▩ 연구논문

비대칭 외판원문제에서 3-Opt를 이용한 효율적인 국지탐색 알고리즘

An Efficient Local Search Algorithm for the Asymmetric Traveling Salesman Problem Using 3-Opt

김 경 구*
Kim, Kyong Goo
권 상 호*
Kwon, Sang Ho
강 맹 규*
Kang, Maing Kyu

Abstract

The traveling salesman problem is a representative NP-Complete problem. It needs lots of time to get a solution as the number of city increase. So, we need an efficient heuristic algorithm that gets good solution in a short time. Almost edges that participate in optimal path have somewhat low value cost.

This paper discusses the property of nearest neighbor and 3-opt. This paper uses nearest neighbor's property to select candidate edge. Candidate edge is a set of edge that has high probability to improve cycle path. We insert edge that is one of candidate edge into intial cycle path. As two cities are connected, It does not satisfy hamiltonian cycle's rule that every city must be visited and departed only one time.

This paper uses 3-opt's method to sustain hamiltonian cycle while inserting edge into cycle path. This paper presents a highly efficient heuristic algorithm verified by numerous experiments

1. 서론

외판원문제(traveling salesman problem)는 외판원이 본사를 출발하여 고객이 있는 모든 지점을 오직 한 번씩만 방문하고 본사로 돌아오는 최소비용의 순환로를 찾는 문제이다. 외판원문제는 1959년 아일랜드의 수학자인 Hamilton에 의해 연구가 시작된 이래 수리계획법의 대표적인 문제로서 수많은 연구가 행해져왔다.

외판원문제는 제약조건을 변경하여 다양한 형태의 문제로 바꿀 수 있다. 예를 들면, 차량경로문제, 집적회로삽입문제, PCB의 구멍 뚫기 문제[4] 및 항공기 노즐의 유도 날개 문제[8] 등 다양한 분야에서 외판원문제를 적용하고 있다. 또한 실생활에서 보면, 우편함에서 우편물을 수거하는 경우나 학교 셔틀버스의 순환경로 등에서도 찾아볼 수 있다.

그러나 외판원문제가 여러 분야에서 적용되고 있음에도 불구하고 다항식 계산량을 갖는 알

* 한양대학교 산업공학과

고리즘이 발견되지 않은 NP-Complete문제이다. 즉, 문제의 크기가 커질수록 최적해를 구하는 데 걸리는 시간이 지수적으로 증가한다. 두 교점에서 시작점에 따라 거리 또는 비용이 틀린 비대칭 외판원문제에서 조합 가능한 순환로의 개수는 (N-1)!로 교점 수가 조금만 커져도 계산량이 급격하게 증가하므로 해를 구하는 데 많은 시간이 필요하다.

외판원문제는 최적해를 찾는 최적화 알고리즘과 최적해를 보장하지는 못하지만 상대적으로 빠른 시간 내에 최적해에 가까운 해를 찾는 발견적 알고리즘이 있다. 최적화 알고리즘으로는 분지한계법과 동적계획법이 있으며 발견적 알고리즘은 경로구성 방법과 경로개선 방법으로 구분할 수 있다[4, 5].

경로구성방법은 순환로를 형성해 가는 법칙에 따라 하나의 해밀턴순환로를 만든다. 이 방법에 의한 발견적 알고리즘으로 최근거리인접점(nearest-neighbor) 알고리즘, 삽입(insertion) 알고리즘, Christofides 알고리즘, Savings 알고리즘 등이 있다[4]. 경로구성 방법은 빠른 시간에 국지해에 도달하지만 경로개선방법에 비해 해가 좋지 않다. 일반적으로 경로구성방법으로 형성한 순환로를 경로개선방법의 초기 순환로로 사용한다[7].

경로개선방법은 임의의 초기 순환로에서 개선법칙에 따라 해밀턴의 순환로를 개선해 간다. 그리고 정지조건에 따라 알고리즘을 끝낸다. 일반적으로 초기 순환로에 따라 경로개선방법에 의해 구해진 해는 달라진다. 이 방법에 의한 알고리즘은 k-opt 알고리즘, 교점삽입(node-insertion) 알고리즘, 호 삽입(edge-insertion) 알고리즘, Lin-Kernighan 알고리즘 등이 있다. Lin-Kernighan 알고리즘은 2-opt와 교점삽입 알고리즘을 이용하는 것으로 2-opt를 이용하여 해가 개선이 안될 경우 k-opt를 이용하여 국지해를 벗어난다. Kanellakis[6]는 Lin-Kernighan 알고리즘을 비대칭 외판원문제에 적용하였다.

이외에 여러 분야에 적용되는 발견적 알고리즘으로는 Guided 국지탐색 알고리즘[10], Simulated annealing 알고리즘, Tabu search 알고리즘, Genetic 알고리즘 등이 있다[3]. 이러한 알고리즘들은 전역탐색을 하는 데 사용된다.

2. 3-Opt 알고리즘

3-opt 알고리즘의 기본개념은 임의의 초기 순환로에서 3개의 호를 선택한 뒤 순환로에 포함되지 않은 다른 3개의 호를 가지고 삽입을 시도하여 값이 개선되면 순환로를 재구성하여 위의 절차를 반복하는 것이다[1]. 그림 2.1의 비대칭 외판원 문제에서 세 개의 호를 선택하여 삭제할 경우, 삽입할 수 있는 세 개의 호 조합은 교점을 v라 할 때 선택한 호가 (v_i, v_{i+1}), (v_j, v_{j+1}), (v_k, v_{k+1})이면 삽입할 수 있는 호는 (v_i, v_{j+1}), (v_j, v_{k+1}), (v_k, v_{i+1}) 한 가지이다. 교점 수가 N인 순환로에서 3개의 호를 선택할 수 있는 경우 수는 ${}_NC_3$이다. 가능한 모든 경우 수의 호를 삽입하여도 값이 개선되지 않으면 알고리즘을 끝낸다.

3-opt알고리즘에서 값을 개선하는 방법에는 두 가지가 있다. 앞에서 설명한 것처럼 값이 개선되는 새로운 이웃순환로를 찾을 때마다 순환로를 재구성하는 방법이 있고, 다른 한 가지 방법은 순환로에서 전체 이웃순환로를 모두 조사하여 그 중에서 가장 좋은 해를 갖는 순환로를 찾아 이를 초기 순환로로 대체하는 방법이다. 첫 번째 방법은 값을 개선시킬 때 걸리는 시간이 대체로 짧고 해가 빨리 개선되지 않는다. 두 번째 방법은 가능한 모든 이웃순환로를 조사하기 때문에 시간이 많이 걸리지만 해가 빨리 개선된다. 일반적으로 시간과 해를 고려하여 국지탐색을 할 때는 첫 번째 방법을 사용하며, 전역탐색에서의 부분 알고리즘으로 3-opt를 사용할 때는 두 번째 방법을 주로 사용한다.

3-opt 알고리즘을 이용한 알고리즘으로 KSH 알고리즘[2]이 있다. 이 알고리즘은 적은 비용을 가진 호들을 선정하여 이를 순환로에 삽입하여 경로개선을 하는 알고리즘이다. 적은 비용을 갖는 호들을 선정하는 방법은 비용행렬에서 행 별로 가장 적은 비용을 그 행의 모든 원소에서 삭감한 뒤 열에서도 같은 방식으로 삭감한다. 이를 비용행렬의 삭감이라 한다.

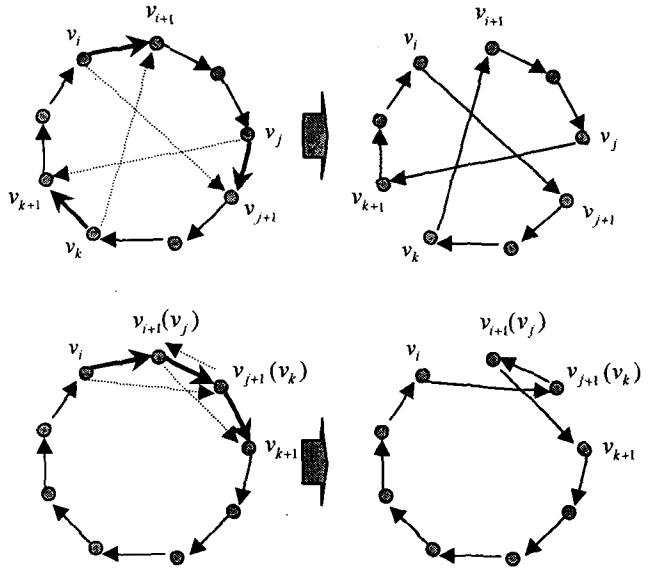

그림 2.1 3-opt 알고리즘에서 호의 삭제와 삽입

그런 다음 모든 호들을 비용이 적은 순서로 정렬하여 비용이 적은 호부터 10%를 선정한다. 선정한 호들을 가지고 임의의 해밀턴 순환로에 차례대로 삽입을 시도한다. 임의의 해밀턴 순환로에 호를 삽입을 시도해 볼 때는 3-opt에서 3개의 호를 삭제하고 3개의 호를 삽입하는 방식으로 한다. 3-opt 알고리즘은 세 개의 호를 제거한 뒤 순환로에 포함되지 않은 다른 세 개의 호를 삽입하지만 KSH 알고리즘은 선정한 호를 삽입해 보기 위해 3개의 호를 삭제하고 다른 세 개의 호를 삽입한다. 이 알고리즘은 3-opt보다 8~10배 빠른 시간 내에 3~5% 더 좋은 해를 찾아준다.

KSH 알고리즘의 절차는 다음과 같다.

단계 1. 비용행렬 삭감.

단계 2. 모든 호를 비용이 적은 순서로 정렬하여 호의 비용이 적은 순서로 10% 선정.

단계 3. 최근거리인접점 알고리즘을 사용하여 임의의 초기 해밀턴 순환로 구성.

단계 4. 정렬된 10%의 호들에 대해 차례로 해밀턴 순환로에 삽입을 시도. 이 때 삽입을 시도하여 해가 개선되면 호를 삽입하여 해밀턴 순환로를 대체, 그렇지 않으면 해밀턴 순환로를 그대로 유지하고 다음 호에 대해 삽입을 시도.

단계 5. 정렬된 10%의 모든 호들을 삽입하여도 해가 개선되지 않으면 알고리즘을 끝냄.

3. 제안하는 알고리즘

3.1 제안하는 알고리즘의 개념

제안하는 알고리즘은 선별과정을 통해 비용이 적은, 최적순환로에 포함될 가능성이 높은 호들을 선정하여 이를 초기 순환로에 삽입하는 알고리즘이다. 본 연구에서는 최근거리인접점 알고리즘을 이용한다.

이 알고리즘은 각 교점에서 아직 방문하지 않은 교점 중 가장 가까운 교점을 선택하여 경로구성을 하는 알고리즘이다. 매우 빠른 시간 내에 해를 찾아주지만 해는 좋지 않다. 그림 3.1

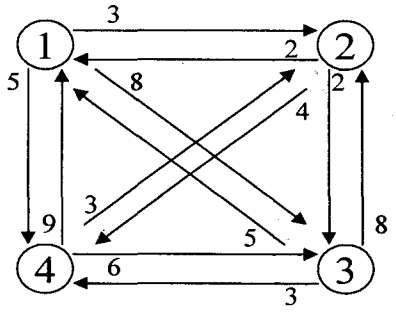

그림 3.1 네트워크 예제

에서 모든 교점을 시작교점으로 최근거리인접점 알고리즘을 적용하면 다음과 같다. 한 교점에서 다른 교점을 선택하는 것을 한 단계라 하자.

① → ② → ③ → ④ → ①
② → ① → ④ → ③ → ②
③ → ④ → ② → ① → ③
④ → ② → ① → ③ → ④

단계 : (1) (2) (3) (4)

교점 수가 N개인 외판원 문제에는 N개의 단계가 있다. 단계 (1)에서 각 교점이 가장 가까운 교점을 선택할 확률은 1이다. 단계를 거듭할수록 가까운 교점을 선택할 확률은 낮아지며, 마지막 단계에서는 해밀턴 순환로를 만족해야 하기 때문에 다음 교점은 시작교점으로 고정되어 있다. 최적순환로가 나올 경우는 단계별로 다음 교점을 선택할 때 아직 방문하지 않은 교점 중에 비용이 가장 적은 교점이 있는 경우와 모든 교점을 시작점으로 하여 최근거리인접점 알고리즘을 적용하였을 때 단계 (1)에서 선택된 호들로 해밀턴 순환로가 구성될 경우이다. 단계가 높아질수록 그때 선택된 호는 최적순환로에 포함될 가능성이 낮다.

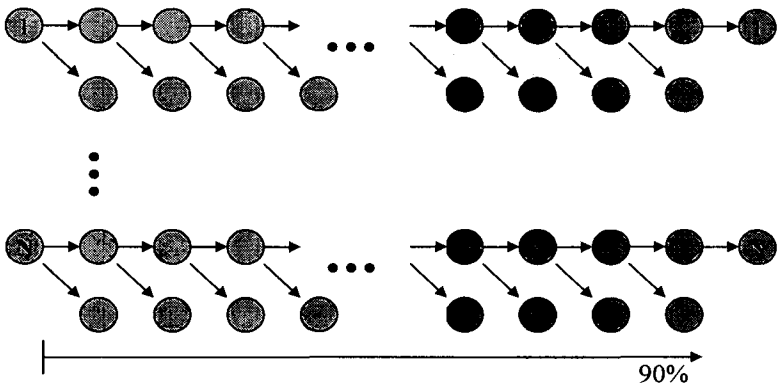

그림 3.2 후보호의 선정

그림 3.2에서 각 교점을 시작점으로 최근거리인접점 알고리즘을 적용하면 순환로가 만들어지는데 이 때 선택된 호들을 후보호(candidate edge)로 구성한다. 또한 순환로를 만들 때 선택된 교점과 방문한 교점을 제외한 교점 중, 가장 가까운 교점을 연결하는 호도 후보호에 넣어준다. 그림 3.2에서 처음의 1번 노드와 끝의 1번 노드는 이 것이 순환로임을 나타낸다.

후보호란 순환로를 개선할 확률이 높은 호들의 집합이다. 단계를 거듭할수록 순환로를 개선시킬 수 있는 호가 선택될 가능성은 낮아지며 마지막 단계에서는 좋은 호가 선택될 가능성이 매우 희박하다. 따라서 비율을 고려하여 일정한 단계까지 선택된 호만 후보호에 넣어준다.

본 연구에서는 90%의 비율이라고 가정한다. 후보호를 비용행렬의 삭감을 하였을 때의 값들로 정렬한다. 임의의 초기순환로에 후보호를 순서대로 삽입하여 본다. 호를 삽입한다는 것은 관련된 교점과 교점을 연결하는 것이다. 호가 삽입되면 모든 교점은 반드시 단 한 번 방문하고 나가야 하는 조건에 의해 삽입되는 호의 두 교점에서 두 개의 호가 제거된다. 그러면 하나의 부분 순환로와 하나의 마디로 나뉘어진다.

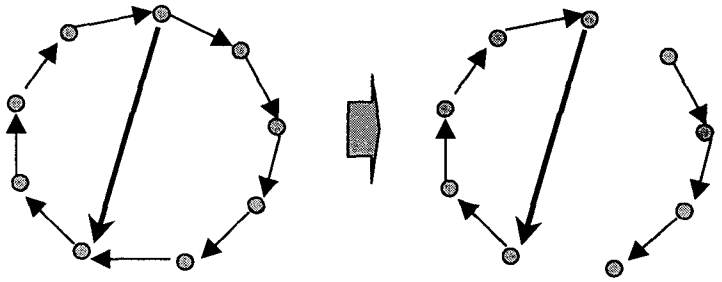

그림 3.3 호의 삽입

그림 3.3의 왼쪽 그림은 호가 삽입된 후의 순환로를 표현하고 있다. 해밀턴 순환로를 유지하기 위해 기존의 두 개의 호가 삭제되었으므로 부분 순환로에서 새로 삽입한 호를 제외한 나머지 호 중에서 한 호를 삭제하여, 3-opt 알고리즘의 방법에 따라 새로운 두개의 호를 삽입한다.

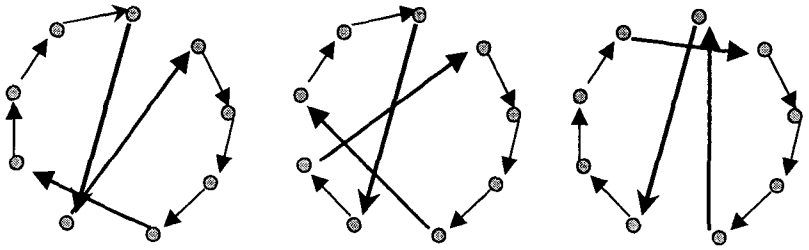

그림 3.4 이웃순환로의 예

가능한 이웃순환로의 수는 부분 순환로에서 삽입되는 호를 제외한 나머지 호의 개수와 같다. 그림 3.4는 가능한 이웃순환로의 예이다. 후보호를 삽입을 시도할 때 이루어질 수 있는 여러 이웃순환로를 조사하여 개선이 되면 기존의 순환로를 새로운 순환로로 대체하고 후보호를 처음부터 다시 삽입하여 본다. 모든 후보호를 삽입하여도 해가 개선되지 않으면 알고리즘을 끝낸다.

제안하는 알고리즘은 순환로를 개선할 좋은 호라고 판단되는 호만을 선별하여 이를 순환로에 삽입해 봄

으로써 빠른 시간 내에 국지해에 도달하는 효율적인 국지탐색 알고리즘이다.

3.2 제안하는 알고리즘의 절차

본 절에서는 제안한 알고리즘을 정식화된 절차로 표현한다. 기호의 정의는 다음과 같다.

N : 교점 수
p : 후보호에서 호의 순서 값
t : 해밀턴 순환로
t' : t의 이웃순환로
$f(t)$: 순환로 t의 비용

제안하는 알고리즘의 절차는 다음과 같다.

단계 1. [초기화]
데이터를 읽어들여 전체 교점 수(N)와 비용행렬을 초기화한다.

단계 2. [후보호의 생성]
각 교점을 시작점으로 하여 최근거리인접점 알고리즘을 적용한다. 단, 전체 단계에 걸쳐 하지 않고 (N * 0.9) 단계까지만 적용한다. 이때 선택된 호와 각 교점에서 두 번째로 가까운 교점을 연결하는 호를 후보호에 넣는다.

단계 3. [후보호의 정렬]
후보호를 비용행렬의 삭감을 통하여 얻은 비용 순서대로 정렬한다.

단계 4. [초기 순환로 구성]
최근거리인접점 알고리즘을 적용하여 초기순환로(t)를 구성한다

단계 5. [후보호의 삽입]
(5.1) p = 1
(5.2) 초기 순환로(t)에 후보 호 중 p 번째 호를 삽입을 시도해 본다.
(5.3) 생성되는 이웃한 순환로(t')를 조사한다.
(5.4) 만약 $f(t') < f(t)$이면 t를 t'로 대체하고 (5.1)로 간다.
그렇지 않으면 (5.5)로 간다.
(5.5) $p = p + 1$
(5.6) p번째 후보호가 존재하지 않으면 단계 6으로 간다.
그렇지 않으면 (5.2)로 간다.

단계 6. [알고리즘 끝냄]
알고리즘을 끝낸다.

4. 실험결과와 분석

본 장에서는 제안하는 알고리즘을 비대칭 외판원 문제에 적용하여 실험을 한다. 3장에서 설명한 알고리즘 절차대로 C++을 이용하여 프로그래밍하였다. 본 연구의 실험은 Pentium 333MHz, RAM 128M의 IBM 호환 컴퓨터에서 수행하였다. 실험에서 제안하는 알고리즘과 KSH 알고리즘을 비교 설명한다.

4.1 실험계획

알고리즘의 평가기준으로 알고리즘의 해와 수행시간을 선정한다. 제안하는 알고리즘과 KSH 알고리즘을 해와 수행시간으로 나누어 각각 비교한다. 실험에 사용하는 비대칭 외판원 문제는 난수를 발생시켜 만든 여러 교점의 랜덤한 문제와 TSPLIB[9]에 있는 문제이다. 초기해

는 각 교점별로 최근거리인접점 알고리즘을 적용하여 구한다. 그리고 교점 수만큼 만들어진 초기해에 제안하는 알고리즘과 KSH 알고리즘을 적용하여 해를 구한 다음 전체 해의 평균을 구한다.

우선 랜덤한 문제로는 교점 수가 20, 200, 400인 문제를 만들어 각 교점 수 별로 5개를 설정하여 실험한다. 다음은 TSPLIB 문제를 사용하여 제안하는 알고리즘과 KSH 알고리즘의 수행시간과 해를 비교한다. 마지막으로 후보호를 만들 때 호의 선택 비율을 변화시켜 그에 따른 해와 수행시간을 살펴본다.

4.2 실험결과

4.2.1 랜덤한 데이터에서의 비교

교점 수 20, 200, 400개인 랜덤한 데이터에서 제안하는 알고리즘과 KSH 알고리즘의 해와 수행시간을 비교한다.

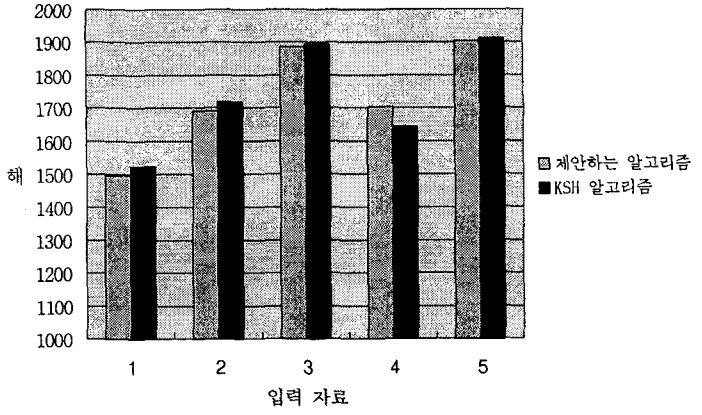

그림 4.1 교점 수 20개인 5개의 문제에 대한 해의 비교

교점 수 20인 문제에서는 수행시간이 0이었다. 그리고 그림 4.1에서 제안하는 알고리즘의 해가 더 좋은 경우가 많음을 알 수 있다. 이는 제안하는 알고리즘은 교점 수가 적은 문제에서도 후보호의 선정작업을 통하여 많은 수의 후보호를 선정하여 해를 개선시키지만 KSH 알고리즘에서는 해를 개선시키기 위해 사용하는 호의 개수가 $(N^2-N)/10$ 으로 고정되어 좋은 해를 가질 호를 충분히 포함하지 않기 때문이다.

4.2.2 TSPLIB 데이터에서의 비교

표 4.1에서 실행시간이 0인 경우를 제외하고는 제안하는 알고리즘이 KSH 알고리즘보다 시간이 약 2~8배 빠른 것을 알 수 있다. 해를 보면 교점의 수가 적은 문제에서는 대부분 제안하는 알고리즘이 KSH 알고리즘보다 좋은 해를 가진다. 그리고 문제 rbg323, rbg358를 보면 두 알고리즘의 수행시간이 차이가 많음을 알 수 있다. 이 문제는 각 교점간의 거리가 대부분 비슷한 값으로 이루어져 있다.

KSH 알고리즘을 적용할 경우, 교점간의 거리를 비용행렬의 값을 삭감하여 정렬하면 0인 값이 많이 나온다. 그러므로 값이 적은 순서대로 상위 10%의 호를 선정하여도 특정 교점에서 시작하는 호를 포함하지 않은, 좋은 해를 가질 가능성이 낮은 호들도 포함되어 해를 개선하는데 기여를 하지 못하여 시간이 많이 걸린다.

표 4.1 TSPLIB 문제에서의 비교

문제	해		시간(단위:초)	
	제안하는 알고리즘	KSH 알고리즘	제안하는 알고리즘	KSH 알고리즘
br17	39	41	0	0
ft53	7546	7569	0	0
ft70	39550	39700	0	0.01
p43	5629	5691	0	0
ftv33	1369	1388	0	0
ftv35	1535	1563	0	0
ftv38	1613	1615	0	0
ftv44	1687	1698	0	0
ftv47	1864	1866	0	0
ftv55	1717	1718	0	0
ftv64	1961	1944	0	0
ftv70	2089	2093	0	0
ftv170	3113	3083	0.08	0.19
ry48p	15274	15595	0	0
kro124p	38787	38838	0.02	0.04
rbg323	1333	1331	1.89	14.97
rbg358	1170	1168	3.56	16.74

그러나 제안하는 알고리즘은 후보호를 선정할 때 각 교점별로 최근거리인접점 알고리즘을 적용하여 이 때 선정된 호를 후보호에 넣어준다. 그러므로 모든 교점에서 값이 작은, 해를 개선시킬 가능성이 높은 호가 선정되는 것을 보장한다. 따라서 빠른 시간 내에 해를 구할 수가 있다.

4.2.3 후보호의 변화에 따른 해와 시간비교

본 연구에서 후보호를 선정할 때 호의 선택 비율을 전체 단계의 90%까지 적용하여 호를 선정하였다. 이 선택 비율을 조정하면 선정되는 후보호의 개수가 변화하여 해와 수행시간이 달라진다. 본 절에서는 호의 선택 비율을 α라 하여 α의 변화에 따른 제안하는 알고리즘의 해와 시간에 대해 비교한다. 문제는 TSPLIB에서 ft170 문제와 교점 수가 200, 400개인 랜덤한 데이터를 이용하였다.

표 4.2 TSPLIB ft170에서 α의 변화에 따른 비교

α	0.70	0.75	0.80	0.85	0.90	0.95
해	3127	3114	3113	3113	3113	3113
시간	0.064	0.070	0.070	0.070	0.076	0.076

표 4.3 교점 200개의 랜덤한 데이터에서 α의 변화에 따른 비교

α	0.70	0.75	0.80	0.85	0.90	0.95
해	2489	2489	2479	2471	2471	2471
시간	0.220	0.225	0.230	0.240	0.250	0.265

표 4.4 교점 400개의 랜덤한 데이터에서 α의 변화에 따른 비교

α	0.70	0.75	0.80	0.85	0.90	0.95
해	2901	2899	2898	2897	2896	2896
시간	2.448	2.495	2.548	2.608	2.685	2.795

표 4.2, 4.3, 4.4에서 후보호의 선택 비율이 커질수록 해는 좋아지지만 시간은 더 소요됨을 알 수 있다. 그러나 일정한 선택 비율 이상이 되면 더 이상 해가 좋아지지 않는다. 교점 170개인 TSPLIB ft170문제에서는 α가 0.80이상이면 해에 변화가 없다. 그리고 랜덤한 데이터인 교점 200개인 문제에서 α가 0.85 이상이면 교점 400개에서는 α가 0.90이상이면 해에 변화가 없음을 알 수 있다.

좋은 해를 얻기 위해서는 교점 수가 커지면 α의 값을 높게 설정할 필요가 있지만 특정 비율 이상이 되면 해에 변화가 없기에 선택비율을 조절해야 한다. 본 연구에서 선택 비율을 5% 단위로 나누어 실험하였지만 교점 수가 매우 커질 경우는 선택 비율을 1%단위로 작게 세분하여 선택할 수 있다.

5. 결론

본 연구에서 해를 개선할 가능성이 높은 호를 우선 삽입하는 방법으로 순환로를 개선한다. 호를 선정할 때는 최근거리인접점 알고리즘을 이용한다. 각 교점에 최근거리인접점 알고리즘을 적용하여 선택된 호를 호를 후보호로 선정한다. 또한 선택된 교점과 방문한 교점을 제외한 나머지 교점에서 가장 적은 비용을 가진 호도 후보호로 선정한다. 호를 선정할 때 단계가 높아질수록 좋은 호가 선택될 가능성이 낮아지므로 일정 단계까지 최근거리인접점 알고리즘을 적용한다.

순환로에 호를 삽입하여 관련된 두 교점을 연결한다. 두 교점을 연결하면 모든 교점은 반드시 한 번씩 방문하면서 나가야 하므로 순환로에 있는 다른 두 호를 삭제한다. 해밀턴순환로를 유지하기 위해 k-opt를 이용할 수 있는데 본 연구에서는 계산량을 고려하여 3-opt를 이용한다. 순환로에 후보호를 차례대로 삽입하여 해를 개선한다.

실험 결과를 통해 제안하는 알고리즘이 KSH 알고리즘에 비해 매우 빠른 시간 내에 국지탐색 해를 구함을 알 수 있다. 특히 교점의 수가 증가할수록 국지탐색을 하는 데 더욱 효율적이다. 이는 교점 수가 많음에도 불구하고 선정된 후보호는 호의 개수는 적으면서도 순환로를 개선시킬 가능성이 높은 호들로 이루어져 순환로 개선이 초기에 계속 이루어지기 때문이다.

해를 비교할 경우 제안하는 알고리즘과 KSH 알고리즘은 문제에 따라 달라진다. 교점 수가 30개 이하로 적을 경우 제안하는 알고리즘이 더 좋은 해를 구함을 알 수 있다. 이는 KSH 알고리즘은 교점 수가 적을 경우 후보호의 수가 작아 순환로를 충분히 개선시키지 않는다. 이 경우 제안하는 알고리즘의 후보호 개수가 더 많다.

시간을 비교할 경우 교점 200개인 랜덤한 데이터에서는 15~25% 정도 빠르고 교점 400개에서는 35~40% 빠르다. 이는 후보호의 개수를 고려하면 KSH 알고리즘은 $(N^2-N)/10$으로 고

정되어 있지만 제안하는 알고리즘은 후보호를 선정할 때 호가 중복되어 KSH 알고리즘에 비해 많은 호를 선정하기 않기 때문이다. 따라서 교점 수가 커질수록 시간의 차이는 커진다.

TSPLIB 문제의 경우 교점 수가 작은 문제에서 제안하는 알고리즘이 구한 해가 더 좋음을 알 수 있다. 해가 나쁜 경우에도 그리 큰 차이를 보이지 않는다. 특정 문제의 경우 제안하는 알고리즘이 KSH 알고리즘보다 최고 90% 빠름을 보여준다. 제안하는 알고리즘은 각 교점마다 최소한의 호가 선택되는 것을 보장한다. 따라서 비용이 같거나 비슷한 호가 많은 문제의 경우에서도 빠른 시간 내에 해를 구한다.

후보호의 선택 비율에 따라 해의 값과 시간이 달라진다. 비율이 커지면 해는 좋아지고 시간은 더 걸린다. 그러나 일정 비율 이상이 되면 해는 변화가 없다. 교점 수에 따라 다르겠지만 일반적으로 선택 비율을 80~90%로 설정하면 빠른 시간에 좋은 해를 구할 수 있다.

참 고 문 헌

[1] 강맹규, *네트워크와 알고리듬*, 박영사, 서울, 1991.

[2] 권상호, 강맹규, "비대칭 외판원문제에서 3-Opt를 응용한 새로운 발견적 알고리듬," *공업경영학회*, 제22권, 제52집, pp. 97-107, 1999.

[3] 김여근, 윤복식, 이상복, *메타 휴리스틱*, 영지문화사, 1997.

[4] B. Golden, L. Bodin, T. Doyle, and W. Stewart, Jr., "Approximate Traveling Salesman Algorithms," *Operations Research*, Vol. 28, pp. 694-710, 1980.

[5] Gendreau, M., A. Hertz, and G. Laporte, "New Insertion and Postoptimization Procedures for the Traveling Salesman Problem," *Operations Research*, Vol. 40, pp. 1086-1094, 1992.

[6] Kanellakis, P. C. and C. H. Papadimitriou, "Local Search for the Asymmetric Traveling Salesman Problem," *Operations Research*, Vol. 28, pp. 1086-1099, 1980.

[7] Lin, S. and B. W. Kernighan, "An Effective Heuristic Algorithm for the Traveling Salesman Problem," *Operations Research*, Vol. 21, pp. 498-516, 1973.

[8] Plante, R., T. Lowe, and R. Chandrasekaran, "The Product Matrix Traveling Salesman Problem: An Application and Solution Heuristic," *Operations Research*, Vol 35, pp. 772-783, 1987.

[9] Reinelt, G., "*The Traveling Salesman Computational Solutions for TSP Applications*," Spring-Verlag, Heidelberg, pp. 73-160, 1994.

[10] Voudouris, C. and E. Tsang, "Guided Local Search and It's Application to the Traveling Salesman Problem," *European Journal of Operational Research*, Vol. 113, pp. 469-499. 1999.