$
        특정 퍼센트의 적합도가 나쁜 인공대장균을 선정
        인공대장균의 등급에 따라서 돌연변이를 시킴
end

$$f_1 = 3000 - 3(x^2 + y^2), where \ -20 \le x, y \le 20$$

$$f_2 = e^{(-(\frac{x+3}{3})^2-(\frac{y+8}{3})^2)} + 0.8e^{(-(\frac{x-10}{4})^2-(\frac{y-6}{4})^2)} + 0.34e^{(-(\frac{x+10}{5})^2-(\frac{y-6}{4})^2)} + 0.19e^{(-(\frac{x+22}{4})^2-(\frac{y+25}{6})^2)} + 0.13e^{(-(\frac{x+1}{9})^2-(\frac{y-16}{6})^2)} + 0.10e^{(-(\frac{x+13}{8})^2-(\frac{y+6}{8})^2)} + 0.07e^{(-(\frac{x-11}{7})^2-(\frac{y+10}{8})^2)}, \ where -20 \le x, y \le 20 \quad (1)$$

$$f_3 = \sum_{i=1}^{27} A_i e^{(-(\frac{x+B_i}{C_i})^2-(\frac{y+D_i}{E_i})^2)}, \ where -20 \le x, y \le 20 \text{ and } A_i, \ B_i, \ C_i, \text{and} D_i \ are given \ as \ Table 1$$

$$f_4 = 0.5 - \frac{\sin(\sqrt{x^2+y^2})\sin(\sqrt{x^2+y^2}) - 0.5}{(1.0+0.001(x^2+y^2))(1.0+0.001(x^2+y^2))}, \ where -10 \le x, y \le 10$$

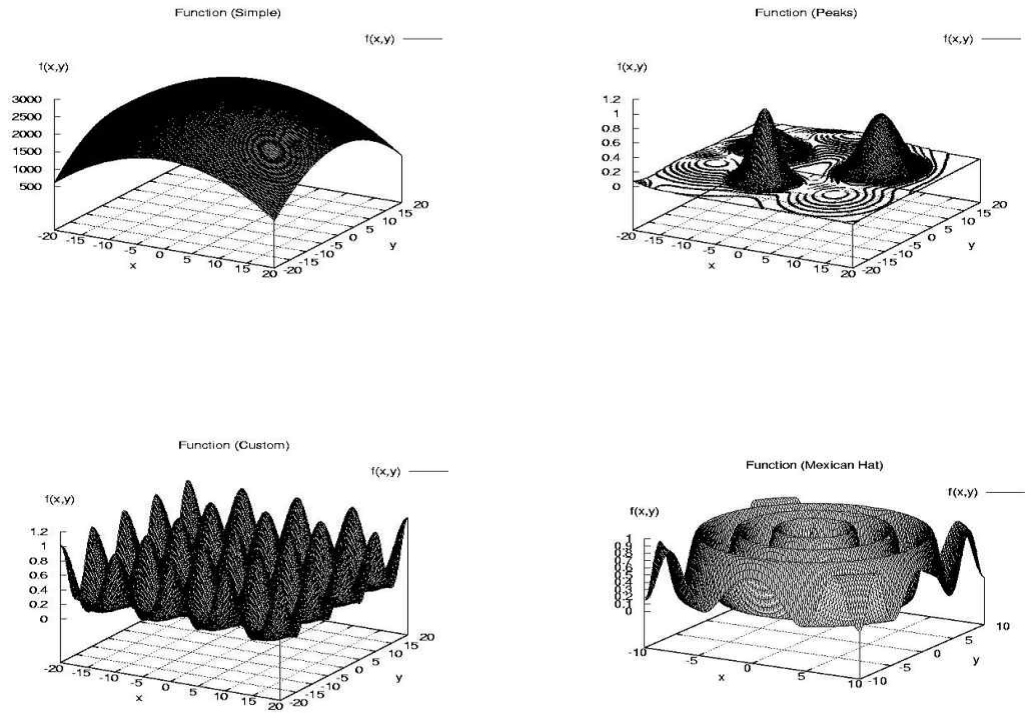

그림 1. 함수최적화 문제
Fig.Function optimization problems

IV. 실험 및 결과

제안한 방법의 성능을 측정하기 위하여 네 개의 함수 최적화 문제를 이용하여 최적화를 수행하였다. 네 개의 함수최적화 문제는 최적화 난이도에 따라 선정하였다. 수식 (1)은 네 개의 함수수식이며 표 1은 함수 $f3$의 파라미터이다. 그림1은 각 함수의 최적화 난이도를 보여주기 위한 것으로서 입출력관계를 보여준다. 먼저 함수 f_1은 가장 단순한 함수로서 하나의 전역 최적해 영역을 갖고 있으며 지역 최적해 영역은 없다. 이 경우에는 지역 최적해 영역에 빠질 수가 없어서 경사를 이용하여 언덕을 오르는 방법이 유리하다. 두 번째 함수 f_2는 여러 개의 지역최적해가 있는 피크함수이다. 비교적 넓은 영역에 걸쳐서 여러 개의 지역 최적해가 있기 때문에 최적화하기가 어렵다. 세 번째 함수는 전역 최적해와 거의 비슷한 모양의 지역 최적해가 전 영역에 넓게 걸쳐있는 문제로서 최적화에 상당히 어려운 문제이다.

표 1. 함수 f_3의 파라미터
Table 1. The parameter of function f_3

i	1	2	3	4	5	6	7	8	9
A	1.00	0.99	0.98	0.99	0.98	0.98	0.99	0.99	0.99
B	0	0	0	0	0	10	10	10	10
C	3	3	3	3	3	3	3	3	3
D	0	-10	-20	10	20	20	10	0	-10
E	3	3	3	3	3	3	3	3	3
i	10	11	12	13	14	15	16	17	18
A	0.98	0.98	0.99	0.99	0.99	0.98	0.98	0.99	0.99
B	10	20	20	20	20	20	10	10	10
C	3	3	3	3	3	3	3	3	3
D	-20	20	10	0	-10	-20	20	10	0
E	3	3	3	3	3	3	3	3	3
i	19	20	21	22	23	24	25	26	27
A	0.99	0.98	0.98	0.99	0.99	0.99	0.98	0.04	0.04
B	10	10	20	20	20	20	20	0	0
C	3	3	3	3	3	3	3	30	30
D	-10	-20	20	10	0	-10	-20	0	0
E	3	3	3	3	3	3	3	30	30

마지막으로 네 번째 함수는 멕시칸모자(Mexican hat)라고 알려진 문제로서 가운데 좁은 영역에 있는 전역 최적해 영역을 여러 지역해가 겹겹이 둘러친 모양으로 최적화 문제에 있어서 상당히 난이도가 있는 문제이다.

실험을 위하여 (B_n, B_m) 과 (D_n, D_m) 파라미터를 각각 (3, 9)와 (3, 9)로 설정하였다. 이는 실제 대장균에서 측정한 데이터를 이용하여 모델링한 결과 값으로 그대로 사용하였다[7]. 인공대장균의 속력은 각 대장균의 등급과 같은 값으로 주었으며 돌연변이는 등급이 낮을 경우 더 크게 돌연변이가 일어나게 하였다. 돌연변이는 다음의 수식처럼 가우시안함수를 써서 등급에 비례하여 커지게 주었다. 수식(1)에서 보듯이 i번째 인공대장균의 돌연변이는 돌연변이 이전의 (x_i^t, y_i^t)의 위치에서 $(x_i^{(t+1)}, y_i^{(t+1)})$의 위치로 돌연변이 한다. 이때 돌연변이 위치는 이전의 위치에 평균이 0이고 x와 y축으로 각각 (σ_x, σ_y)의 표준편차를 갖는 가우시안 랜덤 잡음 $N(0, \sigma_x)$와 $N(0, \sigma_y)$를 섞어준다.

$$x_i^{(t+1)} = x_i^t + N(0, \sigma_x), \quad y_i^{(t+1)} = y_i^t + N(0, \sigma_y) \cdots\cdots\cdots (1)$$

이때 표준편차 σ_x와 σ_y는 수식(2)와 같이 주었다.

$$\sigma_x = \frac{X^{unit}}{6}\frac{(r_i - r_{th})}{(N - r_{th})}, \quad \sigma_y = \frac{Y^{unit}}{6}\frac{(r_i - r_{th})}{(N - r_{th})} \cdots\cdots\cdots (2)$$

위 수식에서 X^{unit}와 Y^{unit}는 각각 인공대장균이 움직이는 공간의 x축 y축 크기이며 r_i는 i번째 개체의 등급이고 r_{th}는 돌연변이 시작등급이고 N은 인공대장균의 개수이다. 예를 들어 인공대장균의 개체가 20개이고 80퍼센트의 나쁜 개체를 돌연변이 한다고 할 때 $r_{th} = 20(1-0.8) = 4$ 가 되어 돌연변이는 5등급부터 20등급까지 실행된다. 20개의 인공대장균 중에서 가장 적합도가 나쁜 개체인 20등급인 개체는 x축 y축으로 각각 $\sigma_x = X^{unit}/6$, $\sigma_y = Y^{unit}/6$ 의 표준편차로 가우시안 랜덤잡음을 섞어서 돌연변이하게 된다. 여기에서 분모 6은 가장 나쁜 인공대장균에 섞어주는 가우시안 랜덤 잡음이 99.73퍼센트의 확률로 각 축의 공간의 크기에 반 안에 만들어지게 하기 위함이다. 즉 평균이 μ이고 표준편차가 σ인 가우시안분포에서 99.73 퍼센트의 확률은 $P[\mu - 3\sigma \le x \le \mu + 3\sigma] = 0.9973$이기 때문에$3\sigma_x = X^{unit}/2$, $3\sigma_y = Y^{unit}/2$ 가 된 것이다. 결국 x와 y축의 정 가운데에 있는 가장 적합도가 나쁜 인공대장균이 99.73퍼센트의 확률로 각 축의 최대 크기 안으로 돌연변이를 발생하게 된다. 만약에 돌연변이가 x나 y축의 최대 좌표보다 더 커지면 다시 각 축의 처음으로 돌아오는 환상 면(toroidal)공간으로 가정하였다. 물론 돌연변이 하는 개체 중 가장 좋은 개체는 수식(2)에 따라서 비교적 표준편차가 작게 돌연변이 된다.

실험 시에 인공대장균의 개수 n과 돌연변이를 수행할 적합도가 나쁜 인공대장균의 퍼센트 η 도 실험별로 주어진다. 모든 인공대장균은 그들에게 설정된 파라미터에 따라서 최적화 영역에서 이동한다. 어떤 한 인공대장균이 전역 최적해를 찾은 경우 최적화를 완료한 것으로서 그 당시의 시간 t를 기록한다. 실험을 위하여 각 파라미터별로 10번씩 실험하여 통계를 내었다. 통계 값으로는 평균과 표준편차를 구했는데, 단순화를 위해 표준편차는 실험결과 표에서 생략하였다. 본 논문에서 제안한 방법과 기존방법의 성능을 비교하기위하여 유전자알고리즘과 단순 박테리아협동 최적화에서 기존의 제안된 방법들을 모두 실험하여 비교하였다. 실험결과 표2에서 sGA, sBCO, sBCO_RR, sBCO_VS, sBCO_MUT, sBCO_RBP는 각각 단순 유전자알고리즘, 단순 박테리아협동 최적화, 등급치환을 갖는 단순 박테리아협동 최적화, 개체속력을 갖는 단순 박테리아협동 최적화, 돌연변이를 갖는 단순 박테리아협동 최적화, 속력버전과 돌연변이를 함께 적용한 등급기준 교란을 갖는 박테리아협동 최적화를 나타낸다. 등급치환을 갖는 단순 박테리아협동 최적화 실험에서는 해당 실험에서 가장 성능이 잘 나온 50퍼센트의 나쁜 개체를 무작위로

표 2. 기존 방법과의 비교 실험 결과
Table 2. Comparing experimental results of existing methods

함수	개체수 n	sGA	sBCO	sBCO_RR	sBCO_VS	sBCO_MUT ($\eta=0.9$)	sBCO_RBP ($\eta=0.9$)
f_1	100	10276.8	697.6	176.4	172.1	66.9	65.0
	200	9622.1	388.0	158.1	101.8	65.5	49.3
	300	4956.0	440.2	146.2	85.4	57.3	44.4
f_2	100	308776.0	917.4	238.0	263.4	125.7	159.7
	200	352490.8	841.4	183.9	143.4	82.5	74.3
	300	97377.0	662.1	158.6	151.0	76.4	118.2
f_3	100	14642.0	817.4	200.2	254.5	147.1	164.0
	200	13887.4	853.9	197.4	153.6	89.0	213.4
	300	15627.2	620.0	174.5	151.5	115.9	78.9
f_4	100	40676.5	1155.8	509.5	551.8	802.3	573.7
	200	25782.0	639.7	366.0	339.3	456.5	775.4
	300	8992.5	681.3	378.4	556.9	408.0	386.0

표 3. η에 따른 실험 결과
Table 3. Experimental results of various η

함수	개체수 n	$\eta=0.5$		$\eta=0.7$		$\eta=0.9$	
		sBCO_MUT	sBCO_RBP	sBCO_MUT	sBCO_RBP	sBCO_MUT	sBCO_RBP
f_1	100	178.7	110.6	113.3	89.3	66.9	65.0
	200	140.3	87.8	99.9	69.6	65.5	49.3
	300	144.8	76.8	89.1	62.6	57.3	44.4
f_2	100	250.8	150.4	173.8	131.9	125.7	159.7
	200	203.2	95.1	150.3	125.6	82.5	74.3
	300	177.9	104.7	118.6	93.0	76.4	118.2
f_3	100	215.1	196.4	169.2	159.4	147.1	164.0
	200	148.9	157.4	125.2	120.7	89.0	213.4
	300	152.8	197.3	112.4	129.7	115.9	78.9
f_4	100	746.5	959.9	1112.5	719.7	802.3	573.7
	200	353.9	768.5	529.9	701.2	456.5	775.4
	300	329.9	523.6	426.6	380.0	408.0	386.0

생성한 새로운 개체로 대체하게 하였다. 본 논문에서 제안한 돌연변이는 90퍼센트의 나쁜 개체를 돌연변이 하는 경우에 결과가 가장 좋아서 $\eta=0.9$일 때의 결과와 비교하였다.

실험결과에서 가장 좋은 성능을 보이는 것에 밑줄을 표시하였다. 표2에서 보듯이 등급기준 교란을 갖는 방법이 많은 경우에서 좋았다. 그러나 돌연변이의 경우가 좋은 경우도 그 다음으로 많았으며 특히 비교적 난이도가 높은 f_2 와 f_3 함수에서는 돌연변이가 더 성능이 좋은 경우가 많았다. 함수의 최적화 관점에서 가장 어려운 문제인 f_4함수에서는 오히려 기존의 방법인 등급치환을 갖는 단순 박테리아협동 최적화와 개체속력을 갖는 단순 박테리아협동 최적화 방법이 좋았다. 이는 함수 f_4의 경우에 전역 최적해 영역을 지역 최적해 영역이 둘러싸고 있어서 어느 정도 무작위적인 새로운 개체 생성이 오히려 도움이 되는 것으로 판단된다. 보다 면밀한 검토를 통하여 함수 f_4에서도 성능이 좋은 새로운 방법을 개발할 필요성이 있다고 하겠다.

표 3은 돌연변이에서 돌연변이 할 나쁜 개체의 개수를 결정하는 퍼센트를 여러 가지 값으로 실험한 결과를 보여준다. 돌연변이의 경우 함수 f_1, f_2, f_3에서는 f_3의 300의 경우만 제외하고는 모든 실험에서 η값이 증가할수록 성능이 좋아졌

다. 그러나 함수 f_4에서는 이러한 경향성이 보이지 않았는데 이 실험결과에서도 보듯이 함수 f_4의 경우에는 오히려 많은 돌연변이가 전역 최적해 영역으로의 접근을 방해하는 것으로 보인다. 등급기준 교란 방법에서는 더욱 더 η값에 따른 경향성이 보이지 않는다. 그러므로 등급기준 교란 방법에서는 보다 더 세심하게 파라미터를 설정할 필요가 있다고 하겠다.

V. 결 론

본 논문에서 우리는 기존 단순 박테리아협동 최적화 알고리즘의 성능을 향상시키기 위하여 인공대장균에 등급기준 교란을 가하는 방법을 제안하였다. 이러한 교란은 인공대장균이 지역 최적해 영역에 빠지지 않도록 도움을 주고 지역 최적해 영역에 빠진 경우에도 벗어날 수 있도록 한다. 이러한 교란의 방법으로 기존의 등급기준으로 속력을 할당하는 방법에 더불어 등급에 비례하여 돌연변이를 수행하는 방법을 제안하였다. 네 개의 함수최적화 문제에 적용한 결과 성능이 향상됨을 볼 수 있었다. 특히 속력방법과 돌연변이를 동시에 적용한 경우 대부분의 함수에서 가장 좋은 성능을 보였다. 이런 결과로 보았을 때 등급기준 교란이 지역 최적해 영역에 빠지는 문제를 상당부분 완화하는 것으로 판단된다. 그러나 등급기준 교란이 주는 효과를 보다 면밀히 분석하여 성능을 더 향상시킬 수 있는 방법을 고안하는 것이 필요하다.

참고문헌

[1] M. Dorigo and T. Stutzle, “Ant Colony Optimization, ” The MIT Press, 2004.

[2] M. Clerc, “Particle Swarm Optimization, ” ISTE Publishing Company, 2006.

[3] L. N. de Castro and J. Timmis, “Artificial Immune Systems: A New Computational Intelligence Approach, ” Oxford University Press, 2002.

[4] S. H. Jung and T.-G. Kim, “A Novel Optimization Algorithm Inspired by Bacteria Behavior Patterns,” Journal of Korean Institute of Intelligent Systems, vol. 18, pp. 392-400, June 2008.

[5] S. H. Jung, “Simple Bacteria Cooperative Optimization with Rank Replacement,” Journal of Korean Institute of Intelligent Systems, vol. 19, pp. 432-436, June 2009.

[6] S. H. Jung, "Bacteria Cooperative Optimization Applying Individual's Speed for Performance Improvements", Journal of Institute of Electronics Engineers of Korea, vol. 47-CI, no. 3, pp. 67-75, June 2010.

[7] M. Kim, S. Baek, S. H. Jung, and K.-H. Cho, “Dynamical characteristics of bacteria clustering by self-generated attractants,” Computational Biology and Chemistry, vol. 31, pp. 328-334, Oct. 2007.

[8] K. M. Passino, “Biomimicry of Bacterial Foraging for Distributed Optimization and Control,“ IEEE Control Systems Magazine, vol. 22, no. 3, pp. 52-67, Jun. 2002.

[9] Sibylle D. Muller, Jarno Marchetto, Stefano Airaghi, and Petros Koumoutsakos, “Optimization Based on Bacterial Chemotaxis, “ IEEE Transactions on Evolutionary Computation, vol. 6, no. 1, pp. 16-29, Feb. 2002.

저 자 소 개

정 성 훈

1988 : 한양대학교 공학사.

1991 : 한국과학기술원 공학석사.

1995 : 한국과학기술원 공학박사.

1996~현재 : 한성대학교 정보통신공학과 교수

관심분야 : 지능시스템, 시스템생물학, 뇌공학

Email: shjung@hansung.ac.kr