J. Soc. Korea Ind. Syst. Eng Vol. 44, No. 2 : 58-65, June 2021
https://doi.org/10.11627/jkise.2021.44.2.058
ISSN : 2005-0461(print)
ISSN : 2287-7975(online)

Development of New Meta-Heuristic For a Bivariate Polynomial

Sung-Ho Chang · Moonsoo Kwon · Geuntae Kim · Jonghwan Lee†

School of Industrial Engineering, Kumoh National Institute of Technology

이변수 다항식 문제에 대한 새로운 메타 휴리스틱 개발

장성호 · 권문수 · 김근태 · 이종환†

금오공과대학교 산업공학부

Meta-heuristic algorithms have been developed to efficiently solve difficult problems and obtain a global optimal solution. A common feature mimics phenomenon occurring in nature and reliably improves the solution through repetition. And at the same time, the probability is used to deviate from the regional optimal solution and approach the global optimal solution. This study compares the algorithm created based on the above common points with existed SA and HS to show advantages in time and accuracy of results. Existing algorithms have problems of low accuracy, high memory, long runtime, and ignorance. In a two-variable polynomial, the existing algorithms show that the memory increases and the accuracy decrease. In order to improve the accuracy, the new algorithm increases the number of initial inputs and increases the efficiency of the search by introducing a direction using vectors. And, in order to solve the optimization problem, the results of the last experiment were learned to show the learning effect in the next experiment. The new algorithm found a solution in a short time under the experimental conditions of long iteration counts using a two-variable polynomial and showed high accuracy. And, it shows that the learning effect is effective in repeated experiments.

Keywords : Meta-Heuristic, Bivariate Polynomial, Simulated Annealing, Harmony Search

1. 서 론

해결해야할 수식의 변수가 많아져 해를 구하기 위한 시간은 늘어났고, 점점 해를 구하는 것이 어려워져 컴퓨터의 개발과 알고리즘의 발전을 촉진시켰다. 해결해야 하는 식은 점점 커져 나가며 P(Polynomial) 부류의 다차 함수 시간 결정성 알고리즘 또는 다차 함수 결정성 튜링 기계로 해결되는 결정 문제들의 집합문제가 생겨났다. 그리고 NP(Non-deterministic Polynomial) 부류인 비결정성(non-deterministic) 알고리즘인 비결정성 튜링 기계로 시간 내에 해결될 수 있는 결정 문제들의 집합이 생긴다. 이 문제는 yes 아니면 no로 답할 수 있는 상황에서 yes라고 답이 주었을 때, 답이 yes가 맞는지를 polynomial time안에 확인할 수 있게 된다면 NP문제라고 한다. 또는 비결정성 튜링 기계로 시간 안에 풀 수 있는 문제를 NP문제라고 한다. NP-hard 문제는 해결할 수 있는 다차 함수 결정성 알고리즘이 존재한다면 만족성 문제로부터 다차 함수 변환이 가능한 문제이다[10]. 오늘날의 컴퓨터는 Polynomial에 관한 문제는 일정 term을 주면 해결 가능할 것이다. 하지만 NP는 그와 반대로 긴 실행 사이클을 생성하며 해결을 위한 시간이 길어졌다. 즉, NP-Hard란 TSP(Traveling Salesman Problem)문제처럼 모든 경우의 수를 일일이 확인하는 방법이외에는 다항식처럼 답을 구할 수 없는 문제들을 말한다. NP 최적화 문제를 해결하는 방법으로 메타 휴리스틱 알고리즘이 많이 사용되고 있으며 대부분이 자연 현상을 모방하여 만들어 지고 있다. 문제의 복잡도가

Received 28 April 2021; Finally Revised 16 May 2021;
Accepted 17 May 2021
† Corresponding Author : shirjei@kumoh.ac.kr

커질수록 기존의 알고리즘은 지역 최적해(Local Optimum)에 빠져서 머무르거나 계산해야 할 양이 증가하는 등의 한계점에 도달하게 되었다. 대표적인 메타 휴리스틱 알고리즘으로는 타부 서치(Tabu Search), 유전 알고리즘(Genetic Algorithm), 시뮬레이티드 어닐링(Simulated Annealing), 개미군집화 알고리즘(Ant Colony Optimization)[2], 입자군집 최적화(Particle Swarm Optimization), 하모니 서치(Harmony Search)[4] 등이 제안되어 사용되고 있다. 메타 휴리스틱 알고리즘은 제한된 반복 횟수 안에 목적함수를 만족하는 최적해(Global Optimum)를 찾는 것을 목적으로 한다[1]. 하지만 시작 값을 랜덤으로 가지는 함수를 기반이기 때문에 최적 해에 도달한다는 보장을 할 수 없다. 메타 휴리스틱 알고리즘은 초기 해를 이용하여 랜덤 함수를 이용해 주어진 목적 함수 내에서 임의로 값을 선택하여 계속 업데이트하는 과정을 거친다. 업데이트하여 생성된 해와 기존의 해를 비교하여 나쁜 해를 버리는 과정을 거쳐 제한된 횟수 안에 알고리즘의 최적 해를 구한다. 초기 값을 좋은 값으로 결정을 하면 좋겠지만 좋은 초기 값을 알 수 없고 비록 알아도 랜덤한 값을 생산하는 와중에 소실될 가능성이 크다. 랜덤 선택과 결정 과정은 메모리를 증대시켜 시간을 오래 걸리게 하고 정확도를 떨어지게 한다. 메모리를 개선해가는 유전 알고리즘의 유전자 풀의 경우 하나의 해로 업데이트하는 과정이 랜덤 함수에 의존적이므로 수렴시간이 오래 걸리거나 지역해로 수렴할 가능성이 있다. 유전 알고리즘의 빌딩 블록 문제는 유전자 풀에 좋은 해가 잘 관리되지 못한다[7]. 이변수 이상의 함수에서는 기울기를 구하지 못하므로 방향성을 알지 못하며 미분을 이용한 최소, 최대 값을 구하기가 어렵다. 방향성을 상실하여 원하는 최적 해를 향해 바르게 나아가지 못하게 되면서 랜덤 선택의 문제와 합쳐져 최적 해를 찾는 시간과 정확도에서 큰 손실을 보게 된다. 그리고 최적해와 지역해가 무수히 많이 배치된 함수에서도 같은 문제점을 드러내고 있다. 일변수 다항식을 위한 알고리즘들이 많아 이변수 이상의 함수에서 최적 해를 구하는 과정에 연산과정과 메모리에 문제점을 가지고 있다. 그리하여 기존의 메타휴리스틱 알고리즘의 단점을 중화하고 장점을 유지하여 시간적의 효율성을 제고하고자 본 연구를 제시한다.

2. 이론적 배경

2.1 메타휴리스틱

휴리스틱 기법은 간단히 휴리스틱으로 불리며 문제 해결, 학습, 또는 발견을 위한 접근 방법으로써 최적 또는 완벽한 값을 보장하지는 못하지만 즉각적인 목표 값에 충실한 방법이다. 최적의 해결책을 찾기 불가능하거나 어려울 때 휴리스틱 기법은 만족스러운 해결책을 빠르게 찾는 과정에 이용된다. 메타 휴리스틱 기법에는 Genetic Algorithm (GA), Simulated Annealing(SA), Harmony Search(HS), Tabu Search 등 여러 가지가 존재한다. 각기 다른 특성이 있지만, 공통점은 개념과 이론이 단순하고 해공간의 탐색성능이 우수하여 공학, 자연과학뿐만 아니라 경영학, 사회과학 등의 최적화 분야 또는 의사결정분야에 응용 가능하다. 각기 독립적으로 여러 분야에 적용될 수 있을 뿐만 아니라, 각 기법의 단점을 상호 보완하면서 장점을 결합하여 함께 적용될 수 있다. 각각의 메타 휴리스틱 알고리즘은 장점뿐만 아니라 단점을 가지고 있고, 보완하기 위해 하이브리드형 메타 휴리스틱 알고리즘을 사용하기도 한다. 이러한 해법들은 조합최적화문제로 해석되는 현실의 문제들을 해결하기 위해 다양한 영역으로부터 아이디어를 얻은 결과이다.

2.2 선행 연구

2.2.1 시뮬레이티드 어닐링(SA : Simulated Annealing)

SA는 고체를 유한 온도의 열탕에 넣어 고체가 액체 상태가 될 때까지 가열한 후 열탕의 온도를 점점 낮추어 가장 안정된 상태의 고체를 얻는 어닐링 과정을 모사한 알고리즘으로 최적화 문제에 응용한 방법이다.

<Table 1> Simulated Annealing Algorithm

Begin
Get an initial solution S;
Get an initial temperature T > 0;
while not yet “frozen” do
for 1 = i to P (Iteration) do
Pick a random neighbor S' of S;
△ : input cost(S')-cost(S);
if △ <= 0 then S ← S' /* downhill move */
move to the good state
if △ > 0 then S ← S' /* uphill move */
If $\exp\left(\frac{-\Delta}{T}\right) \geq random(0, 1)$ then
move to the new state
end for
T ← rT;
return S
end
output: the final state s

물체의 온도를 x로 보고 상태를 목적함수로 보아 온도가 가장 낮을 때의 결정 상태를 얻으면 목적함수의 최적화가 이루어지는 것으로 보는 것이다. SA에서는 상태를 고정시켜서 온도를 변화하여 더 우수한 결정 상태에 도달하면 해를 이동하고 그렇지 않으면 다른 확률 $\exp\left(-\frac{\Delta}{T}\right)$로 이동하는 과정을 충분히 많이 반복하여 안정상태로 수렴시킨다.

그 후 상태를 변화시켜 다시 같은 과정을 반복을 하여 최적해가 이루어지도록 유도하게 된다[9]. SA는 모사한 상황과 최적화 문제와의 관련성이 명확하고 변수가 단순하며 수렴이 이론적으로 증명되었다는 장점이 있고, 산업 공정 및 디자인을 비롯한 실용문제에서 효과를 인정받았다[9].

2.2.2 하모니 서치(HS : Harmony Search)

HS는 연주자가 자신의 악기를 조율하는 과정에서 완벽한 하모니의 상태를 찾아가는 과정을 흉내 내서 만든 알고리즘이다. 악기에서 음을 찾을 범위는 제한되므로 변수의 범위로 보고 하나의 음을 정하여 다음으로 나아가는 과정을 최적화 과정으로 보았다. 또한 나는 소리들을 모은 것을 하모니 기억으로 하여 해집합을 이루도록 보고 경험이 많을수록 좋은 하모니를 찾는 것처럼 반복 진행을 하였다. 진행과정은 아래와 같은 과정을 따른다[4].

[1단계] 문제와 HS의 파라미터를 초기화 : 경계값 $LB_i \leq x_i \leq UB_i$를 가진 최소화(또는 최대화) 함수 $f(x)$를 목적함수로 가지고 x는 N개의 결정변수 x_i로 구성된다. 그리고 LB_i와 UB_i는 각 결정변수의 최소 경계값과 최대 경계값이다. 추가적으로 HS는 이 단계에서 HS만의 파라미터를 가진다. 이 파라미터들로는 HMS(harmony memory size), HMCR(harmony memory considering rate), PAR(pitch adjusting rate) 그리고 NI(number of improvisations)가 있다.

[2단계] 하모니 기억을 초기화 : 초기 하모니 기억은 균등분포로 된 $[LB_i, UB_i]$를 범위로 가지며 $1 \leq i \leq N$이다. 이것은 $x_i^j = LB_i + r \times (UB_i - LB_i), j = 1, 2, \cdots, HMS, r \sim U(0, 1)$을 따른다.

[3단계] 새로운 하모니 처리 : 즉흥이라 불리는 새로운 하모니를 생성한다. 새 하모니 벡터는 $\mathrm{x}' = (x_1', x_2', \cdots, x_N')$이며 다음과 같은 과정을 거친다.

[4단계] 하모니 메모리 업데이트 : 생성된 하모니 벡터 $\mathrm{x}' = (x_1', x_2', \cdots, x_N')$의 적응도가 최악의 하모니의 적응도보다 좋을 경우 HM안의 최악의 값과 교체된다.

[5단계] 정지 기준 확인 : 즉흥의 최대 수에 도달하면 종료한다.

HS의 HMCR과 PAR 파라미터들은 반복적으로 개선된 최적해와 지역 해를 찾는 방법을 돕는다. PAR과 bw는 HS의 수행도에 엄청난 영향을 가진다. 그러므로 좋은 튜닝을 위해 이 두 파라미터는 매우 중요하다. HS는 이러한 조율을 통해 최적 해를 찾아 나가는 것으로 PAR의 값을 좀 더 세밀하게 조정해 최적화된 bw의 값을 구하는 개선한

<Table 2> Harmony Search Algorithm

```
begin
Set a parameter such as HM, HMCR, PAR.
for each i∈[1, N] do
i is number of HM.
if U(0, 1) ≤ HMCR then /*memory consideration*/
begin
x = x_i^j then (i~U(1, ···HMS)
if U(0, 1) ≤ PAR then /*pitch adjustment*/
begin
x_i = x_i ± r×bω then (r~U(0, 1), bω
is the amount of maximum change in pitch adjustment
else /* random selection */
x_i^j = LB_i + r×(UB_i − LB_i)
end
```

IHS(Improved Harmony Search Algorithm)을 쓴 논문도 존재하며[8] PAR값의 best값을 이용해 HS를 개선한 GHS (Global-best harmony search)를 연구한 논문처럼 다양한 발전을 이루고 있다. 또한 HM의 개수를 HMPR(Harmony Memory Preservation RATE)라는 유지 비율에 의해 결정하여 비율 내의 좋은 결과는 유지하고 그 외의 결과는 HM을 수정하여 성능을 향상시키기도 하였다[7].

2.3 기존 알고리즘의 문제점

2.3.1 정확성

메타 휴리스틱 알고리즘은 한정된 시간 내에 목적함수를 만족시키도록 하는데 최적화문제의 최적 해를 구하는 일은 불가능에 가까운 일이므로 최적 해와 가장 근사한 지역 해를 찾게 된다. 기존 알고리즘들은 초기의 랜덤 값을 바탕으로 더 나은 값을 찾아가게 된다. 초기의 랜덤 값을 보정해주지 못하면 초기의 랜덤 값은 잘못된 방향으로 목적함수를 이끌게 되고 정확하지 못한 결과를 초래하게 된다. 또한 메타 휴리스틱 알고리즘들은 반복적 개선에 의하여 지역 해에 빠져버리는 단점을 개선하도록 확률적인 이동의 여지를 두고 있다. 하지만 확률적인 이동 역시 임의의 함수를 이용한 방법으로써 보정이 없이 진행되므로 지역해가 많아질수록 최적 해를 구하는 일이 어려워지며 정확성은 더욱 낮아지게 된다. SA의 경우 초기의 랜덤 값으로 시작하여 $\exp\left(-\frac{\Delta}{T}\right)$의 확률을 가지고 다른 값으로 이동할 수 있도록 하였다. 그리고 반복횟수가 늘어날수록 $\exp\left(-\frac{\Delta}{T}\right)$의 확률은 점점 줄어들게 되며 알고리즘의 후반에서는 다른 값으로 이동하기 힘들게 설계 되어있다. 하지만 초기에 무작위의 확률로 더 안 좋은 값으로 갈수가 있으며 후기에는 지역 해에서 고착되어 버릴 수도 있다는 단점이 있다.

2.3.2 메모리와 시간

메타 휴리스틱 알고리즘은 최적 해를 찾아가기 위해 수많은 변수를 입력하고 수정하고 출력하게 된다. 이로 인해 컴퓨터의 많은 메모리를 차지하고, 값을 구하는데 프로그램 사용시간이 길어진다.

본래의 HS 방법은 멀티모델의 최적화 문제를 다루는데 제한적이다. 결점을 보완하고자 메모리의 요소에 효율적인 다양성 유지정책을 적용한 방법을 제안하였지만 Harmony Memory의 데이터가 늘어날수록 처리속도가 늦어지는 문제가 생기게 되었다[3, 6]. NP문제는 해결하기 위해 일일이 모든 값을 구해보아야 되어서 많은 사이클 타임을 요구하여 이는 점점 지수적인 시간을 필요로 하게 되어 결국에는 매우 긴 시간이 걸리게 된다. 이변수 다항식은 일변수 다항식에 비해 범위가 곱으로 늘어남에 따라 처리해야 할 메모리가 증가하고 시간을 오래 걸리게 하는 단점으로 이어졌다.

2.3.3 기억성

NP문제는 많은 지역 해가 존재하기 마련이다. 많은 지역 해 중 한 곳에 갇히게 되므로 이를 초기의 랜덤 값과 확률적인 이동을 통하여 보정을 하고 있다. 초기 값과 확률 적인 이동은 전부 무작위로 정해지게 되므로 자유도가 높은 장점을 가지지만 반대로 올바른 길을 벗어나게 만들 수도 있다. 최적 해를 향한 올바른 길로 찾아가다가 무작위로 올바르지 않은 값이 도출될 수도 있다. 긴 시간에 걸쳐 얻어낸 결과가 최적해가 아닐 경우도 많아서 수많은 횟수의 실험을 필요로 하게 되지만 실험의 처음에 가지는 값은 랜덤 값을 받고 다시 처음의 상태로 돌아와서 실험을 진행하면서 더 큰 시간적 손실을 초래한다. 확률의 독립시행처럼 이전의 실험의 결과 값을 다음 실험에서 사용하지를 못하게 되는 것이다. 본 연구의 알고리즘에서는 이와 같은 오류를 막고자 알고리즘 자체에 지난 실험의 최적의 결과와 초기 랜덤 값을 조합하여 사용하도록 설계하였다[5].

3. 개발 알고리즘

3.1 이론

첫 번째로 단일 개체를 이용하여 값을 평가하는 것 보다는 HS의 Harmony Memory처럼 여러 개의 개체를 이용하여 값을 구할 경우에 정확도가 증가하였다. 무조건 최적 해를 찾을 수 있는 알고리즘을 구현하면 좋지만 최적 해에 정확성이 높아질수록 시간도 오래 걸리는 점을 고려할 필요가 있다. 메타 휴리스틱 알고리즘의 문제점인 Random은 한 개의 개체를 이용하여 최적 해를 찾기보다는 여러 개의 개체로 랜덤의 문제점을 줄였다.

두 번째로 방향성을 도입하여 최적 해를 찾도록 하였다. 랜덤의 초기 값을 가지고 방향을 가질 수가 없고 원점으로부터 방향벡터는 의미가 없어 곡선의 기울기를 구하는 미분의 개념에서 방향벡터를 구하도록 하였다. 랜덤의 초기 값을 랜덤한 방향으로 미세하게 이동을 시키게 되면 두 점의 좌표를 가지게 된다. 두 점의 좌표를 가지는 것은 두 점을 지나는 한 직선을 알게 되는 것이고 직선은 기울기와 벡터를 가지고 있으므로 방향을 알게 된다. 예를 들어, 목적함수가 최솟값을 원할 때 x와 μ의 변화량으로 인해 z값이 줄어들 경우 벡터는 옳은 방향과 유사하게 가리키게 되는 것이므로 알고리즘은 진행한다. 또한 z값이 늘어나면 벡터는 옳지 않은 방향으로 가는 것으로 보고 x와 μ의 변화량의 반대 방향으로 알고리즘을 진행한다. 이후 옳은 방향으로 갈 경우 확률을 주어 다시 같은 방향으로 진행하게 하여 z값을 비교하여 방향이 옳을 경우 진행하나 아닐 경우 새로운 방향을 지정하게 한다. 마찬가지로 z값이 늘어나 방향을 벡터의 -방향으로 진행하게 될 경우에 여전히 z값이 줄어들지 않을 경우에도 새로운 방향을 지정하도록 한다. 이를 단계로 설명하면 아래와 같다.

0단계 : 기존의 기억이 있는지 탐색. 없을 경우 1단계. 있을 경우 2단계

1단계 : 임의의 최초 값을 생성한다. 목적함수는 최솟값을 구하는 문제로 정한다.

2단계 : 방향을 생성하기 위한 최초 값 근처에 임의의 점을 생성한다.

3단계 : 두 점을 이용하여 최초 값의 방향을 확인한다. z값을 비교하여 방향이 옳을 경우이다.

- 3.1단계 : 같은 벡터로 진행하였을 때
  - 3.1.1단계 : 일정 확률일 경우 같은 벡터로 진행하며 아닐 경우 멈춘다.
  - 3.1.2단계 : z값이 더 높아지면 새로운 방향 점을 생성한다.
- 3.2단계 : -벡터로 진행시킨 경우
  - 3.2.1단계 : z값이 더 낮아지면 같은 벡터로 진행한다.
  - 3.2.2단계 : z값이 더 높아지면 새로운 방향 점을 생성한다.

4단계 : 기존의 나아진 결과를 가진 점을 새 점으로 삼고 2단계로 돌아간다.

5단계 : 반복횟수가 끝날 경우 그룹 내의 최솟값을 최적해로 지정한다.

세 번째로 지난 실험의 최적의 결과 값을 기억해두고 다음 실험에서 사용할 수 있도록 하였다. 메타 휴리스틱 알고리즘은 시간의 제약이 존재하기 마련이고 일정 시간이 지나면 알고리즘은 멈추게 된다. 긴 시간을 들여 학습한 결과를 다음 실험에서 사용하여 초기의 랜덤 값에 의존하여 정확성을 낮추는 실수를 범하지 않도록 하였다. 즉, 지난 실험의 최적 해를 기억해두고 다음 실험에서 최초 값에 포함하도록 하였다. 새 알고리즘의 경우 개체수가 크므로 한 개의 지난 실험의 최적 값은 그룹 내에서 큰 비중을 차지 않을 것으로 예상된다.

3.2 알고리즘

이를 도식화한 알고리즘은 아래와 같다.

<Table 3> Developed Algorithm

```
begin
Get an initial group number g_n;
Get an initial random group g;
for i=1 to Iteration do
Get an initial value eta;
Get an initial new random group ng; //ng is g plus eta
delta = ng-g;
Find the right direction
if delta < 0
Get an initial random value rv;
if rv < 0.05
Get an initial new random group nng; nng is ng plus eta
delta = nng-ng;
Find the right direction
if delta > 0
nng ← ng
Get an initial value eta;
else
Get an initial new random group ng; //ng is g minus eta
delta = ng-g;
Find the right direction
if delta > 0
ng ← g
Get an initial value eta;
end for
return ng
```

4. 연구방법

4.1 테스트 함수

사용할 테스트 함수는 다음 표와 같다. $(x, y) = 1000$, $(x, y) = -1000$이고 해의 범위는 없으며 $-3 \le x, \mu \le 3$로 주어졌다. 일종의 NP문제를 가진 함수이다. 함수는 지역 해를 무수히 많이 포함하고 있으며 최솟값과 최대값을 포함하고 있다. 하지만 Sin과 Cos함수를 내장한 주기 함수이며 범위 내에 지역해의 개수가 무한하게 늘어남으로 메타 휴리스틱 알고리즘들을 이용한 최적해 찾기의 정확도와 시간을 비교하기에 용이하다. 본 함수의 최적 해는 아직 알려지지 않았다.

<Table 4> Test Function

$$
\begin{aligned}
f(x, y) = \; & \log\left(\sin\left(\frac{y}{x-1000}\right)^2 + (\sin(x+1000) - y^2)^2\right) \\
& - \log\left(\sin\left(\frac{x}{y-1000}\right)^2 + (\sin(y+1000) - x^2)^2\right) \\
& + \log\left(\sin\left(\frac{y}{x-1000}\right)^2 + (\cos(x+1000) - y^2)^2\right) \\
& - \log\left(\sin\left(\frac{x}{y-1000}\right)^2 + (\cos(y+1000) - x^2)^2\right)
\end{aligned}
$$

4.2 연구 프로세스

이변수 다항식과 시뮬레이티드 어닐링 알고리즘, 하모니 서치 알고리즘, 그리고 새로 개발한 알고리즘을 접목하여 결과 값을 구하도록 한 코드를 통하여 자체 제작한 해를 찾는 함수인 formemo(이변수 다항식의 x, μ을 대입하셨을 때의 z값을 구하는 함수)로 결과 값이 나오도록 하였다. SA는 초기 그룹의 개체 수를 1개로 그리고 HS의 HM은 20으로 설정하였으며 새 알고리즘의 개체 수는 100으로 설정하였다. SA는 단일 개체의 온도 변동을 통한 상태를 확인하는 알고리즘으로 한 개체가 최적 해를 향해 나아가는 알고리즘이기 때문에 개체 수를 1로 하였으며 HS는 상대적으로 긴 시간을 고려하며 멀티 모델에 불리하여 개체 수를 높지 않은 20으로 설정하였다[3, 6]. 새 알고리즘은 그룹의 개체 수를 100과 1000으로 나누어 진행하였는데 그룹의 개체 수에 따른 정확성을 확인하기 위해서이다. 1000개를 이용하였을 때의 반복횟수를 늘려서 반복횟수가 늘어남에 따른 런타임과 정확성 향상을 확인하고자 한다. 그리고 알고리즘의 학습 효과를 확인하기 위하여 짧은 런타임의 작은 개체 수를 포함한 실험을 여러 차례 반복 실험한다. 구해진 초기의 실험값과 최종 실험값의 비교 분석을 통해 학습 효과가 영향을 주는지 확인한다.

테스트 함수는 미지의 최적 해를 가지고 있으며 지역해가 너무 많은 문제점을 가지고 있다. 그래서 대외적으로 알려져 있는 다양한 함수들 중 Michalewicz Function과 Rosenbrock's Banana Function 그리고 Bukin Function을 통하여 최적 해를 구하는 정확성과 소요시간을 확인하고자 한다. 세 함수는 지역 해가 많지 않아서 최적 해를 구하기 쉽고 최적화 알고리즘을 시험해볼 용도로 제작된 함수이므로 채택하였다. Iteration은 10,000회를 반복하여 테스트하며 테스트 결과 값이 기대 이하일 경우 반복 횟수를 늘려서 결과를 확인하였다.

5. 연구결과

5.1 테스트 함수

알고리즘을 이용해 반복횟수(Iteration)를 동일하게 하여 실험을 진행하였다. 반복횟수는 1,00,000번을 진행하였으며 SA는 단일 개체, HS의 HM은 20, 새로 구현한 알고리즘의 그룹은 100으로 진행하였다.

<Table 5> Comparison with Results of Test Function

Iteration	SA(1)		HS(20)		New(100)	
	Value	Time	Value	Time	Value	Time
1,000,000	-29.3	4910	-53.1	7849	-59.9	1849

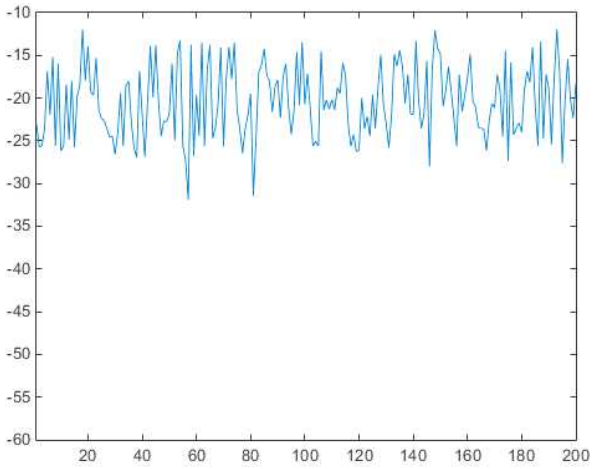
각 그룹의 괄호 안은 시작시의 개체 수를 의미하며 시간의 단위는 초(Second)이다. 결과는 같은 조건 아래 여러 번의 실험을 통하여 나온 값 중 가장 낮은 값과 그에 따른 시간을 기록한 결과로 SA의 경우 -29.3을 가장 낮게 가졌으며, 걸린 시간은 4910초이다. HS는 가장 작은 값으로 -53.1을 구하였으나 시간은 7849초가 걸려 SA보다 긴 시간이 걸렸다. 그리고 새 알고리즘으로 구한 값으로는 -59.9가 최저 값으로 나왔으며 몇몇 값은 HS의 -53.1보다 좋지 않은 결과를 가졌다. 그러나 시간은 1849초로 훨씬 짧은 시간 내에 최적 해를 구하는 것을 보여주었다.

<Table 6> Result of Increasing Number of Repetition and Entity

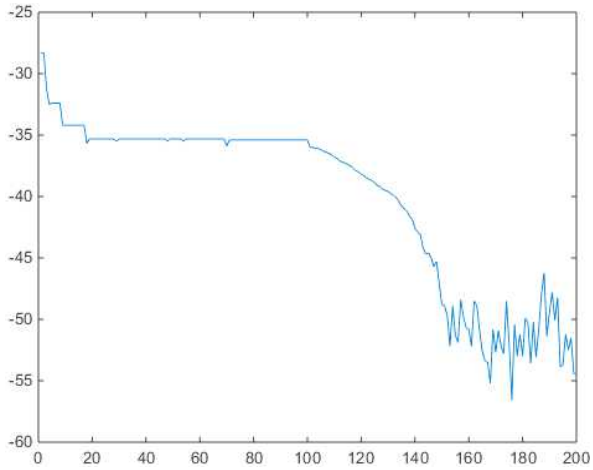
Iteration	New(1000)	
	Value	Time
1,000,000	-68	7782
10,000,000	-99	13678

알고리즘의 소요 시간이 상대적으로 짧아 반복횟수를 각각 1배, 10배 늘리고 초기 그룹의 개체 수는 각각 1,000번으로 늘려서 실험을 진행해 본 결과이다. 반복횟수를 10배 늘린 경우 Value가 매우 좋은 값을 구하였지만 반복횟수와 그룹이 커지면서 시간 또한 길어졌다. 구한 값이 테스트 함수의 최적의 값은 아니더라도 결과는 초기 그룹의 수에 비례해 정확성이 늘어나는 것을 확인할 수 있으며 메모리가 커져 시간이 증가하는 것을 보여준다.

<Figure 1>은 새로운 알고리즘의 학습 효과를 사용하여 같은 초기 값의 개수(100개)와 동일한 반복횟수(10,000회)의 실험을 200번 진행하였다. 그룹의 크기가 커질수록 시간이 길어져 짧은 실험의 반복으로 효과를 확인하였다. 1회 차에는 학습하지 않은 상태에서 시작하였으며 2회 차부터는 전 실험의 최적해 값을 초기 값의 1개로 가지고 시작하였다. 결과 값은 <Figure 1>과 같이 학습 효과를 사용하지 않을 경우에 비해 학습 효과를 사용한 알고리즘은 꾸준한 결과 값의 상승을 볼 수 있고 이는 학습 효과가 작용을 하고 있다.



<Figure 1> 200 Iterations of without Training



<Figure 2> 200 Iterations of Training

5.2 NP-hard 함수

5.2.1 Michalewicz Function

<Table 7>은 $f(x) = -\sum_{i=1}^{d} \sin(x_i) \sin^{2m}(\frac{ix^2}{\pi})$을 가지는 그림과 같은 그래프를 가진 Michalewicz function의 최적 해를 구한 결과이다. Michalewicz function의 해는 익히 알려져 있으며 d(dimensions)의 값이 2일 경우 최적의 해는 -1.8013이다. 각각의 메타 휴리스틱 알고리즘으로 최적 해를 구하였을 때 SA가 시간대비 가장 좋은 결과값을 보여주었지만 개발 알고리즘은 최적 해를 찾아 좋은 결과값을 도출하였다.

<Table 7> Michalewicz Function

Iteration =10000	SA(1)		HS(20)		New(100)	
	Value	Time	Value	Time	Value	Time
Best	-1.7994	0.5607	-1.7093	1.7818	-1.8013	3.5109
Average	-1.7641	0.56465	-0.9420	1.76998	-1.7582	3.49774

5.2.2 Rosenbrock's Banana Function

<Table 8>은 $f(x) = \log(1+(1-x)^2+100(y-x^2)^2)$을 사용하여 최적 해를 구한 결과로 식은 Rosenbrock's Banana Function의 변형 형이다. 그래프를 그려 보면 길고 좁은 포물선 모양의 골짜기가 나타나며 최적 해는 0이 된다. 실험의 경우에는 SA와 HS는 매우 좋지 못한 결과를 얻어냈지만 새 알고리즘은 최적 해에 근사한 값을 얻어낼 수 있었다. 또한 SA와 HS의 값이 너무 좋지 않아 반복 횟수를 1,000,000번으로 늘려서 실험한 결과 세 알고리즘 모두 최적해인 0을 구하였으나 SA는 3843.1초, HS는 5919.5초, 새 알고리즘은 1052.2초가 걸렸다.

<Table 8> Rosenbrock's Banana Function

Iteration =10,000	SA(1)		HS(20)		New(100)	
	Value	Time	Value	Time	Value	Time
Best	13.108	0.44285	13.064	1.6161	0.0001	2.5917
Average	13.1355	0.4495	13.7055	1.6361	0.0013	2.6339

5.2.3 Bukin Function

<Table 9>는 $f(x) = 100\sqrt{|\mu-0.01x^2|}+0.01|x+10|$을 사용하여 최적 해를 구한 결과이다. 식은 Bukin Function으로 최적 해는 알려져 있으며 값은 0이다. 세 알고리즘 모두 최적 해에 근사한 값이었으며 SA가 최단 시간이었고, 개발된 알고리즘이 가장 정확한 값을 보여주었다.

<Table 9> Bukin Function

Iteration =10,000	SA(1)		HS(20)		New(10)	
	Value	Time	Value	Time	Value	Time
Best	0.0154	0.2057	0.0057	0.6985	0.0009	1.4723
Average	0.465	0.2087	2.2142	0.7201	0.0052	1.4887

6. 결론 및 향후과제

모든 종류의 문제에 다른 알고리즘보다 우수한 알고리즘은 존재하지 않는다는 NFL(No Free Lunch)에도 불구하고 NP문제를 해결하기 위한 많은 종류의 메타 휴리스틱 알고리즘의 개발되어 왔다. 개발한 알고리즘이 모든 종류의 문제에 우수하지 못할지라도 한 종류의 문제에 관하여 우수하다면 의미가 있다.

개발된 알고리즘을 NP-hard문제의 함수에서 값을 비교해 보았다. SA가 가장 좋지 못한 결과를 보였고 HS가 가장 오래 걸렸으며 개발한 알고리즘은 상대적으로 좋은 결과를 얻었다. 초기의 그룹 개체 수와 반복 횟수를 증가하여 최적 해에 더 가까운 값을 얻는 결과를 얻었다. 학습 효과를 확인하기 위한 실험 조건으로 같은 실험을 200번 반복하여 초기의 값보다 점점 더 좋은 값을 찾아 나가는 결과를 보이며 알고리즘에 도입한 학습 효과가 잘 작동되는 것을 확인하였다. 그리고 최적 해를 구하기 위한 여러 함수들 중 Michalewicz function, Rosenbrock's banana function, Bukin Function을 사용하여 SA와 HS 그리고 개발한 알고리즘과의 비교를 실시하였다. 함수들은 전부 이변수 다항식으로써 최적 해를 구하는데 까다로움이 존재하였지만 새 알고리즘은 상대적으로 정확한 값을 보여주며 좋은 결과를 얻을 수 있었다. 마지막으로 반복횟수가 10,000번 일 경우 SA와 HS의 시간이 새 알고리즘에 비해 짧았지만 반복 횟수를 1,000,000번으로 늘렸을 때에는 새 알고리즘이 더욱 짧은 시간이 걸리는 것을 확인하였다.

향후 과제는 많은 개체를 투입하여 최적의 투입개수를 구하는 연구가 필요하다. 지역 해의 개수가 무수히 많은 멀티 모델인 이변수 다항식을 테스트 함수로 사용하였지만 다차원 방정식과 외판원 문제(TSP)와 Global-best harmony search와 같이 여러 종류의 함수에서의 결과 값 비교가 필요하다.

Acknowledgement

This paper was supported by Kumoh National Institute of Technology.

References

[1] Bianchi, L., Dorigo, M., Gambardella, L.M., and Gutjahr, W.J., A Survey on Metaheuristics for Stochastic Combinatorial Optimization, *Natural Computing*, 2009, Vol. 8, pp. 239-287.

[2] Dorigo, M., Maniezzo, V., and Colorni, A., The Ant System : Optimization by a Colony of Cooperating Agents, Systems, Man, and Cybernetics, Part B: Cybernetics, *IEEE Transactions on*, 1996, Vol. 26, No. 1, pp. 29-41

[3] Gao, X.Z., Wang, X., and Ovaska, S.J., Harmony Search Methods for Multi-Modal and Constrained Optimization, *Studies in Computational Intelligence*, 2009, pp. 39-51.

[4] Geem, Z.W., Kim, J.H., and Loganathan, G.V., A New Heuristic Optimization Algorithm : Harmony Search,

Simulation, 2001, Vol. 76, pp. 60-68.

[5] Kattanl, A., Abdullah, R., and Salam, R.A., Harmony Search Based Supervised Training of Artificial Neural Networks, *International Conference of Intelligent Systems, Modelling And Simulation*, 2010, pp. 105-110.

[6] Kim, M.K., Ko, K.E., and Sim, K.B., Harmony Search Methods for Multi-Modal Optimization Using Optimal Vectors, *Journal of Korea Institute of Electronics and Information Engineers*, 2010, pp. 2235-2238.

[7] Lee, S.K., Ko, K.E., and Sim, K.B., Study on Improvement of Convergence in Harmony Search Algorithms, *Journal of the Korean Institute of Intelligent Systems*, 2011, Vol. 1, No. 21, pp. 31-34.

[8] Mahdavi, M., Fesanghary, M., and Damangir, E., An Improved Harmony Search Algorithm for Solving Optimization Problems, *Applied Mathematics and Computation*, 2007, Vol. 188, No. 2, pp. 1567-1579.

[9] Yoon, B.S., Meta Heuristics that Mimics Natural Phenomenon, *Korean Institute of Industrial Engineers*, 2011, pp. 1-10.

[10] Yoon, H.Y., Jung, S.J., and Kim, K.S., Harmony Search Combined with Art Colony Optimization, *Journal of Society of Korea Industrial and Systems Engineering*, 2013, pp 1067-1074.

ORCID

Sung-Ho Chang | https://orcid.org/0000-0002-9758-2902
Moonsoo Kwon | http://orcid.org/0000-0002-8980-152X
Geuntae Kim | http://orcid.org/0000-0002-9166-7702
Jonghwan Lee | http://orcid.org/0000-0001-9630-5236