Journal of the Korean Institute of Industrial Engineers
Vol. 30, No. 2, pp. 120-129, June 2004.

외판원문제에서 국지해를 탈출하기 위한 비용완화법

권상호† · 김성민 · 강맹규

한양대학교 산업공학과

Cost Relaxation Method to Escape from a Local Optimum of the Traveling Salesman Problem

Sang-Ho Kwon · Sung-Min Kim · Maing-Kyu Kang

Department of Industrial Engineering, Hanyang University, Ansan, 425-791

This paper provides a simple but effective method, cost relaxation to escape from a local optimum of the traveling salesman problem. We would find a better solution if we repeat a local search heuristic at a different initial solution. To find a different initial solution, we use the cost relaxation method relaxing the cost of arcs. We used the Lin-Kernighan algorithm as a local search heuristic. In experimental result, we tested large instances, 30 random instances and 34 real world instances. In real-world instances, we found average 0.17% better above the optimum solution than the Concorde known as the chained Lin-Kernighan. In clustered random instances, we found average 0.9% better above the optimum solution than the Concorde.

***Keywords*:** traveling salesman, cost relaxation, Lin-Kernighan,

1. 서 론

외판원문제(TSP: traveling salesman problem)는 고전적인 최적화문제로서 외판원이 고객이 위치하고 있는 n개의 모든 지점을 오직 한 번씩만 방문하는 순환경로를 결정하는 과정에서 순환비용 또는 순환거리를 최소화하는 문제이다. 다시 말해, 고객이 위치한 교점집합 $V=\{1,2,\ldots,n\}$, 교점을 연결하는 호집합 $E=\{(i,j): i\neq j, \forall i,j\in V\}$, 호 (i,j)의 비용이 c_{ij}인 네트워크 $G(V,E)$에서 시작교점을 출발하여 n개의 모든 교점을 반드시 한 번만 방문하고 시작교점으로 돌아오는 최소비용의 해밀턴순환로(Hamiltonian cycle)를 찾는 문제이다. 외판원문제는 제약조건을 완화시키거나 변형시킴으로써 다양한 형태의 문제로 바꿀 수 있기 때문에 수리계획법의 주된 관심대상이다. 외판원문제는 문제의 단순함에도 불구하고 최초로 NP-hard로 증명된 최적화문제들 중 하나이다(Held and Karp, 1970). 따라서 최적해를 보장하는 최적화 해법보다는 최적해에 근사한 해를 빠르게 구하는 발견적 해법에 대한 많은 연구가 이루어져 왔다.

외판원문제에 대한 발견적 해법은 크게 경로구성(tour construction)방법과 경로개선(tour improvement)방법으로 구분할 수 있다. 경로구성방법은 하나의 해밀턴순환로를 구성하는 방법으로 Nearest Neighbor, Insertion, Christofides, Savings, Patching 등의 해법이 있다(Applegate *et al*., 1995; Reinelt, 1994; Lawler *et al*., 1985). 이들 중 Savings와 Patching이 좋은 해를 구하는 것으로 알려져 있다. 경로개선방법은 초기해를 구성한 상태에서 해를 개선법칙에 따라 개선하다가 정지조건에 따라 알고리즘을 종료하는 방법이다. 경로개선방법에는 Lin(1965)의 k-opt, Lin과 Kernighan(1973)의 Lin-Kernighan이 잘 알려져 있다. Chained Lin-Kernighan(Applegate *et al*., 1999)은 Lin-Kernighan을 반복 수행하는 해법으로 가장 좋은 해를 효율적으로 구하는 해법들 중 하나이다. 최근에는 Zhang(2002)과 Helsgaun(2000)이 각각 Zhang과 Helsgaun 알고리즘을 개발하였다(Johnson and McGeoch, 2002).

Lin-Kernighan은 호들을 교환하여 이웃해를 구성하는 효율적인 방법으로, 이를 반복 수행하면 더 좋은 해를 구할 수 있다. Lin-Kernighan을 반복 수행하기 위해서는 국지해에 빠진

† 연락저자 : 권상호, 425-791 경기도 안산시 사동 1271 한양대학교 산업공학과, Fax : 031-409-2423, E-mail : ksha@logistex.hanyang.ac.kr
2003년 12월 접수; 2004년 3월 수정본 접수; 2004년 4월 게재 확정.

상태에서 벗어나야 한다. 지금까지 국지해에서 벗어나기 위한 방법에는 Multi-Start, Hill-Climbing, Perturbation 방법이 있다. Multi-Start는 국지해에서 벗어나기 위해 국지해와는 다른 임의의 초기해를 구성하여 해법을 반복 수행하는 방법이다. 이 방법은 쉽고 잘 알려진 방법이지만 매 반복마다 새로운 초기해에서 국지해를 찾는 데 많은 이웃해를 거쳐야 하기 때문에 효율적이지 못하다. Hill-climbing 방법에는 Tabu(Applegate *et al*., 1995), Simulated Annealing(Martin and Otto, 1996) 등이 있다. 이 방법은 국지해에서 벗어나기 위해 전체 순환비용이 증가하더라도 어떤 특정한 조건을 만족시키면 그 때의 해를 이웃해로 선택하는 방법이다. 이 방법을 사용하면 국지해에서 벗어날 수 있지만 여러 관련 파라미터를 설정해야 하는 복잡성과 어떤 특정 조건을 만족시키기 위해 많은 이웃해를 거쳐야 하는 단점이 있다.

본 논문에서는 지금까지 국지해에서 벗어나기 위한 방법들의 단점을 극복하는 새로운 Perturbation 방법인 비용완화법을 제시한다. 이 방법은 국지해에서 벗어나기 위해 한 번에 많은 이웃해를 효과적으로 건너뛰어 국지해에서 일정한 거리에 떨어진 해를 이웃해로 선택한다. 이 방법은 국지해에서 벗어나기 위해 몇 개의 교점을 기준으로 이 교점들과 연결된 모든 호들의 비용을 완화하는 방법이다. 이렇게 비용을 완화하면 Lin-Kernighan이 탐색하지 못한 이웃해를 탐색하여 효율적으로 국지해를 탈출한다. 기존 Chained Lin-Kernighan 방법은 국지해를 벗어나기 위해 Double-Bridge 방법을 사용했다. 이 방법은 Perturbation 방법의 한 종류로서 Lin-Kernighan 방법이 탐색하지 못하는 4개 호의 교환을 수행하는 방법이다. 본 논문에서는 많은 실험을 통해 Double-Bridge를 사용하는 Chained Lin-Kernighan의 해와 본 논문에서 제시하는 비용완화법을 사용한 해를 비교하고 이의 우수성을 보인다.

2. 국지탐색 해법의 반복방법

국지탐색 해법을 반복 수행하면 한 번 수행한 것보다 더 좋은 해를 구할 수 있다. 국지탐색 해법을 반복 수행하면 한 번만 수행하는 것보다 수행시간이 많이 소요되지만 효율적인 국지탐색 해법을 반복 수행하는 시간은 최적해법을 수행하는 시간에 비하면 극히 적은 시간이다. 국지탐색 해법을 반복하는 방법은 크게 Multi-Start, Hill-Climbing, 그리고 Perturbation으로 구분할 수 있다.

Multi-Start 방법은 국지탐색 해법을 반복하는 가장 간단한 방법이다. 이 방법은 매 반복마다 임의의 새로운 초기해를 구성한 후 국지탐색 해법을 적용하여 결국 전체 반복에서 발견된 해들 중 가장 좋은 해를 취하는 방법이다. 만약 문제가 작은 경우라면 이 방법을 사용하여 쉽게 최적해를 구하기도 한다.

<Figure 1>은 Multi-Start 방법의 개념을 보이고 있다. <Figure1>에서 T는 해밀턴순환로이고 f는 해밀턴순환로의 비용이다. <Figure 1>에서 trial 1, 2, 3은 임의로 생성한 세 개의 초기해에 대해 국지탐색 해법을 적용하여 세 개의 국지해를 발견하는 모습이다.

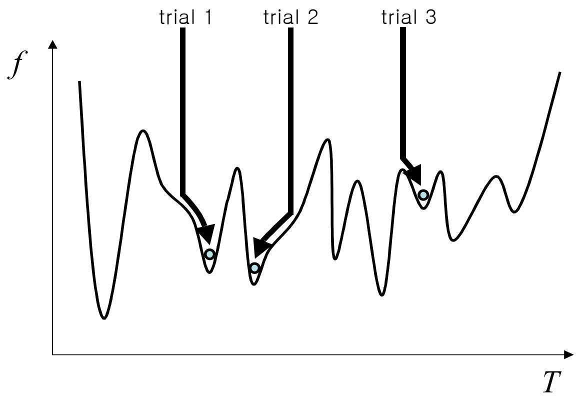

Figure 1. Multi-Start.

Multi-Start 방법은 사용하기 쉬운 반면 그 단점도 크다. 이 방법은 매 반복마다 임의의 새로운 초기해, 즉 바로 이전 국지해와는 전혀 관련이 없는 초기해에서 시작하기 때문에 매 반복에서 거쳐야 하는 이웃해가 많아 비효율적이다. 또한 이 방법은 바로 이전 국지해와는 전혀 다른 멀리 떨어진 새로운 초기해에서 시작하기 때문에 바로 이전 국지해가 좋을 경우, 이를 개선하기는 쉽지 않다.

Multi-Start 방법이 국지탐색 해법을 반복할 때 매번 새로운 임의의 초기해에서 시작하는 반면, Hill-Climbing 방법은 바로 이전 국지해에서 탈출하여 국지탐색 해법을 반복한다. 국지해에서 탈출하는 방법에는 Tabu와 Simulated Annealing 등이 있다. 이 방법들은 국지해에서 탈출하기 위해 국지해보다 해가 나빠져도 어떤 조건에 의해 해를 이동시킨다. 예를 들어 2-opt를 국지탐색 해법으로 사용하는 Tabu 방법은 국지해에 빠지면 2-opt로 구해지는 국지해의 이웃해 들 중 해가 가장 적게 증가하는 이웃해를 선택한다. Simulated Annealing 방법은 온도로 표현되는 에너지 상수를 사용하여 에너지량에 따라 탐색하는 이웃해가 국지해보다 증가해도 이를 선택한다. <Figure 2>는 Hill-Climbing 방법인 Tabu로 국지해를 탈출하는 모습이다.

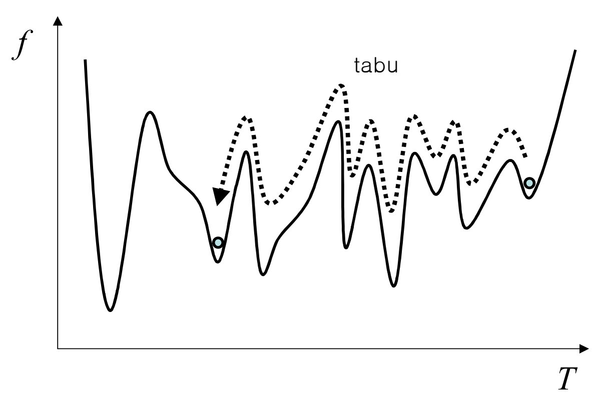

Figure 2. Escaping from a local optimum by tabu method.

Hill-Climbing 방법은 Multi-Start 방법과는 달리 국지해를 탈출하는 방법이지만 국지해를 벗어나기 위한 조건을 만족시켜야 하며 이 과정에서 많은 이웃해를 거쳐야 한다. 또한 국지해를 벗어나기 위한 조건을 두기 위해 파라미터를 임의적으로 조절해야 하는 문제가 있다. 예를 들어 Simulated Annealing 방법은 에너지 상수에 의해 순환비용이 더 큰 이웃해를 취할지를 결정한다.

Perturbation 방법은 Hill-Climbing 방법과 같이 국지해를 탈출하는 방법이다. 그러나 이 방법은 Hill-Climbing 방법과는 달리 한 번에 이웃해를 효과적으로 건너뛰어 국지해를 탈출하는 장점이 있다. Perturbation 방법은 Martin *et al*.(1992)이 제안한 개념으로 이전 반복에서 발견한 국지해를 다음 반복의 초기해로 변환하는 방법이라고도 할 수 있다. 이 방법을 intermediate (Codenotti *et al*., 1996) 또는 kick이라고 부르기도 한다.

Perturbation 방법으로 생성한 초기해는 임의의 새로운 초기해보다 좋을 확률이 높다. <Figure 3>은 교점수가 100인 외판원문제에서 임의의 초기해 6138개를 히스토그램으로 나타낸 것이다. 임의의 새로운 초기해는 히스토그램에서 발생빈도가 가장 높은 중앙부분에 위치할 확률이 높다. 그러나 Perturbation 방법의 초기해는 이전 반복의 국지해를 약간 변환하여 생성하기 때문에 매 반복에서 좋은 초기해를 생성할 수 있다(Martin *et al*., 1992).

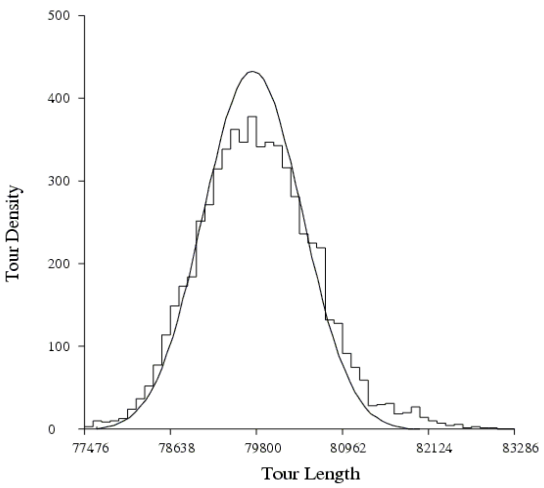

Figure 3. A distribution of random initial 6138 tours from an instance($n = 100$).

Perturbation 방법의 효과는 그 수행 방법에 따라 달라진다. <Figure 4>는 세 가지의 수행방법을 보이고 있다. <Figure 4>에서 a는 이전 반복에서 발견한 국지해를 이용하여 멀리 떨어진 다음 반복의 초기해를 생성하는 경우이다. 이 경우 Perturbation 방법으로 생성한 초기해는 멀리 떨어질수록 임의로 생성한 초기해와 큰 차이가 없어 Multi-Start 방법과 비슷한 결과를 초래한다. b는 a의 반대인 경우로, 다음 반복의 초기해를 이전 반복의 국지해에 가깝게 생성하는 경우이다. 이 경우 이전 반복의 국지해를 다시 탐색할 가능성이 높기 때문에 반복의 효과가 줄어든다. c는 이전 반복의 국지해에서 적당하게 떨어진 초기해를 생성하는 경우로 반복의 효과를 가장 높일 수 있는 경우이다.

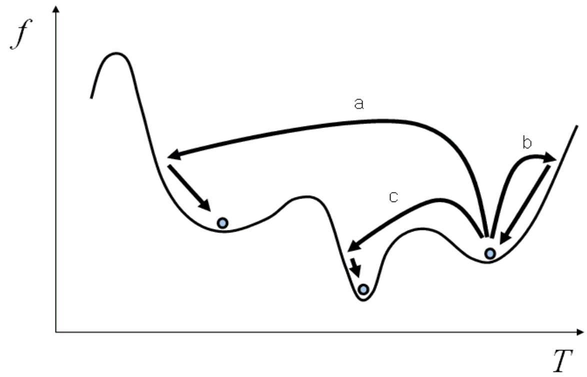

Figure 4. Perturbation method.

Perturbation 방법에는 4개 호를 교환하는 Double-Bridge 방법(Applegate *et al*., 1999; Martin *et al*., 1992)이 있다. 이 방법은 국지해에 빠지면 이를 탈출하기 위해 4개의 호를 교환한다. Double-Bridge를 사용한 대표적인 해법은 Chained Lin-Kernighan이다. 이 해법은 Lin-Kernighan 해법을 수행하여 국지해에 도달하면 Double-Bridge 방법으로 4개의 호를 교환하여 국지해를 탈출한다.

Lin-Kernighan 해법(Lin and Kernighan, 1973)은 효율적으로 좋은 해를 구하는 국지탐색 해법으로 잘 알려져 있어, 이를 응용한 해법이 많이 연구되었다(Helsgaun, 2000). 이 해법은 k-opt 해법에서 k를 가변적으로 증가시켜 가며 호들을 교환한다. k-opt 해법이 k를 고정시켜 k개의 호만을 교환하여 이웃해를 탐색하는 반면 Lin-Kernighan 해법은 교환하는 호의 개수인 k를 2, 3, 4, …으로 순차적으로 증가시켜 가며 이웃해를 탐색한다. 이 해법은 k를 증가시키다가 전체 해가 감소되는 시점에서 그 동안 가장 좋은 해를 구하는 k개의 호를 교환한다.

Lin-Kernighan 해법은 4개의 호를 교환할 수 있지만 이는 순차적인 교환이다. 반면 Double-Bridge 방법은 비순차적으로 4개의 호를 교환하는 방법이다. 따라서 Lin-Kernighan 해법으로 찾을 수 없는 이웃해를 Double-Bridge 방법을 사용하면 찾을 수 있다. <Figure 5>는 Double-Bridge 방법으로 4개 호를 교환하는 모습을 보인 것이다.

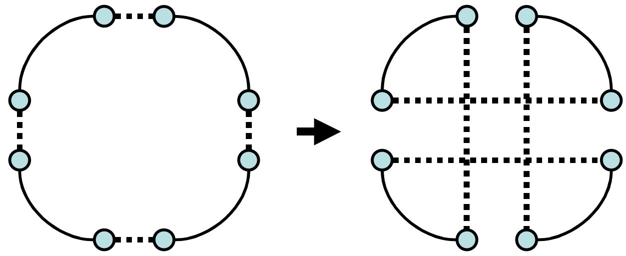

Figure 5. Exchange of four arcs by double-bridge.

3. 비용완화법

본 연구에서 제시하는 비용완화법은 국지해를 탈출하는 새로운 방법이다. 비용완화법은 임의의 교점에 연결된 모든 호의 비용을 0으로 완화하는 방법이다. 현재의 해밀턴순환로를 T, 변경된 해밀턴순환로를 T', 원문제의 비용행렬을 C, 변경된 비용행렬을 C', 호 (i, j)에 대한 C'의 원소를 c'_{ij}라고 하면 비용을 완화하고 국지해를 탈출하는 절차는 다음과 같다.

Procedure **Cost-Relaxation**(T, C)
  $C' \leftarrow C$
  Select randomly a distinct node i
  $c'_{ij} \leftarrow 0$, $c'_{ji} \leftarrow 0$ for all j
  Call Local-Search(T, C') to produce a tour T'
Return T'

비용완화 절차에서 Local-Search(T, C')는 임의의 국지탐색 해법으로 이를 호출한 이유는 비용행렬을 완화한 다음 국지해를 탈출하기 위해서이다. 본 논문에서는 가장 좋은 국지탐색 해법으로 알려진 Lin-Kernighan 해법을 사용한다.

비용완화법은 비용이 큰 호가 장벽이 되는 것을 막아 국지해에서 탈출한다. Lin-Kernighan 해법은 순환비용이 큰 초기해에서 k개의 호를 한 묶음으로 교환하여 한 번에 많은 비용을 감소시킨다. 그러나 최적해에 가까이 갈수록 더 좋은 해를 찾기는 어렵다. 만일 k가 2이고 교환하려는 현재의 해에 포함된 두 개의 호의 비용은 각각 5, 6이고 현재의 해에 포함되지 않은 두 개의 호의 비용은 각각 2, 10이라고 하자. 그러면 비용이 2인 호는 비용이 적음에도 불구하고 비용이 10인 호에 의해 전체 호의 비용이 12가 되어 호들을 교환하지 못한다. 이때 만약 비용이 10인 호를 0으로 완화하면 호들을 교환할 수 있으며 비용이 2인 좋은 호를 해밀턴순환로에 포함시킬 수 있다. 따라서 비용완화법은 비용이 큰 호의 비용을 낮추어 상대적으로 비용이 큰 호의 역할을 크게 한다.

비용완화법은 하나 이상의 교점에서 호들을 완화할 수 있다 완화하는 교점의 개수를 m이라고 하자. 만약 m이 적으면 국지해에서 가깝게 탈출하고 m이 크면 국지해에서 멀리 탈출한다. m을 포함하여 비용행렬을 완화하고 국지해를 탈출하는 절차는 다음과 같다.

Procedure **Cost-Relaxation-1**(T, C, m)
  $C' \leftarrow C$
  Do m times
    Select randomly a distinct node i
    $c'_{ij} \leftarrow 0$, $c'_{ji} \leftarrow 0$ for all j
  End
  Call Local-Search(T, C') to produce a tour T'
Return T'

비용완화법은 새로운 종류의 Perturbation 방법이다. 이 방법은 비용행렬을 완화하고 국지탐색 해법을 수행한 후 다시 비용행렬을 원상태로 복구하여 국지해를 탈출한다. 비용행렬을 완화하는 방법에 따라 국지해에서 탈출한 해는 국지해에 가깝게 탈출하거나 멀리 탈출하게 할 수 있다.

<Figure 6>은 비용완화법을 사용하여 국지해를 탈출한 후 새로운 국지해를 찾는 개념도이다. <Figure 6>에서 a는 이전 국지해이고 비용을 완화하면 b로 해가 이동한다. 국지탐색 해법을 수행하면 b에서 다시 해가 c로 이동하는데, 이 때 원문제로 문제를 복구하면 해는 c에서 d로 이동한다. 다시 국지탐색 해법을 수행하면 해가 d에서 e로 이동한다. 이렇게 비용완화법을 사용하면 이전 국지해를 탈출하여 일정한 거리의 이웃해를 초기해로 하여 더 좋은 국지해를 찾을 수 있다.

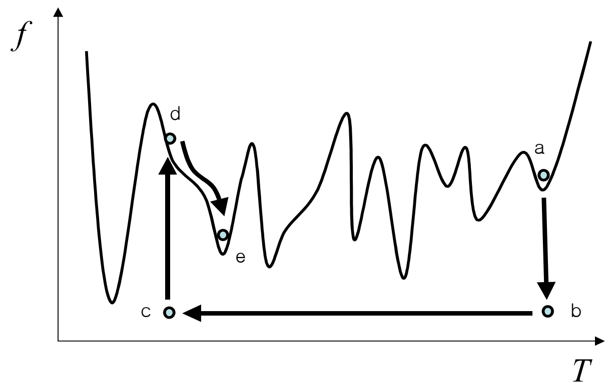

Figure 6. Escaping from a local optimum by the cost-relaxation method.

본 논문에서 제시하는 해법이 Chained Lin-Kernighan 해법과 다른 부분은 국지해를 탈출하는 방법이다. 국지해를 탈출하는 방법으로 Chained Lin-Kernighan은 Double-Bridge 방법을 사용하는 반면 제시하는 해법은 비용완화법을 사용한다 또한 제시하는 해법은 한 번의 반복에서 두 번 Lin-Kernighan 해법을 수행한다. 제시하는 해법은 국지해를 탈출(<Figure 6>에서 b에서 c로 이동)하기 위해 Lin-Kernighan 해법을 수행하고, 원문제로 복구한 후 해를 개선(<Figure 6>에서 d에서 e로 이동)하기 위해 Lin-Kernighan 해법을 수행한다. 다음은 전체 해법의 절차이다.

Procedure **Cost-Relaxation-Lin-Kernighan**(T)
  Call Lin-Kernighan(T) to produce a local optimum T'
  Do n times
    Call Cost-Relaxation-1(T', C, m) to produce a tour T''
    $C' \leftarrow C$
    Call Lin-Kernighan(T'') to produce a local optimu T'''
     If $T''' < T'$ then $T' \leftarrow T'''$
  End
Return

해법 절차에서 $C' \leftarrow C$ 부분은 비용행렬을 원상태로 복구하는 부분이다.

네트워크 $G(V,E)$에서 $S \subset V$인 집합 S에 대해 각 호 (u,v)에서 두 교점 중 하나의 교점인 u가 집합 S에 포함되는 호의 집합을 $\delta(S)$라 하고, 두 교점 u, v가 모두 집합 S에 포함되는 호의 집합을 $\gamma(S)$라 하자. 그러면 집합 $\delta(S)$와 $\gamma(S)$를 각각 다음 식 (1), (2)와 같이 정의할 수 있다.

$$\delta(S)=\{(u,v): u \in S, v \notin S\} \tag{1}$$

$$\gamma(S)=\{(u,v): u \in S, v \in S\} \tag{2}$$

완화하려는 교점의 집합을 S라고 하면 $|\delta(S)|$와 $|\gamma(S)|$는 식 (1)과 식 (2)에 따라 각각 다음 식 (3), (4)와 같이 표현할 수 있다.

$$|\delta(S)| = m \cdot (n-m) \tag{3}$$

$$|\gamma(S)| = m \cdot (m-1) \tag{4}$$

비용완화법에서 비용을 0으로 완화하는 전체 호의 집합을 E_0라고 하자. 그러면 $|E_0|$은 비용을 0로 완화하는 전체 호의 개수가 되어 Perturbation의 정도를 나타낸다. 다시 말해 $|E_0|$은 국지해에서 탈출할 때 국지해에서 얼마나 멀리 탈출할지를 결정하는 척도이다. $|E_0|$은 식 (3)의 $|\delta(S)|$과 식 (4)의 $|\gamma(S)|$을 합한 값이므로 다음 식 (5)와 같다.

$$|E_0| = m \cdot (n-1) \tag{5}$$

본 연구에서는 교점의 수가 증가함에 따라 해법의 효율이 저하되는 것을 방지하기 위해 m을 일정한 상수 p로 고정한다.

4. 실험결과

교점수가 적은 문제들에 대해서는 최근의 발견적 해법만으로도 최적해를 찾기 때문에 본 논문에서는 교점수가 1,000개 이상인 기존에 연구된 문제들을 실험한다. 교점수가 1,000개 이상인 문제들을 세 종류의 문제군으로 나누어 실험한다. 세 종류의 문제군은 다음과 같다.

1. 교점들이 균등(uniform)하게 분포하는 문제
2. 교점들이 군집(cluster)을 이루며 분포하는 문제
3. TSPLIB(Reinelt, 1991)에 등록된 문제

문제 1군의 문제들은 일반적인 외판원문제이며, 문제 2군의 문제들은 교점들이 군집을 이루는 문제들로서 문제 1군의 문제들보다 풀기 어렵다. 문제 3군의 문제들은 균등하지도 군집을 이루지도 않은 혼합된 문제들이다.

실험결과로는 모든 문제에 대해 10번 반복 실험하여 발견한 해의 평균값과 평균 수행시간을 이용한다. 제시하는 해법의 종료조건으로는 반복수를 사용하며 교점수 n을 반복수로 사용한다. 종료조건으로 반복수를 사용하는 이유는 본 논문의 요지가 국지해를 탈출하는 방법이기 때문에 같은 반복에서 해의 질을 평가하기 위해서이다. 만약 일정한 시간으로 해법을 평가하게 되면 해법에 따라 반복수가 다르게 나타나기 때문에 같은 반복에서 해의 질을 평가하기는 어렵다. 실험은 Intel Pentium III 1GHz의 CPU와 메모리 512MB를 가진 IBM 호환 PC 하드웨어와 Linux 2.4.7의 소프트웨어 환경에서 수행한다. 제안하는 해법을 비교하기 위해 Applegate *et al.*(1999)이 개발한 Concorde를 사용한다. 이는 Chained Lin-Kernighan 해법을 구현한 소프트웨어 중에서 가장 좋은 성능을 나타내는 것으로 알려져 있다.

실험에 앞서 완화되는 교점수 m의 최적값을 결정하기 위해

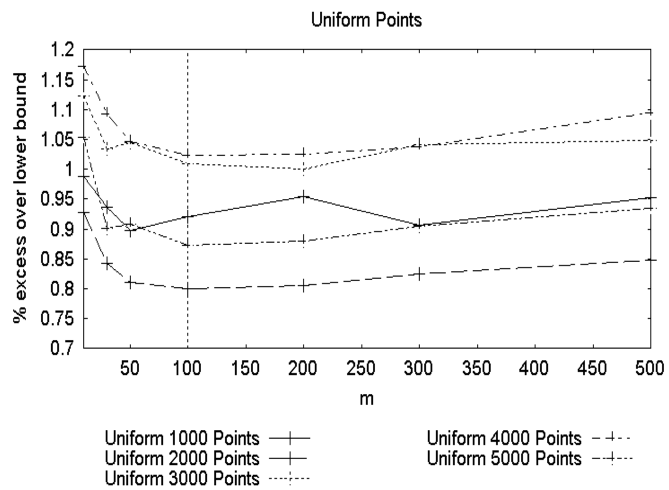

Figure 7. Excess over opt. as change of m in the uniformly distributed instances.

문제를 두 종류로 나누어 실험하였다. 해는 최적해에서 벗어난 정도, 즉 ((해-최적해)/최적해)를 %로 나타냈다. 교점들이 균등하게 분포하는 문제들을 실험한 <Table 1>과 <Figure 7>에서 문제크기에 관계없이 $m=100$인 경우, 가장 좋은 해를 발견하였다. 또한 <Table 2>와 <Figure 8>에서도 문제크기에 관계없이 $m=100$인 경우 가장 좋은 해를 발견하였다.

$m=100$을 사용하여 기존 연구된 문제들 중 문제 1군에 해당하는 15개 문제와 문제 2군에 해당하는 15개 문제, 문제 3군의 34개 문제를 대상으로 실험하였다. 초기해는 Quick Boruvka 해법(Applegate *et al*., 1995)을 사용하였다. <Table 3, 4, 5>에서 excess(%)는 최적해에서 벗어난 정도를 %로 표시한 것이고, difference는 각 해법의 excess(%)의 차이다. <Table 3>은 문제 1군의 문제들을 실험한 것으로, 제시한 해법이 Concorde보다 평균 0.128% 더 좋은 해를 발견하였다. 그러나 수행시간은 약 4~6배 더 소요되었다. <Figure 9>는 문제 1군의 문제들 중 교점이 3000개인 E3k.1문제에 대해 제시한 해법과 Concorde와의 수렴속도를 비교한 것이다. 제시한 해법이 더 빠르게 수렴하는 것으로 나타났다.

Table 1. Excess over opt. as change of m in the uniformly distributed instances

n	m						
	10	30	50	100	200	300	500
1000	0.988	0.936	0.897	0.921	0.955	0.907	0.952
2000	0.928	0.842	0.811	0.800	0.806	0.824	0.847
3000	1.123	1.034	1.045	1.009	1.001	1.042	1.048
4000	1.172	1.093	1.046	1.023	1.025	1.037	1.095
5000	1.053	0.901	0.909	0.873	0.880	0.904	0.935
average	1.053	0.961	0.942	0.925	0.933	0.943	0.975

Table 2. Excess over opt. as change of m in the clustered instances

n	m						
	10	30	50	100	200	300	500
1000	3.932	3.859	3.620	3.248	3.246	3.256	3.261
2000	1.774	1.564	1.561	1.450	1.427	1.389	1.353
3000	2.242	1.822	1.744	1.633	1.657	1.662	1.685
4000	2.372	1.964	1.927	1.790	1.806	1.838	1.866
5000	2.655	2.082	1.847	1.946	1.971	2.074	1.943
average	2.475	2.258	2.140	2.013	2.021	2.044	2.022

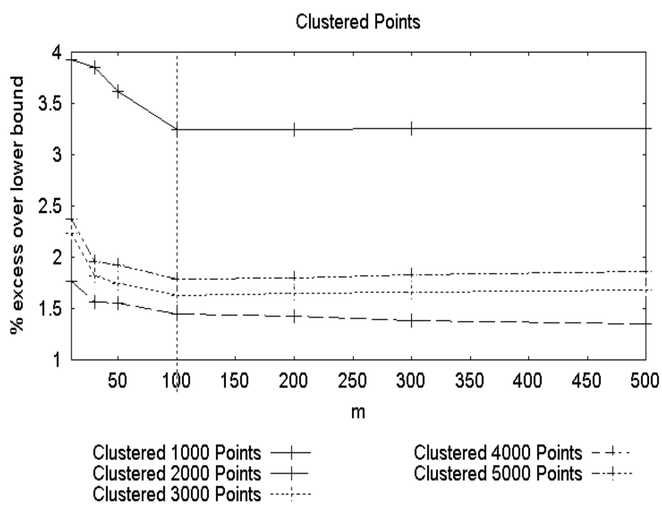

Figure 8. Excess over opt. as change of m in the clustered instances.

Table 3. Results of instance family 1

Instance	Optimum	Concorde		Proposed		difference
		excess(%)	run time(sec.)	excess(%)	run time(sec.)	
E1k.0	23360648	0.326	2.21	0.188	12.36	0.138
E1k.1	22985695	0.266	2.34	0.088	11.65	0.179
E1k.2	23023351	0.462	2.50	0.205	13.18	0.257
E1k.3	23143748	0.158	1.92	0.147	11.19	0.011
E1k.4	22698717	0.304	2.08	0.135	11.96	0.169
E1k.5	23192391	0.310	2.25	0.110	12.16	0.200
E1k.6	23349803	0.232	2.14	0.143	12.55	0.089
E1k.7	22879091	0.285	2.11	0.134	11.41	0.152
E1k.8	23025754	0.184	2.02	0.110	11.31	0.074
E1k.9	23356256	0.198	2.03	0.083	10.72	0.115
E3k.0	40634081	0.202	9.78	0.148	46.54	0.054
E3k.1	40315287	0.306	9.56	0.181	43.07	0.126
E3k.2	40303394	0.246	10.02	0.121	46.20	0.125
E3k.3	40589659	0.311	10.46	0.193	48.85	0.118
E3k.4	40757209	0.242	9.60	0.126	42.31	0.116
average		0.268	4.73	0.141	23.03	0.128

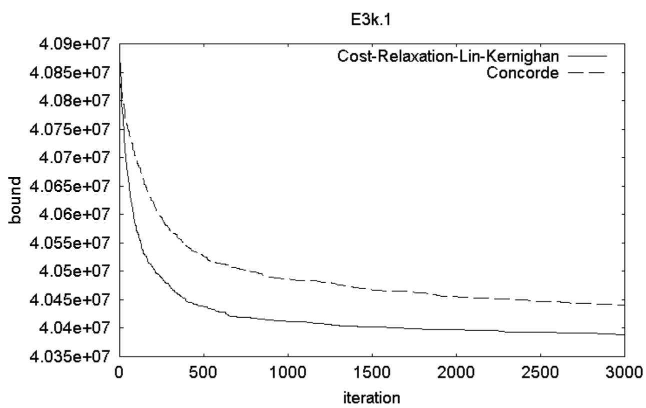

Figure 9. Convergency of solutions of the E3k.1.

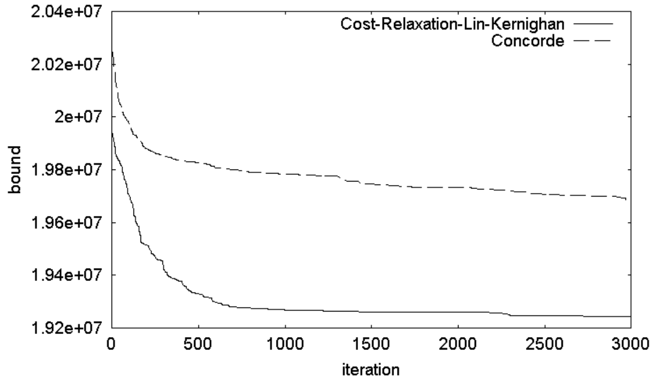

Figure 10. Convergency of solutions of the C3k.0.

<Table 4>는 문제 2군의 문제들을 실험한 것으로 제시한 해법이 Concorde보다 평균 0.900% 더 좋은 해를 발견하였다. 그러나 수행시간은 약 9～15배 더 소요되었다. <Figure 10>은 문제 2군의 문제들 중 교점이 3000개인 C3k.0 문제에 대해 제시한 해법과 Concorde와의 수렴속도를 비교한 것이다. 제시한 해법이 더 빠르게 수렴하는 것으로 나타났다.

문제 3군의 TSPLIB 34개 문제에서는 29개 문제에서 제시한 해법이 Concorde보다 더 좋은 해를 구했다. <Table 5>는 문제 3군의 문제들을 실험한 것으로, 제시한 해법이 Concorde보다 평균 0.170% 더 좋은 해를 발견하였다. 그러나 수행시간은 약 3～10배 더 소요되었다.

<Figure 11>은 문제의 유형에 따라 제시한 해법과 Concorde의 해를 비교한 것으로, <Table 3, 4, 5>에서 차이 항목의 값을 문제의 크기별로 나타낸 것이다. 점들이 0 위에 위치하면 제시한 해법이 Concorde보다 더 좋은 해를 발견한 경우이다. 제시한 해법은 Concorde보다 대부분의 문제에서 더 좋은 해를 구했으며 특히 문제 2군, 군집을 이루며 분포하는 문제에서는 월등히 좋은 해를 발견하였다.

<Figure 12>는 제시한 해법과 Concorde의 수행시간을 비교한 것이다. 제시한 해법이 Concorde보다 약 3～15배 수행시간이 더 걸리는 것으로 나타났다. 그러나 문제가 커짐에 따라 시간 차이가 작아지는 것을 확인했다.

Table 4. Results of instance family 2

Instance	Optimum	Concorde		Proposed		Difference
		excess(%)	run time(sec.)	excess(%)	run time(sec.)	
C1k.0	11387430	0.500	9.20	0.343	131.10	0.157
C1k.1	11376735	0.091	7.55	0.030	92.51	0.061
C1k.2	10855033	1.736	10.71	0.950	162.70	0.786
C1k.3	11886457	0.025	9.97	0.026	109.10	-0.001
C1k.4	11499958	0.058	9.48	0.051	123.01	0.007
C1k.5	11394911	0.263	9.18	0.059	131.29	0.205
C1k.6	10166701	1.818	8.01	0.134	102.56	1.684
C1k.7	10664660	1.297	9.25	0.079	116.22	1.218
C1k.8	11605723	2.385	13.64	0.093	145.84	2.293
C1k.9	10906997	0.743	11.83	0.166	122.88	0.577
C3k.0	19198258	2.416	42.44	0.221	409.23	2.195
C3k.1	19017805	1.721	39.94	1.154	485.53	0.567
C3k.2	19547551	1.685	42.51	0.942	461.90	0.743
C3k.3	19108508	2.281	41.26	1.080	419.03	1.201
C3k.4	18864046	2.365	39.41	0.554	382.56	1.812
average		1.292	20.29	0.392	226.36	0.900

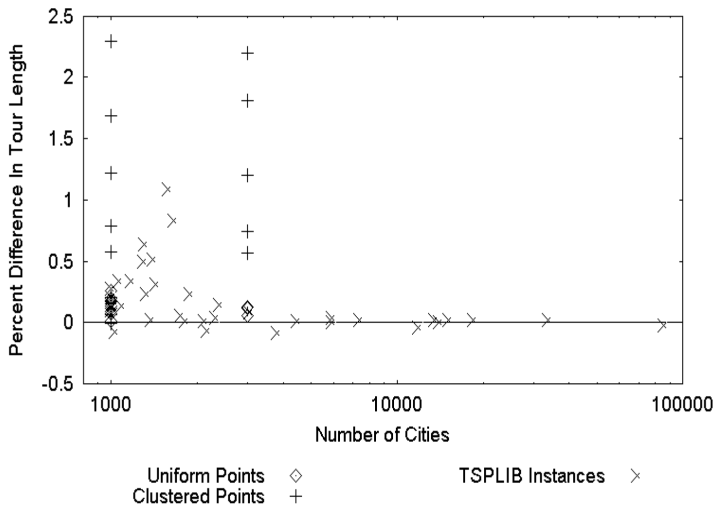

Figure 11. Difference of solutions between Concorde and proposed method.

Table 5. Results of instance family 3

Instance	Optimum	Concorde		Proposed		Difference
		excess(%)	run time(sec.)	excess(%)	run time(sec.)	
dsj1000	18659688	0.360	5.55	0.257	56.70	0.103
pr1002	259045	0.388	2.21	0.112	12.17	0.276
si1032	92650	0.094	2.86	0.170	23.94	-0.077
u1060	224094	0.416	3.58	0.074	23.41	0.341
vm1084	239297	0.194	3.21	0.059	17.89	0.136
pcb1173	56892	0.420	1.77	0.083	10.25	0.338
d1291	50801	1.028	4.07	0.530	25.31	0.497
rl1304	252948	0.901	4.38	0.260	25.63	0.642
rl1323	270199	0.660	3.75	0.430	28.25	0.230
nrw1379	56638	0.122	2.30	0.101	13.66	0.021
fl1400	20127	0.715	17.73	0.202	183.74	0.514
u1432	152970	0.515	3.74	0.205	19.52	0.310
fl1577	22249	3.893	13.24	2.807	132.82	1.086
d1655	62128	1.151	5.46	0.319	40.90	0.832
vm1748	336556	0.156	5.65	0.099	35.77	0.057
u1817	57201	0.827	3.51	0.818	17.45	0.009
rl1889	316536	0.593	7.21	0.358	49.96	0.235
d2103	80450	0.229	6.80	0.217	63.54	0.011
u2152	64253	0.638	4.67	0.705	19.86	-0.067
u2319	234256	0.161	18.56	0.123	78.56	0.039
pr2392	378032	0.576	5.24	0.432	24.71	0.144
pcb3038	137694	0.336	6.31	0.253	27.91	0.083
fl3795	28772	1.233	30.58	1.313	284.33	-0.080
fnl4461	182566	0.167	11.35	0.151	49.45	0.016
rl5915	565530	0.676	25.07	0.636	136.32	0.040
rl5934	556045	0.458	24.57	0.452	148.80	0.006
pla7397	23260728	0.421	46.20	0.398	207.18	0.023
rl11849	923288	0.373	62.83	0.414	237.70	-0.041
usa13509	19982859	0.261	93.18	0.241	314.20	0.020
brd14051	467128	0.670	62.58	0.663	226.85	0.007
d15112	1573084	0.195	80.72	0.172	288.35	0.023
d18512	642117	0.687	79.48	0.670	274.05	0.018
pla33810	65705438	0.856	236.18	0.830	833.96	0.026
pla85900	141806385	0.679	651.79	0.699	2055.71	-0.020
	average	0.619	45.19	0.449	176.14	0.170

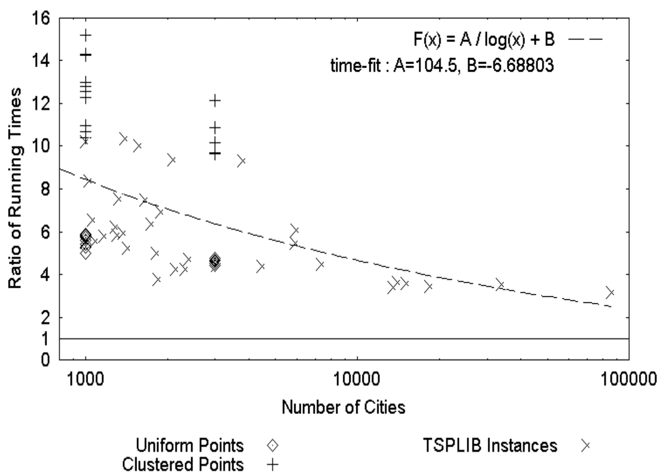

Figure 12. Ratio of running times between Concorde and proposed method.

5. 결 론

본 연구에서는 국지탐색 해법을 반복 수행하여 더 좋은 해를 찾기 위해 새로운 Perturbation 방법인 비용완화법을 제시하였다. 이 방법은 Lin-Kernighan 해법이 호들을 교환할 때 발생하는 단점을 보완함으로써 Lin-Kernighan이 탐색하지 못하는 이웃해를 탐색하고 효과적으로 국지해를 탈출하게 한다. 국지해를 탈출하여 초기해를 구성할 때 국지해에서 벗어나는 정도는 매우 중요하다. 본 연구에서는 효율적으로 국지해를 벗어나기 위해 완화하는 교점의 개수인 m을 고정하여 실험하였다. 실험에서 교점의 개수와 관계없이 $m = 100$일 경우 가장 좋은 해를 구하는 것으로 나타났다. 실험결과 본 연구에서 제시한 해법은 Chained Lin-Kernighan 해법을 사용한 Concorde에 비해 교점들이 군집을 이루어 분포하는 문제에서 가장 효과적이었다. 이는 호의 비용을 완화함으로써 군집 간에 연결할 수 있는 많은 경우를 고려하기 때문이다. 제시한 해법은 교점들이 균등하게 분포하는 문제와 교점들이 군집를 이루며 분포하는 문제에서 각각 Concorde보다 평균 0.128%와 0.9% 더 좋은 해를 발견하였다. 또한 제시한 해법은 TSPLIB의 34개 문제들 중 29 문제에서 Concorde보다 더 좋은 해를 구했다. 제시한 해법의 수행시간은 Concorde보다 약 3∼15배 더 소요되었다. 그러나 문제가 커짐에 따라 시간 차이가 줄어드는 것을 발견하였다.

참고문헌

Applegate, D., Bixby, R., Chvatal, V., and Cook, W. (1995), Finding Tours in the TSP, DIMACS Technical Report, 95-105.

Applegate, D., Bixby, R., Chvatal, V., and Cook, W. (1999), *Concorde*, Computer Software, http://www.math.princeton.edu/tsp/concorde.html.

Applegate, D., Cook, W., and Rohe, A. (1999), Chained Lin-Kernighan for Large Traveling Salesman Problems, Technical Report No. 99887, Forschungsinstitut fur Diskrete Mathematik, University of Bonn, Germany.

Codenotti, B., Manzini, G., Margara, L., and Resta, G. (1996), Perturbation: An Efficient Technique for the Solution of Very Large Instances of the Euclidean TSP, *INFORMS Journal on Computing*, **8**, 125-133.

Held, M. and Karp, R. M. (1970), The Traveling Salesman Problem and Minimum Spanning Trees, *Operations Research*, **18**, 1138-1162.

Helsgaun, K. (2000), An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristics, *European Journal of Operational Research*, **126**, 106-130.

Johnson, D. S. and McGeoch, L. A. (2002), Experimental Analysis of Heuristics for the STSP, *The Traveling Salesman Problem and Its Variations*, edited by Gutin G. and Punnen A., Kluwer Academic Publishers, Boston, MA, 369-444.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., and Shmoys, D. B. (1985), *The Traveling Salesman Problem*, John Wiley & Sons, NY.

Lin, S. (1965), Computer Solution of Traveling Salesman Problem, *Bell System Tech. Journal*, **44**, 2245-2269.

Lin, S. and Kernighan, B. W. (1973), An Effective Heuristic Algorithm for the Traveling Salesman Problem, *Operations Research*, **21**, 498-516.

Martin, O., Otto, S. W., and Felten, E. W. (1992), Large-step Markov Chains for the TSP Incorporating Local Search Heuristics, *Operations Research Letters*, **11**, 219-224.

Martin, O. C. and Otto, S. W. (1996), Combining Simulated Annealing with Local Search Heuristics, *Operations Research*, **63**, 57-75.

Reinelt, G. (1991), TSPLIB-A Traveling Salesman Library, *Operation Research Society of America Journal on Computing*, **3**, 376-384.

Reinelt, G. (1994), *The Traveling Salesman Problem*, Springer-Verlag, Berlin.

Zhang, W. (2002), Depth-First Branch and Bound Versus Local Search: A Case Study, *17th National Conf. on Artificial Intelligence (AAAI-2000)*, Austin, TX, 930-935.