https://doi.org/10.5392/JKCA.2020.20.11.636

외판원 문제를 위한 난수 키 표현법 기반 차분 진화 알고리즘

Differential Evolution Algorithm based on Random Key Representation for Traveling Salesman Problems

이상욱
목원대학교 정보통신융합공학부

Sangwook Lee(slee@mokwon.ac.kr)

요약

차분 진화 알고리즘은 연속적인 문제 공간인 실수 최적화 문제를 해결하기 위해 개발된 메타휴리스틱 기법 중에 하나이다. 본 연구에서는 차분 진화 알고리즘을 불연속적인 문제 공간인 외판원 문제 해결에 사용하기 위하여 차분 진화 알고리즘에 난수 키 표현법을 적용하였다. 차분 진화 알고리즘은 실수 공간을 탐색하고 오름차순으로 정렬된 해의 인덱스의 순서를 도시 방문 순서로 하여 적합도를 구한다. TSPLIB에서 제공하는 표준 외판원 문제에 적용하여 실험한 결과 제안한 난수 키 표현법 기반 차분 진화 알고리즘이 외판원 문제 해결에 가능성을 가지고 있음을 확인하였다.

■ 중심어 : | 차분 진화 알고리즘 | 난수 키 표현법 | 외판원 문제 | 연속 공간 탐색 | 불연속 공간 문제 |

Abstract

The differential evolution algorithm is one of the meta-heuristic techniques developed to solve the real optimization problem, which is a continuous problem space. In this study, in order to use the differential evolution algorithm to solve the traveling salesman problem, which is a discontinuous problem space, a random key representation method is applied to the differential evolution algorithm. The differential evolution algorithm searches for a real space and uses the order of the indexes of the solutions sorted in ascending order as the order of city visits to find the fitness. As a result of experimentation by applying it to the benchmark traveling salesman problems which are provided in TSPLIB, it was confirmed that the proposed differential evolution algorithm based on the random key representation method has the potential to solve the traveling salesman problems.

■ keyword : | Differential Evolution Algorithm | Random Key Representation | Traveling Salesman Problems | Continuous Space Search | Discrete Space Problem |

I. 서 론

외판원 문제(Traveling Salesman Problem, TSP)는 가장 유명한 조합 최적화 문제 중 하나로 기사의 여행 문제를 해결하는 데 관심이 있었던 오일러(Eulerus)가 1759년에 최초로 제안하였다[1]. 컴퓨터 과학, 공학, 운영 연구, 이산 수학, 그래프 이론 등의 다양한 분야에서 기본적인 문제로 다루어지고 있는 외판원 문제는 모든 도시를 정확히 한 번 여행하고 최초 출발한 도시로 돌아가는 총 이동 거리를 최소화하는 문제로 간단히 설명할 수 있다. 외판원 문제는 거리 행렬의 구조에 따라 대칭(symmetric) 및 비대칭(asymmetric)의 두 그룹

접수일자 : 2020년 11월 02일
수정일자 : 2020년 11월 20일
심사완료일 : 2020년 11월 20일
교신저자 : 이상욱, e-mail : slee@mokwon.ac.kr

으로 분류할 수 있다. 대칭 외판원 문제는 두 도시 사이의 거리가 출발지와 도착지 여부와 상관없이 일치하는 값을 가지며, 반대로 비대칭 외판원 문제는 두 도시 사이의 거리가 출발지, 도착지에 따라 서로 다른 값을 가진다. 본 연구에서는 대칭 외판원 문제에 대해서만 다룬다.

지난 수십년 동안 외판원 문제는 최적화 분야에서 상당한 관심을 받았으며 문제를 해결하기 위해 분기 및 경계(Branch and Bound)[2], 절단면(Cutting Planes)[3], 2-opt[4], 입자 군집 최적화(Particl Swarm Optimization)[5], 시뮬레이티드 어닐링 (Simulated Annealing)[6], 개미 군집(Ant Colony)[7], 신경망(Neural Network)[8], 타부 서치 (Tabu Search)[9], 유전 알고리즘 (Genetic Algorithm)[10-12], 입자 군집 최적화(Particle Swarm Optimization)[13], 차분 진화 알고리즘 (Differential Evolution)[14]와 같은 다양한 접근 방식이 제안되었다. 이러한 방법론 중 일부는 최적해를 찾는 정확한 방법(Exact Method) 이고 일부는 근사해를 찾는 발견적인 기법(Heuristic Method)이다. TSP는 도시의 수가 많아 복잡도가 높아지면 정확한 방법으로는 주어진 시간 내에 풀 수 없는 경우가 대부분이므로 최적해를 찾지는 못하더라도 주어진 시간 내에 근사해를 찾기 위한 휴리스틱 방법이 주로 사용된다.

본 연구에서는 연속공간 탐색 기법인 차분 진화 알고리즘을 조합 최적화 문제인 외판원 문제 해결에 적용하기 위해 복잡한 연산을 추가한 기존 연구와는 달리 난수 키 표현법을 사용하여 기존의 연산만으로도 적용 가능하도록 설계한다.

II. 배 경

1. 차분 진화 알고리즘

차분 진화(Differential Evolution, DE)알고리즘은 비선형이고 미분 불가능한 연속 공간 함수를 최적화하기 위해 만들어진 해집단 기반 메타휴리스틱 검색 기법으로 1995년 스톤(Storn)과 프라이스(Price)에 의해 처음 소개되었다[15]. DE는 진화 과정을 기반으로 후보해를 반복적으로 개선하여 문제를 최적화하며 진화하는 방법으로 해집단에 속한 다른 해들의 정보를 사용한다. DE는 구조와 연산이 간단하고 수렴성이 뛰어나 매우 큰 문제 공간을 빠르게 탐색할 수 있으며 알고리즘을 실행하기 위해 사용하는 환경변수가 다른 메타휴리스틱 알고리즘에 비해 적어서 사용하기가 편리한 장점이 있다. 이러한 장점으로 인해 DE는 다중 모드 문제 해결에 대한 견고성을 보여주는 가장 다재다능하고 안정적인 해집단 기반 메타휴리스틱 검색 알고리즘 중 하나로 알려져 있다.

DE 검색 알고리즘의 검색 과정 흐름도는 [그림 1]과 같다. 초기 해집단을 랜덤하게 생성하며 개별 해를 벡터(Vector)로 취급한다. 개별 벡터를 진화시키기 위해 돌연변이(Mutation) 과정을 통해 시행 벡터(Trial Vector)를 생성한다. 개별 벡터와 시행 벡터 간의 교차(Crossover) 연산을 수행하여 자식 벡터를 생성하고 생성한 자식 벡터를 평가한 후 자식 벡터가 개별 벡터보다 성능이 우수할 경우 개별 벡터를 자식 벡터로 대치하는 과정으로 진행된다. 이러한 과정은 종료 조건을 만족할 때까지 계속 진행된다.

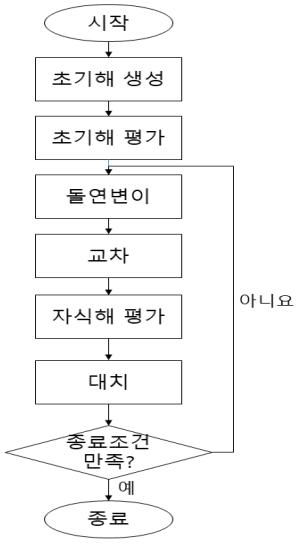

그림 1. 차분 진화 알고리즘 검색 과정 흐름도

돌연변이 연산에서 시행 벡터를 생성하는 방법은 다음과 같이 크게 3가지 방법으로 나뉜다.

$$\text{DE1: } \overline{V}(t) = \overline{X}_{r1}(t) + F \cdot \left(\overline{X}_{r2}(t) - \overline{X}_{r3}(t)\right) \quad (1)$$

$$\text{DE2: } \overline{V}(t) = \overline{X}_{i}(t) + F \cdot \left(\overline{G}(t) - \overline{X}_{i}(t)\right) + F \cdot \left(\overline{X}_{r1}(t) - \overline{X}_{r2}(t)\right) \quad (2)$$

$$\text{DE3: } \overline{V}(t) = \overline{G}(t) + F \cdot \left(\overline{X}_{r1}(t) - \overline{X}_{r2}(t)\right) \quad (3)$$

여기서 $\overline{V}$는 시행 벡터, $\overline{X}$는 개별 벡터, $\overline{G}$는 현재 세대까지 발견한 최적해, t는 세대, F는 [0, 2] 사이의 값을 갖는 돌연변이 변수, i는 교차 연산을 할 개별 벡터 인덱스, r_1, r_2, r_3는 해집단에서 교차 연산을 할 개별 벡터 인덱스와 서로 겹치지 않게 랜덤하게 선택된 개별 벡터의 인덱스이다.

DE1은 DE의 초기 버전으로 랜덤성에 기반하여 넓은 해 공간에서 지역 최적해에 잘 빠지지 않고 탐색이 가능하다는 특징을 가지고 있어 DE를 적용할 때 가장 많이 사용되는 기법이다. DE3는 현재까지 발견한 최적해 주변을 위주로 탐색하는 방법으로 DE 중에서 가장 빠른 수렴 특성을 보이나 지역 최적해에 빠질 위험성을 가지고 있다. DE2는 DE1의 랜덤 탐색과 DE3의 발견한 최적해 주변 탐색을 겸한 알고리즘으로 지역 최적해에 빠질 위험성을 줄이면서 수렴성을 강화하는 탐색 방법이다.

해집단에 있는 각 개별 벡터는 각각 별개의 돌연변이 연산을 통해 시행 벡터를 생성하며 이렇게 생성한 시행 벡터와의 [그림 2]와 같은 교차 연산을 통해 자식 벡터를 생성한다. 자식 벡터 각 요소는 [0, 1] 사이의 난수를 발생시켜 그 값이 CR보다 작으면 개별 벡터의 요소를 가져오며, 반대의 경우면 시행 벡터의 요소를 가져온다. 여기서 CR은 [0, 1] 사이의 값을 갖는 교차 확률이다.

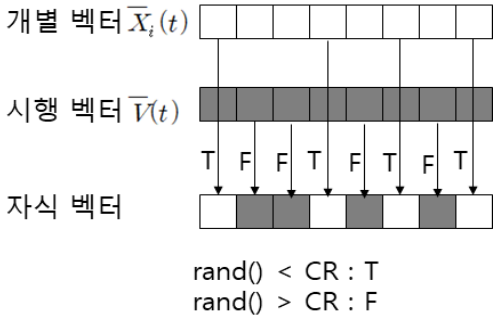

그림 2. 교차 연산 개념도

교차 연산을 통해 생성한 자식 벡터를 평가하고 만일 자식 벡터가 개별 교차 연산에 사용된 개별 벡터보다 더 우수하다면 해당 개별 벡터의 요소를 자식 벡터의 요소로 모두 대치한다.

차분 진화 알고리즘은 초기해를 생성하여 평가하는 과정은 유전 알고리즘과 동일하나 교차 및 대치하는 과정은 유전 알고리즘과 다소 차이가 있다. 유전 알고리즘의 경우 현재 해집단에서 부모해 둘을 선택하여 교차를 통해 자식해 둘을 만들어 부모를 대치한다는 측면에서 차분 진화 알고리즘과 차이가 있다.

2. 난수 키 표현법

난수 키 표현법(Random Key Representation)은 스케줄링 문제와 같이 순서 기반의 조합 최적화 문제(Combinatorial Problem)를 유전 알고리즘(Genetic Algorithms)으로 접근할 때 수리(Repair) 연산을 고려하지 않고 다양한 교차 및 돌연변이 연산을 적용하고자 개발된 표현(Representation) 기법이다[16]. 난수 키 표현법을 사용하면 불연속적인 탐색 공간을 연속 공간 탐색으로 변환할 수 있다. 난수 키 표현법의 사용법은 [그림 3]과 같다.

순서 기반 조합 최적화 문제의 차원(Dimension) 크기만큼 난수를 차원의 인덱스를 붙여 생성한다. 생성한 난수를 오름차순 또는 내림차순으로 정렬한다. 난수 생성 시 붙인 차원의 인덱스를 정렬한 순서대로 나열하여 해를 해석한다.

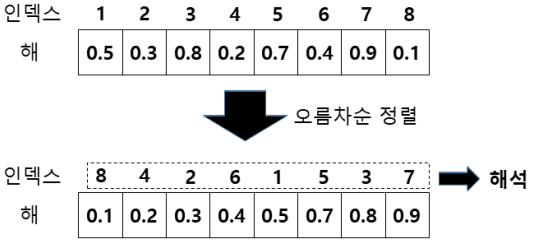

그림 3. 난수 키 기반 표현 개념도

3. 외판원 문제

외판원 문제 (Traveling Salesman Problems)는 주어진 도시를 한번씩 모두 방문하고 출발점으로 돌아오

는 경로 중 최소 거리를 가지는 경로를 찾는 문제이다. 예를 들어 어떤 외판원이 n개의 도시를 방문할 계획을 수립하고 있다고 가정하자. 각 도시는 다른 모든 도시와 도로로 연결되어 있다. 출장 비용을 최소로 줄이기 위하여 외판원이 거주하고 있는 도시에서 각 도시를 한 번씩만 방문하고 다시 출발한 도시로 돌아오는 가장 최소 비용의 일주 여행 경로를 찾는 문제이다. 외판원 문제는 대표적인 조합 최적화 문제 중 하나로 도시의 수가 n이면 $(n-1)!$ 만큼의 경로가 발생하며 모든 경로 중 가장 짧은 경로를 찾는 문제이다. 이는 도시 수가 10개이면 3,628,800개의 경로 수가 나오며, 20개이면 2,432,902,008,176,640,000개라는 상상하기도 어려운 경로수가 나온다. 문제는 단순하지만 고려해야 할 정답의 수는 도시의 수가 증가하는 것에 비해 기하 급수적으로 늘어난다. 외판원 문제는 다항 시간에 해결할 수 있는 방법론을 아직 찾지 못했으며, 다항 시간에 해결하지 못한다는 것도 증명하지 못하여서, NP-hard 문제 집합에 속해있다.

외판원 순회 문제는 다음과 같이 그래프 이론의 용어로도 정의할 수 있다. 주어진 완전 그래프 G=(V, E)가, 연결되어 있고(connected) 가중치가 있는(weighted) 완전한(complete) 그래프라고 가정하자. 이 그래프에서 출발 정점에서 다른 모든 정점들을 방문하고 원래의 출발 정점으로 되돌아 오는 순환 경로들 중에서 가중치의 합이 최소가 되는 순환 경로를 찾는 문제가 된다.

III. 외판원 문제를 위한 난수 키 기반 차분 진화 알고리즘

본 장에서는 차분 진화 알고리즘을 난수 키 기반 표현법을 사용하여 외판원 문제에 적용하는 방법에 대해 [그림 1]의 흐름도에 따라 설명한다.

1. 해 표현 및 초기해 생성

주어진 외판원 문제의 도시 수 만큼의 [0, 1] 사이의 난수를 발생시켜 후보해로 정의한다. 이러한 후보해를 집단 수만큼 생성하는 [그림 1]의 '초기해 생성' 과정을 진행한다.

2. 초기해 평가

해를 오름차순으로 정렬하고 난수 생성 시 붙인 차원의 인덱스를 정렬한 순서대로 나열하여 해를 해석한다. [그림 4]에서 '평가에 사용되는 해'를 사용하여 적합도 값을 구한다.

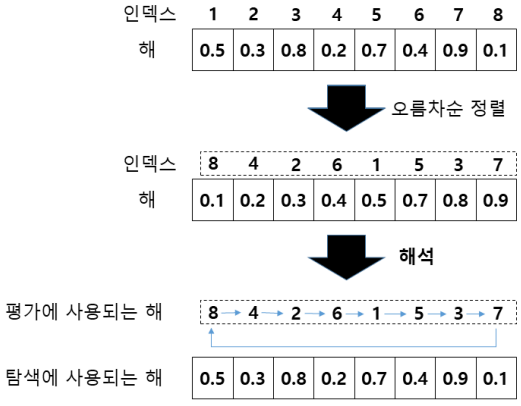

그림 4. 평가를 위한 해 해석의 예

적합도 값은 방문한 순서의 도시 간 거리를 모두 합하여 구한다. [그림 4]의 예에서는 (8, 4), (4, 2), (2, 6), (6, 1), (1, 5), (5, 3), (3, 7), (7, 8) 도시 간의 거리의 총합이 적합도 값이 된다.

탐색에 사용되는 해는 진화를 위한 과정인 돌연변이 및 교차를 위하여 오름차순 정렬 전의 상태로 돌린다.

3. 돌연변이

해를 진화시키기 위해 현재 해집단 각 개별해 별로 돌연변이 과정을 통해 시행 벡터를 생성한다. 시행 벡터를 생성하는 방법은 식(1), 식(2), 식(3)의 방법 중 하나의 방법을 사용한다.

만약 생성한 시행 벡터 각 요소들의 일부가 문제 공간인 [0, 1] 사이의 범위를 벗어날 경우에는 다음과 같은 과정을 거쳐 문제 공간 내에 들어오록 해를 수정한다.

- 시행 벡터의 요소들 중 최소값과 최대값의 차이가 1이 넘는 경우 모든 요소들을 최대값과 최소값의 차이만큼으로 나눈다.

- 시행 벡터의 요소들 중 최소값이 0보다 작을 경우 모든 요소들에게 (최소값) 만큼의 수를 더한다.
- 시행 벡터의 요소들 중 최대값이 1보다 클 경우 모든 요소들에게 (최대값 - 1) 만큼을 감한다.

4. 교차

돌연변이 과정을 통해 생성한 시행 벡터를 [그림 2]와 같은 균일 교차 과정을 통해 자식해를 생성한다.

5. 자식해 평가

돌연변이 및 교차 과정을 통해 생성한 자식해를 III.2에서 설명한 '초기해 평가'와 같은 방법을 평가하여 적합도를 구한다.

6. 대치

자식해의 적합도가 해집단 개별해의 적합도 보다 우수할 경우 개별해를 자식해로 대치한다.

V. 실험 결과

1. 실험 환경

실험환경은 Intel(R) Core(TM) i7-9700 CPU 3.00GHz 성능의 옥타코어 CPU와 8GB 메모리의 컴퓨팅 환경에서 Visual studio 2019 소프트웨어로 코딩하여 구현하였다. 최적의 변수 설정에서 차분 진화 알고리즘의 성능을 관찰하기 위하여 다음과 같은 실험을 진행하였다.

- 최적의 돌연변이 변수 설정 실험
- 최적의 교차 확률 설정 실험
- 다양한 외판원 문제에 대한 성능 실험

실험에 사용한 해집단의 수는 30으로 설정하였고 최대 세대(Generation) 수는 최적의 돌연변이 변수 및 교차 확률 설정 실험(burma14 문제)에서는 1,000으로 설정하였고, 다양한 외판원 문제에 대한 성능 실험에서는 10,000으로 설정하였다. 여기서 세대수란 진화계산 알고리즘이 평가 후 돌연변이 및 교차 과정을 거쳐 자식해를 생성하거나 해를 업데이트하는 과정의 반복 수를 뜻한다. 모든 실험은 10번 반복 실험하여 평균한 값을 작성하였다.

실험에 사용한 외판원 문제는 TSPLIB 웹 사이트에서 Symmetric traveling salesman problem(TSP)의 표준 문제(Benchmark Problems)를 활용하였다[17]. 특히 최적의 돌연변이 변수 설정 실험에서는 표준 문제 중에서 도시 수가 가장 적어 난이도가 가장 낮은 burma14 문제를 사용하였다.

2. 최적의 돌연변이 변수 설정 실험

외판원 문제에서 난수 키 기반 차분 진화 알고리즘의 성능을 극대화할 수 있는 최적의 돌연변이 변수(F)를 찾기 위하여 교차 확률(CR)을 [0, 1] 사이의 값 중 가운데인 0.5의 값으로 고정시키고 돌연변이 변수의 값을 0.2 만큼 변화시키며 [0, 2] 사이의 범위에서 실험하였다. 실험 결과는 [표 1]과 같이 DE1에서는 돌연변이 변수의 값이 0.5 일 때, DE2와 DE2에서는 0.8 및 1.0일 때 성능이 가장 우수하였다.

표 1. 돌연변이 변수 실험 (burma14 [$CR=0.5$])

돌연변이 변수(F)	DE1		DE2		DE3	
	평균 적합도 (거리)	최적해 발견 회수	평균 적합도 (거리)	최적해 발견 회수	평균 적합도 (거리)	최적해 발견 회수
0	32.2799	0	45.4639	0	38.1441	0
0.2	30.9464	8	31.5142	4	33.1451	1
0.4	30.8785	10	31.1099	7	31.7705	2
0.6	30.9406	9	30.9690	8	31.3258	7
0.8	31.0777	6	30.8785	10	30.8785	10
1.0	31.1117	6	30.8785	10	30.8785	10
1.2	31.3013	4	30.9943	7	31.0473	5
1.4	31.0213	8	31.2188	4	31.1902	4
1.6	30.969	8	31.9399	1	31.1647	6
1.8	31.04	7	31.3051	2	31.2725	5
2	31.1445	6	31.3834	1	31.2489	3

3. 최적의 교차 확률 설정 실험

표 2. 교차 확률 실험 (bruma14 [F 고정])

교차 확률 (CR)	DE1 ($F=0.4$)		DE2 ($F=0.9$)		DE3 ($F=0.9$)	
	평균 적합도 (거리)	최적해 발견 회수	평균 적합도 (거리)	최적해 발견 회수	평균 적합도 (거리)	최적해 발견 회수
0	36.1378	0	34.9870	0	36.7961	0
0.1	31.4742	5	31.3815	4	31.946	3
0.2	30.9115	9	31.3815	4	31.3752	6
0.3	30.9115	9	30.9133	9	31.0346	7
0.4	30.9115	9	31.1158	8	31.0021	7
0.5	30.8785	10	30.8785	10	30.8785	10
0.6	30.9115	9	30.9363	9	30.8785	10
0.7	30.8785	10	30.8785	10	30.9396	9
0.8	30.8785	10	30.9115	9	30.8785	10
0.9	30.8785	10	30.8785	10	30.8785	10
1	49.4813	0	49.2278	0	49.1365	0

다음으로 최적의 교차 확률(CR)를 찾기 위하여 돌연변이 변수(F)를 DE1에서는 0.4, DE2와 DE3에서는 0.9의 값으로 고정시키고 교차 확률의 값을 0.1 만큼 변화시키며 [0, 1] 사이의 범위에서 실험하였다. 실험 결과는 [표 2]와 같이 3가지의 차분 진화 알고리즘에서 모두 [0.5, 0.9] 사이의 값에서 성능이 가장 우수하였다.

4. 다양한 외판원 문제에 대한 성능 실험

burma14에서의 실험을 통해 확인한 3개의 차분 진화 알고리즘에 대한 최적의 돌연변이 변수 및 교차 확률을 사용하여 다양한 외판원 문제에 적용하여 성능을 테스트 해보았으며 그 결과는 [표 3]과 같다.

표 3. 외판원 문제 성능 ($CR=0.5$)

외판원 문제	DE1 ($F=0.4$)		DE2 ($F=0.9$)		DE3 ($F=0.9$)	
	평균 적합도 (거리)	발견한 최적해	평균 적합도 (거리)	발견한 최적해	평균 적합도 (거리)	발견한 최적해
burma14	30.8785	30.8785	30.8785	30.8785	30.8785	30.8785
ulysses16	74.0518	73.9876	74.0905	73.9876	74.8024	73.9876
ulysses22	76.4549	75.3097	76.1309	75.3097	76.5	75.9104
bays29	9,278.26	9,108.77	9,384.67	9,076.98	9,417.38	9,074.15
dantzig42	825.7562	720.732	890.2617	718.791	835.738	751.628
pr76	207,084	117,090	313,625	123,249	319,136	279,892
eil101	1,845.1	1,222.3	2,333.1	2,329.3	2525.6	2,373.7
ch150	42,521.5	40,865.5	42,654.3	40,515.9	42,314.4	40,078.3

3가지의 차분 진화 알고리즘 모두 도시 수가 가장 적은 문제인 burma14에서는 10번 모두 최적해인 30.8785를 찾았다. 그리고 ulysses16 문제에서는 최적해 73.9876을 3가지 알고리즘 모두 찾았으나 10번 모두를 찾지는 못하였다. ulysses22 문제는 DE1과 DE2는 최적해인 73.3097을 찾았으나 DE3는 찾지 못하였다. bays29 문제부터는 문제의 복잡도가 증가하여 3가지 알고리즘 모두 최적해를 찾지는 못하였다.

문제의 복잡도가 올라갈수록 DE1의 성능이 가장 우수하고 DE의 성능이 가장 좋지 않은 경향을 확인할 수 있다. 이러한 경향을 보이는 이유는 문제의 복잡도가 올라갈수록 지역 최적해가 다수 존재하는데 DE3의 경우 현재 발견된 가장 우수한 해 주변만 탐색하여 수렴 속도는 빠르나 지역 최적해에서 빠져나오기 어렵기 때문인 것으로 생각된다. 상대적으로 DE1은 가장 우수한 해 주변을 탐색하는 요소가 없기 때문에 지역 최적해에 빠질 가능성이 가장 낮으며 DE2는 DE1의 요소와 DE3의 요소를 혼합한 형태이기 때문에 DE1과 DE3의 가운데에 있는 것으로 판단된다.

[그림 5]부터 [그림 10]까지는 최적해를 찾지 못한 6개의 문제에 대해 3가지 차분 진화 알고리즘별로 세대별 발견한 최적의 적합도를 보여주고 있다. 300,000회 평가(집단 수: 30, 세대 수: 10,000)를 통해 dantzig42 문제까지는 수렴에 도달한 것으로 보이나 pr76문제부터는 수렴에 도달하지 못한 것으로 생각된다.

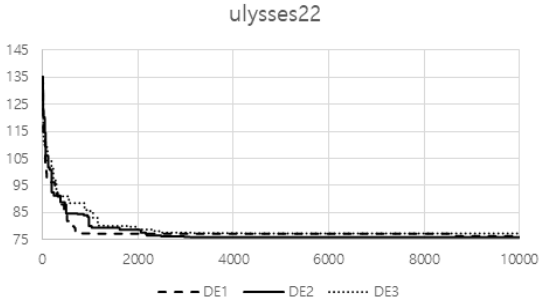

그림 5. ulyeese22 문제 실험 결과

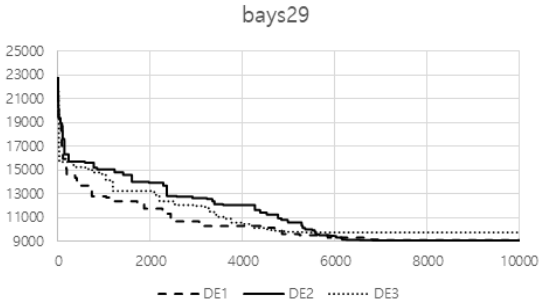

그림 6. bays29 문제 실험 결과

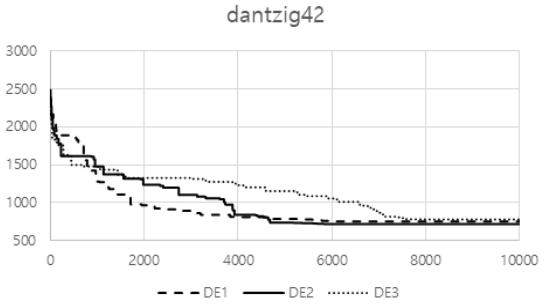

그림 7. dantzig42 문제 실험 결과

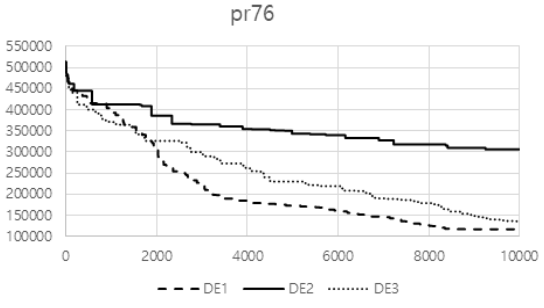

그림 8. pr76 문제 실험 결과

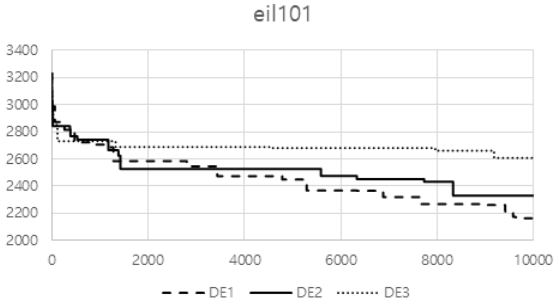

그림 9. eil101 문제 실험 결과

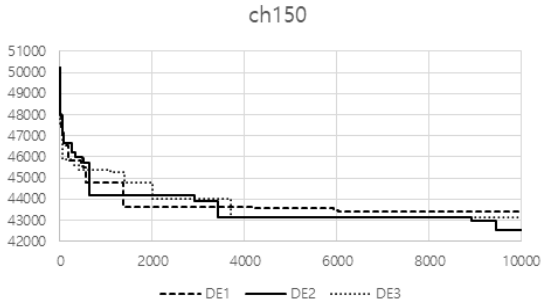

그림 10. ch150 문제 실험 결과

VI. 결 론

본 연구에서는 연속공간 탐색 기법인 차분 진화 알고리즘을 불연속 공간 문제인 외판원 문제에 적용하기 위하여 난수 키 기반 표현법을 사용하였다. 시행 벡터 생성 방법에 따라 3가지의 차분 진화 알고리즘을 제안하였으며 다양한 외판원 표준 문제에 적용하는 실험을 통해 성능을 검증해 보았다. 실험 결과 제안한 난수 키 기반 차분 진화 알고리즘이 외판원 문제 해결에 가능성을 가지고 있음을 확인하였다. 이 결과를 통해 수렴성은 좋으나 연속공간 문제 해결에 국한되었던 차분 진화 알고리즘을 복잡한 연산을 추가하지 않고도 조합 최적화 문제에 적용할 수 있는 가능성을 확인하였다. 그러나 본 연구에서는 도시 수와 관계없이 동일한 평가수인 300,000회(해집단 수: 30 × 세대 수: 10,000)를 적용하여 도시 수가 적은 문제에서는 수렴 결과를 확인하였으나 도시 수가 많아 복잡도가 높은 외판원 문제에 대해서는 수렴한 결과를 확인하지 못하였다. 향후 연구로 도시 수가 많은 외판원 문제에 대해서는 평가의 수를 늘려 얼마만큼의 성능 향상을 할 수 있는지 확인해야 할 필요가 있다.

참 고 문 헌

[1] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, “Genetic algorithms for the travelling salesman problem: a review

of representations and operators," Artificial Intelligence Review, Vol.13, No.2, pp.129-170, 1999.

[2] G. Finke, A. Claus, and E. Gunn, "A two-commodity network flow approach to the traveling salesman problem," Vol.41, pp.167-178, 1984.

[3] P. Miliotis, "Using cutting planes to solve the symmetric Travelling Salesman problem," Mathematical Programming, Vol.15, No.1, pp.177-188, 1978.

[4] S. Lin and B. W. Kernighan, "An effective heuristic 0algorithm for the traveling-salesman problem," O0perations Research, Vol.21, pp.498-516, 1973.

[5] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm intelligence, morgan kaufmann publishers,* Inc., San Francisco, CA, USA, 2001.

[6] S. Kirkpatrick and G. Toulouse, "Configuration space analysis of travelling salesman problems," Le Journal de Physique, Vol.46, No.8, pp.1277-1292, 1985.

[7] M. Dorigo and L. M. Gambardella, "Ant colony system: a coop erative learning approach to the traveling salesman problem," IEEE Transactions on Evolutionary Computation, Vol.1, No.1, pp.53-66, 1997.

[8] S. Bhide, N. John, and M. R. Kabuka, "A Boolean Neural Network Approach for the Traveling Salesman Problem," IEEE Transactions on Computers, Vol.42, No.10, pp.1271-1278, 1993.

[9] F. Glover, "Artificial intelligence, heuristic frameworks and tabu search," Managerial and Decision Economics, Vol.11, No.5, pp.365-375, 1990.

[10] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: a review of representations and operators," Artificial Intelligence Review, Vol.13, No.2, pp.129-170, 1999.

[11] A. Hussain and Y. S. Muhammad, "Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator," Computational Intelligence and Neuroscience, Vol.2017, Article ID 7430125, pp.1-7, 2017.

[12] C. Fu, L. Zhang, X. Wang, and L. Qiao, "Solving TSP problem with improved genetic algorithm," AIP Conference Proceedings 1967, 040057, 2018.

[13] S Saud, H Kodaz, and İ Babaoğlu, "Solving travelling salesman problem by using optimization algorithms," KnE Social Sciences, Vol.3, No.3, pp.17-32, 2018.

[14] I. M. Ali, D. Essam, and K. Kasmarik, "A novel design of differential evolution for solving discrete traveling salesman problems," Swarm Evolution Computation, Vol.52, Article 100607, 2020.

[15] R. Strorn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, Vol.11, No.4, pp.341-359, 1997.

[16] J. C. Bean, "Genetic algorithm and random keys for sequencing and optimization," ORSA Journal on Computing, Vol.6, No.2, pp.154-160, 1994.

[17] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

저 자 소 개

이 상 욱(Sangwok Lee) 종신회원

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

〈관심분야〉 : 휴리스틱 알고리즘, 인공지능, 최적화