분기 함수를 적용한 분산 최근접 휴리스틱
(A Distributed Nearest Neighbor Heuristic with Bounding Function)

김 정 숙 †
(Jung-Sook Kim)

요 약 외판원 문제는 잘 알려진 NP-완전 문제로, 최적해(optimal value)를 구하는 다양한 알고리즘들이 개발되었다. 그러나 최악의 경우 지수 시간이 걸리므로 수행시간을 줄이는 다양한 방법들이 제안되고 있다. 최근접 휴리스틱 알고리즘은 최적해를 구하는 다른 알고리즘들에 비해 구조가 비교적 간단하다. 따라서 본 논문에서는 외판원 문제(Traveling Salesman Problem, TSP)의 최적해를 구할 수 있는 분기 함수(bounding function)를 적용한 분산 최근접 휴리스틱(nearest neighbor heuristic) 알고리즘을 PVM (Parallel Virtual Machine)에서 제공하는 마스터/슬레이브(master/slave) 모델을 사용하여 설계하고 구현하였다. 먼저 최적해를 찾는 수행 시간을 줄이기 위해 최적화 문제에서 좋은 성능을 보이는 분산 유전 알고리즘(distributed genetic algorithm)을 수행해 얻은 근사해(near optimal)를 초기 분기 함수로 사용한다. 특히 더욱 좋은 근사해를 구하고자 유전 연산자인 돌연변이를 새롭게 변형하여 적용하였다.

키워드 : 분기 함수, 분산 최근접 휴리스틱, 외판원 문제, NP-완전, PVM, 유전 알고리즘, 돌연변이 연산자

Abstract The TSP(Traveling Salesman Problem) has been known as NP-complete, there have been various studies to find the near optimal solution. The nearest neighbor heuristic is more simple than the other algorithms which are to find the optimal solution.

This paper designs and implements a new distributed nearest neighbor heuristic with bounding function for the TSP using the master/slave model of PVM(Parallel Virtual Machine). Distributed genetic algorithm obtains a near optimal solution and distributed nearest neighbor heuristic finds an optimal solution for the TSP using the near optimal value obtained by distributed genetic algorithm as the initial bounding value. Especially, we get more speedup using a new genetic operator in the genetic algorithm.

Key words : bounding function, distributed nearest neighbor heuristic, traveling salesman problem, NP-complete, PVM, genetic algorithm, mutation

1. 서 론

외판원 문제는 주어진 n개의 도시들과 그 도시들간의 거리 비용이 주어졌을 때, 처음 출발도시에서부터 정확히 한 도시는 한 번씩만 방문하여 다시 출발도시로 돌아오면서 방문한 도시들을 연결하는 최소의 비용이 드는 경로를 찾는 문제로 최적해를 구하는 것은 전형적인 NP-완전 문제중의 하나이다[1, 2]. 출발도시에서 시작해서 나머지 (n-1)개의 도시를 한 번씩 거치는 경로의 총 수는 (n-1)! 만큼 존재하게 되어 계산 복잡도는 지수시간이 된다. 그리고 이 문제는 n개의 도시 방문 순서를 일부분 바꾸더라도 경로의 합이 증가하게 되는 국소 최소점이 많이 있는 문제이다. 이렇게 국소 최소점이 많은 울퉁불퉁한 표면(rugged landscape)에서 전체 최적해를 찾는 문제는 변수 공간 전체에 대한 검색을 하지 않는 한 불가능하다.

그러나 이렇게 복잡한 외판원 문제가 실제 다른 최적화 문제들, 예를 들어 VLSI 반도체 설계에서 연결선의 길이를 최소화하는 경우에 응용이 될 뿐만 아니라, NP-완전 문제들을 현실적으로 해결하려는 노력의 일환이 된다. 따라서 외판원 문제를 해결하기 위한 다양한 연구들이 시도되고 있다. 분기 한정법(Branch-and-Bound) 알고리즘[2]이나 동적 프로그래밍(Dynamic Programming) 알고리즘[2] 방법과 같이 최적해를 구하는 방법이 있고, 확률

· 이 논문은 2002학년도 김포대학 연구비 지원에 의하여 연구되었음.
† 정 회 원 : 김포대학 컴퓨터계열 교수
kimjs@kimpo.ac.kr
논문접수 : 2000년 12월 14일
심사완료 : 2002년 5월 8일

적 탐색 휴리스틱(probability search heuristic)에 근거해서 근사해(near optimal value)를 구하는 유전 알고리즘 등 다양한 방법들이 있다[3, 4, 5, 6, 7]. 그런데 최적해를 구하는 알고리즘들은 변수 공간 전체에 대한 검색을 하므로 최악의 경우 계산 복잡도가 지수 시간이 되어 많은 수행시간이 걸린다. 따라서 이들의 수행시간을 줄이고자 하는 연구가 많이 진행되고 있다. 특히 최적해를 구하는 방법들을 분산이나 병렬 환경에서 수행하는 알고리즘을 개발하거나, 또는 최적의 해를 구하는 알고리즘을 단독으로 사용하는 것 보다 여러 가지 다양한 알고리즘을 서로 조합한 하이브리드 알고리즘을 작성하여 수행 시간을 줄이고자 하는 방안들도 활발히 연구되고 있다. 지역 탐색(Local Search)과 분기 한정법의 조합 및 유전 알고리즘과 분기 한정법의 조합이 하이브리드 알고리즘의 예이다[1, 8].

본 논문에서는 외판원 문제의 최적해를 구하는데, 휴리스틱 알고리즘인 최근접 휴리스틱을 이용한다. 최근접 휴리스틱 알고리즘은 근사해를 구할 때 많이 사용하는 알고리즘으로 근사해의 성능이 다른 휴리스틱 알고리즘에 비해 좋지 않다. 그 이유는 최근접 휴리스틱 알고리즘은 구성 프로시듀어(construction procedure) 로 시작에서부터 주어진 규칙에 따라서 문제의 해를 점차 구성해가면서 해결하는 알고리즘으로 수행 도중 어떤 다른 성능 향상을 위한 변형을 적용하지 않기 때문이며, 또한 시작점을 어디로 했느냐에 따라 많은 성능의 차이가 있음을 알 수 있다. 그러나 최근접 휴리스틱 알고리즘의 구조는 비교적 간단하다.

그래서 본 논문에서는 최적의 해를 구할 수 있는 다른 알고리즘들에 비해 비교적 구조가 간단한 최근접 휴리스틱 알고리즘으로 모든 경로를 다 탐색해서 최적의 해를 구한다. 물론 수행 시간을 줄이고자 최적화 문제에서 좋은 성능을 보이는 유전 알고리즘으로 얻은 근사해를 초기 분기함수로 사용한다. 특히 유전 연산자인 돌연변이 연산자를 변형하여 유전 알고리즘에 적용하여 더욱 좋은 근사해를 구하도록 설계하였으며, 유전 알고리즘과 최근접 휴리스틱 알고리즘의 속성을 감안하여 병렬성을 추출하여 근거리 통신망(Local Area Network)에 기반한 분산 처리 환경에서 PVM[9]의 마스터/슬래이브 모델을 이용하여 분산시킨 알고리즘을 설계하고 구현하였다.

논문의 구성은 먼저 2장에서 유전 알고리즘과 최근접 휴리스틱 알고리즘으로 외판원 문제를 해결하는 연구를 살펴보고, 3장에서는 외판원 문제의 근사해를 구하는 유전 알고리즘의 성능이 향상될 수 있도록 개발된 돌연변이 연산자를 제안한다. 그리고 4장에서 PVM 환경에서 마스터/슬래이브를 이용한 효율적인 분산 최근접 휴리스틱 알고리즘을 설계한 내용을 기술하며, 5장에서는 본 논문에서 제안한 방법들을 실험한 내용과 방법들을 설명하고, 실험을 통해 얻어진 결과들을 비교 분석하였다. 마지막으로 6장에서 결론을 내리고 향후 연구과제를 제시한다.

2. 외판원 문제를 위한 알고리즘들

본 장에서는 외판원 문제에 대한 많은 연구들이 진행되고 있는데, 그 연구들 중에서 본 연구를 하는데 많은 영향을 준 기존의 알고리즘들을 살펴본다.

2.1 유전 알고리즘

유전 알고리즘을 특정 문제에 적용시키기 위해서는 개체가 문제의 해결책을 어떻게 표시하도록 할 것인지를 설계해야 하고, 개체의 우수한 정도를 나타내는 적응도를 계산하는 적합성 함수(fitness function)를 어떻게 정의할 것인지, 또 우수한 개체들을 집단에서 어떤 방법으로 선택하고 몇 개로 복제해야 하는지, 교차 연산과 돌연변이 연산을 어떻게 정의할 것인지를 결정해야 한다. 이들 적용 방법이 어느 정도로 문제의 특성을 잘 반영하고 있느냐에 따라 알고리즘의 성능이 크게 달라진다.

외판원 문제를 위한 유전 알고리즘들은 다양한 후보해(candidate population)의 표현과 유전 연산을 제시하고 있다. 후보해의 표현 방법은 정수 표현 방법을 사용하고 있으며, 유전 연산자는 주로 교잡 연산과 돌연변이 연산이 사용되고 있다. 따라서 외판원 문제의 입력자료로서 개체는 모든 도시들을 정수로 매핑하여 n개의 도시들을 임의로 생성 가능한 도시들의 배열로 입력한다. 이렇게 생성된 개체의 예를 들어 도시의 수가 17개인 경우 다음과 같다. 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 9 - 10 - 11 - 12 - 13 - 14 - 15 - 16 - 17, V는 도시 "1"에서 출발하여 도시 2를 방문하고 다음 도시 3을 방문하고 등으로 하여 도시 17을 방문하는 경로를 나타내며, V는 목적 함수 값으로 도시들을 방문하면서 걸린 비용의 합을 나타낸다.

초기 후보 집단은 이렇게 임의의 수들의 배열로 생성한 후 생성된 개체들에 대한 비용을 계산한다. 그 비용들을 정렬한 후 비용이 적은 우성 인자들을 이용하여 교잡 연산자와 돌연 변이 및 역순 연산자를 수행하였다. 교잡 연산자는 순서 교잡(Order Crossover, OX)를 사용했으며, 돌연변이는 다음 3장에서 제안한 방법을 이용하였다. 역순 연산자는 한 개체내에서 임의의 위치를 정하여 그를 기점으로 도시의 배열 순서를 역으로 바꾸는

것이다.

그리고 외판원 문제의 목적 함수는 간단한 함수를 적용하는데, 여행 경로중에 걸린 비용의 합을 목적 함수 값으로 정하여 이들에 대해 평가한다. 또한 우수한 개체들을 생성하기 위한 선택 방법은 순위 선택 방법으로 목적 함수값을 정렬하여 좋은 값을 가지는 개체들을 선택하여 일정한 비율은 다음 세대로 재생산(reproduction)하고 나머지 개체에서 유전 연산들을 행하여 새로운 자손을 만든다. 자손의 세대는 항상 일정하게 유지한다.

2.2 최근접 휴리스틱 알고리즘

최근접 휴리스틱 알고리즘은 주어진 문제를 해결하는데 어떤 주어진 규칙에 따라서 문제의 해를 구하고, 그 규칙에 따라 생성된 경로에 대해서는 다른 어떠한 개선을 위한 연산 예를 들면 교잡 연산이나, 돌연변이 연산 등을 적용하는 시도를 하지 않은 구성 프로시듀어 중의 한 방법이다.

외판원 문제를 해결하기 위해 처음 임의의 시작 도시에서부터 다음 방문해야 할 도시를 선정하는데 처음 도시에서 갈 수 있는 다른 모든 도시들의 비용을 비교하여 가장 적은 도시를 방문하도록 한다. 같은 방법으로 현재 방문한 그 도시에서 다음 방문할 도시를 찾는데, 그 도시에서 갈 수 있는 도시들 중에서 지금까지 방문하지 않은 도시이면서 가장 비용이 적은 도시를 선정하여 다음 방문 도시로 한다. 이러한 방법으로 계속 반복하다 주어진 모든 도시들을 방문하여 외판원이 다시 시작도시로 오면 멈추는 알고리즘이다.

그런데 이 알고리즘은 시작도시를 어디로 하느냐에 따라 얻어진 결과에 많은 차이가 있는 휴리스틱이다. 따라서 이 알고리즘에서 최상의 결과를 얻기 위해 처음 시작 도시를 어디로 해야 할지를 찾는 것이 중요한 일이다. 그러나 그 해를 구하기가 쉽지 않으므로 보통은 처음 시작 도시를 문제의 도시 순서에서 반 정도번째 되는 도시, 예를 들면 문제의 도시수가 17개라면 대략 8번째 도시나 9번째 정도의 도시를 처음 시작도시로 하여 방문을 시작한다[5]. 이를 뒷받침하는 이론이 Rosenkrantz, Stearns and Lewis(1977)[5]에 의해 발표되었다. 그 이론은 모든 $r > 1$ 이고, 임의의 큰 n이 주어졌을 때, 최근접 휴리스틱으로 도시수가 n개인 외판원 문제의 최적의 해를 구하기 위해서는 적어도 r번 이상 수행해야 한다는 것이다.

표준 최근접 휴리스틱 알고리즘의 수행 시간이 많이 걸리는 것을 보완하는 많은 변형된 알고리즘들이 제안되었다. 먼저 다중 시작점 알고리즘으로 시작점을 한 곳에서만 시작하여 수행하기 보다는 여러곳에서 시작하는 방법이 있다. 단지 시작점이 여러군데일 뿐 수행하는 알고리즘은 표준 최근접 휴리스틱 알고리즘과 동일하다. 다른 변형 알고리즘인 선계산 근접(Precomputed Neighbors) 방법은 k 최근접 부분 그래프가 있다고 가정하고, 어떤 노드에서 최근접 도시를 찾을 때, 그 노드가 속해 있는 부분 그래프 안에서 인접한 도시들을 먼저 찾고, 이미 그 부분 그래프에 있는 모든 노드가 방문한 경로에 포함되어 있을 경우에는 다른 모든 자유 노드에서 최근접 도시를 찾는 알고리즘이다. 그리고 다른 변형 알고리즘으로 후위노드의 근접(Neighbors of Prodecessors) 알고리즘, 회전 연산자 사용(Using Rotation Operations)등이 있다. 이런 모든 변형 알고리즘과 표준 최근접 휴리스틱 알고리즘에서의 공통점은 시작점을 어디로 하느냐에 따라 성능에 많은 영향을 받는다는 것이다[8].

본 논문에서는 모든 각 도시를 시작도시로 하여 최적의 해를 구한다.

3. 변형된 돌연변이 연산자

특성이 우수한 유전 인자를 만들어 내기 위해 유전자 조작 작업을 유전 연산(genetic operation)이라고 한다. 유전 알고리즘에서 사용되어 유용한 결과를 얻어내는 유전 연산으로는 돌연변이 연산(mutation), 교잡 연산(crossover), 복제 연산(reproduction), 그리고 역순 연산(inversion) 등이 있다. 이러한 연산자 중에서도 외판원 문제에서는 돌연변이 연산자와 교잡 연산자가 많이 사용된다. 교잡 연산자는 돌연변이 연산자에 비해 분산 정도가 크나 단점으로는 유용한 유전 정보를 잃기 쉬우며, 또한 조기 수렴(premature convergence) 현상에 빠지기 쉽다. 이에 반해 돌연변이 연산자는 분산 정도는 교잡 연산자보다 작으나, 한 개체내에서 연산이 일어나므로 다양성을 제공하여 조기 수렴 현상을 해결할 수 있다. 따라서 본 장에서는 새로운 자손이 부모보다 좋은 해를 가질 수 있는 돌연변이 연산자를 제안한다.

상호 교환 돌연변이(reciprocal exchange mutation) 연산자는 한 개체내에서 임의의 두 도시를 뽑아 그 두 도시의 위치를 서로 바꾸는 연산자이다. 본 논문에서 개발한 변형된 돌연변이 연산자는 상호 교환 돌연변이 연산자와 같은 연산을 수행하지만 두 도시를 임의로 선정하는 대신 비용을 고려하여 적은 비용이 걸리는 도시들을 선정하여 수행한다. 먼저 돌연변이 연산을 수행할 개체에서 비용을 다 비교하여 가장 많은 비용이 걸리는 경로를 선정한다. 그 선정된 도시들과 바로 이웃하는 방문 경로들의 비용을 비교하여, 비용이 더욱 많이 소요되는 한 도시를 뽑고, 그 뽑힌 도시에서 갈 수 있는 도시들 중 가장 비용이 적은 도시를 결정하여 다음 방문 경

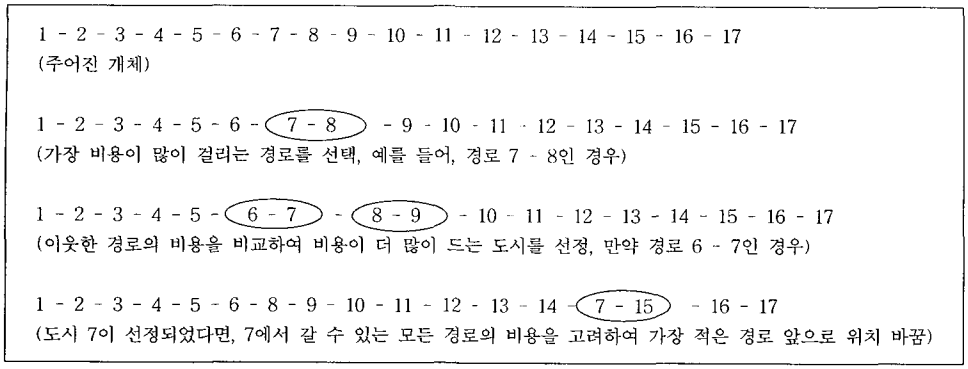

그림 1 도시 17개의 변형된 돌연변이

로로 연결한다.

예를 들어서 한 개체의 도시가 17개 주어졌을 때 돌연변이 연산자를 적용하여 연산한 과정은 다음과 같다. 경로 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 9 - 10 - 11 - 12 - 13 - 14 - 15 - 16 - 17 에서 한 도시를 선정하기 위해 각 경로의 비용을 비교하여 가장 많은 비용이 걸리는 경로의 도시 쌍을 결정한다. 예를 들어 도시 7 - 8 의 경로가 가장 많은 비용을 가진다면 가장 이웃하는 6 - 7 의 경로와 8 - 9 경로의 비용을 비교하여 더욱 비용이 많이 걸리는 도시를 선정한다. 예로 도시 7이 선정되었다면 도시 7에서 가능한 모든 경로들을 비교하여 가장 비용이 적게 걸리는 도시를 선택하여 다음 경로로 연결한다. 가장 비용이 적은 경로가 15이었다면 돌연변이가 일어난 결과는 다음과 같다. 1 - 2 - 3 - 4 - 5 - 6 - 8 - 9 - 10 - 11 - 12 - 13 - 14 - 7 - 15 - 16 - 17. 다음 그림 1에 예를 나타내었다.

4. 효율적인 분산 최근접 휴리스틱

PVM은 여러 가지 다른 기종의 병렬 또는 순차적 컴퓨터를 단일한 가상 병렬 컴퓨터(virtual parallel computer)로 이용하게 해주는 소프트웨어 시스템으로서, 분산 처리 환경에서 보편적으로 사용되는 모델은 주로 마스터/슬래이브 모델이 사용되어진다. 마스터는 프로세스들을 호출하고(spawning), 초기화(initialization)하며, 결과들을 모아서 화면에 보여주는 일들을 책임진다. 이에 반해 슬래이브 프로세서들은 마스터에게서 할당된 실제 계산들을 수행한다.

본 논문에서 도시의 수가 n이며, 도시들간의 거리 비용 distance[n][n] 이 주어진다. 도시 i와 도시 j사이의 거리 비용은 distance[i][j], 1 <= i, j <= n 으로 나타내며, 한 도시에서 다른 모든 도시로 갈 수 있는 경로가 있다고 한다.

하나의 프로세서가 마스터가 되고 여러개의 슬래이브 프로세서가 있다. 먼저 마스터에서 자신의 태스크 ID를 얻은 후 "slavename"를 pvm_spawn( ) 함수를 이용하여 시작하며 성공적으로 수행 시작이 끝난 뒤에는 초기의 후보해를 생성하여 pvm_send( ) 함수를 이용하여 각 슬래이브에게 전달한다. 후보해를 받은 각 슬래이브들은 자신에게 주어진 후보해에 유전 연산을 적용하여 우성인자들을 가진 자손을 만든다. 얻은 자손들의 적합도를 평가하여 가장 좋은 결과를 얻는 이러한 과정을 주어진 세대만큼 반복한 후 최상의 결과를 마스터에게 돌려준다. 그러면 마스터는 각 슬래이브로부터 온 결과들을 비교하여 가장 적합한 해를 구해 이 해를 다시 여러 슬래이브에게 전달하여 분산 최근접 휴리스틱 알고리즘의 초기 분기하는 규칙으로 사용하게 한다. 특히 더욱 좋은 근사해를 구해 초기 분기함수로 사용하면 분기하는 경우가 많아져 수행 시간이 많이 단축된다. 따라서 좋은 근사해를 구하기 위해 돌연변이 연산자를 변형하여 반드시 자손이 더욱 좋은 해를 구할 수 있도록 하였다.

분산 최근접 휴리스틱 알고리즘으로 최적해를 찾아내기 위해서는 주어진 문제의 모든 도시들을 시작도시로 하여 다 탐색을 해야 한다. 이 때, 만약 이용할 수 있는 충분한 프로세서가 있다면 각 시작도시를 새로운 프로세서에게 할당하며 각 프로세서는 최상의 해를 탐색해서 마스터에게 결과를 보내주면 된다. 그러나 이용할 수 있는 프로세서가 한정되어 있으므로 통신 연산 부담을 고려하여 적절하게 작업을 분할, 할당하여 병렬성을 얻을 수가 있다. 본 논문에서는 주어진 도시 n을 이용할 수 있는 슬래이브 프로세서에게 일정한 비율로 나누어 준다. 각 슬래이브 프로세서는 할당된 도시들을 시작도시로 하여 해를 찾는 과정에서 효율적으로 탐색하기 위

해 분기 함수를 적용한다. 분기 함수는 분산 유전 알고리즘으로 구한 근사해로, 각 경로를 탐색하면서 걸린 비용이 분기 함수보다 더 많은 비용이 소요되면 그 부분 경로의 탐색을 중단하고 새로운 경로를 탐색하는 것이다. 이와 같이 분기 함수를 적용하면서 해들의 집합을 구하여 그 해집합의 원소 중 최소의 비용을 드는 경로를 부분 최적해로 선택하여 마스터에게 돌려준다. 마스터는 각 슬레이브들로부터 반환된 결과들을 비교하여 가장 적은 비용을 가진 해를 최적의 해로 선정한다.

다음 알고리즘 1은 본 논문에서 제안한 분산 최근접 휴리스틱 알고리즘이다.

```
알고리즘 분산최근접휴리스틱( )
{
  Generates the populations;
  Sends the subpopulations to the slaves;
  Executes the distributed genetic algorithm then
returns the near optimal to the master;
  Selects the best near optimal among returned
values from the slaves;
  Transmits the initial bounding value and the
starting cities;
  Executes the distributed nearest neighbor heuristic
and returns the result;
  Decides the optimal among returned results;

}
```

알고리즘 1. 분산 최근접 휴리스틱 알고리즘

5. 실험 방법 및 결과

5.1 실험 환경

본 논문에서 제시한 내용들을 C 언어로 PVM 3.3 환경에서 C를 지원하는 라이브러리를 이용하여 구현하였으며, 실험 환경은 Sun 워크스테이션을 이더넷으로 연결하여 사용한다. 유전 알고리즘에서 사용한 파라미터는 후보해의 크기 : 10, 자손의 세대 : 50, 교잡 연산자의 비율 60%, 재생산 연산자의 비율 20%, 돌연변이 연산자의 비율 : 10%, 역순 연산자의 비율 : 10%이다. 그리고 외판원 문제에 대한 거리의 비용은 외판원 문제를 연구하는 많은 연구자들이 인용하고 있는 TSPLIB[10]에 있는 br17.atsp, ftv33.atsp와 ry48p.atsp 등과 같은 값들을 사용하였으며, 일부는 임의로 생성하여 실험하였다.

5.2 실험 방법

실험은 먼저 하나의 프로세서에서 유전 알고리즘을 수행할 때 본 논문에서 제시한 변형된 돌연변이 연산자를 적용하여 유전 알고리즘을 수행하여 얻은 근사해와 상호 교환 돌연변이 연산자를 적용하여 구한 근사해가 얼마만큼 최적의 해에 근접했는지를 최근접 휴리스틱으로 최적의 해를 구하는 실험을 각각의 방법에 대해 10번씩 수행하였다. 각 10번의 결과들에 대한 평균을 구해 비교하였다.

그리고 분산 최근접 휴리스틱 알고리즘의 성능을 평가하기 위해 도시의 수를 17, 33, 48, 75, 100과 같이 점차적으로 증가시키면서 최적의 해를 구하는데 걸린 수행시간을 분기 함수를 적용하지 않을 때와 분기 함수를 적용할 때 걸린 수행 시간을 10번씩 실험하여 얻은 결과값의 평균을 구해 비교하였다. 수행 시간은 유전 알고리즘을 수행하여 근사해를 구하는 시간과 근사해를 전달하여 최근접 휴리스틱 알고리즘을 수행하여 최적의 해를 구할 때까지를 측정한 것이다. 분산 처리하는 과정에서 유전 알고리즘은 한 프로세서에서 후보해들을 생성하여 여러개의 프로세서에 나누어 준 다음 각 프로세서에서 자신들에게 주어진 후보해를 가지고 유전 연산을 수행하였다. 모든 프로세서에서 수행되는 유전 연산의 비율, 세대수 및 후보해의 크기는 동일하게 적용되었다. 그리고 분산 최근접 휴리스틱 알고리즘의 실험에서 통신망내의 부하는 고려하지 않는다. 통신망내의 부하가 얼마이냐에 따라 수행 시간에 많은 영향을 미치며 실험 시 수행하고 있는 동안에 얼마든지 다른 임의의 노드로부터 작업이 시작될 수 있기 때문이다. 본 실험에서는 특정한 서버들을 정해 그 서버들에게 다른 작업이 수행되지 않도록 하면서 사용자가 많이 사용하지 않은 시간을 이용하여 실험하였다.

5.3 실험 결과

그림 2의 결과는 단일 프로세서에서 유전 알고리즘을 수행할 때 상호 교환 돌연변이 연산자와 본 논문에서 제안한 새로 변형된 돌연변이 연산자를 적용하여 얻은 근사해가 최적의 해에 얼마나 근접했는지를 비교한 것

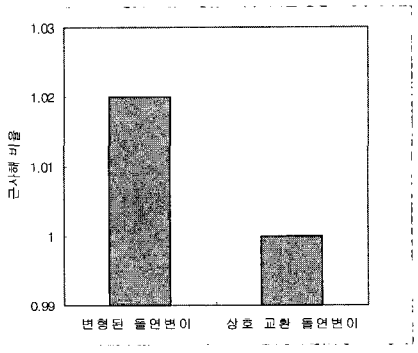

그림 2 변형된 돌연변이 연산 결과 비율

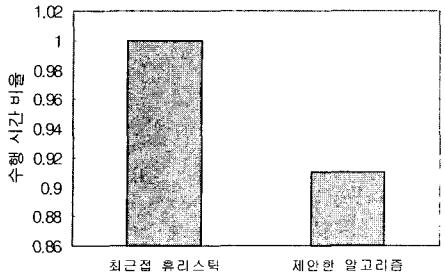

그림 3 수행 시간 비율

이다. 도시간의 거리에 대한 비용을 임의로 생성하여 먼저 변형된 돌연변이 연산자를 적용한 유전 알고리즘을 수행하여 결과를 얻고, 다음으로 상호 교환 돌연변이 연산자를 적용한 유전 알고리즘을 수행하여 결과를 얻고 최종으로 최근접 휴리스틱을 수행하여 최적의 해를 구하여 비교하는 실험을 하였다.

그림 3은 도시의 수를 17, 33, 48, 75, 100과 같이 점차적으로 증가시키면서 유전 알고리즘으로 근사해를 구해 이를 최근접 휴리스틱 알고리즘의 분기하는 규칙에 적용한 수행 시간과 유전 알고리즘을 적용하지 않은 최근접 휴리스틱 알고리즘만으로 해결한 방법의 수행 시간을 측정한 결과이다. 여기서 외판원 문제는 도시의 수가 증가하는 양과 비례하여 경로의 수가 생성되는 것이 아니라 지수적으로 증가한다. 따라서 유전 알고리즘으로 근사해를 구해 적용하면 비록 간단하고 빠르게 처리하여 얻은 근사해이지만 단순하게 최근접 휴리스틱 알고리즘으로 최적의 해를 구하는데 효율적인 분기를 할 수 있다. 그 결과 그림 2에서 알 수 있듯이 유전 알고리즘을 적용한 방법이 그렇지 않은 방법에 비해 더욱 빠른 수행 시간이 얻어짐을 알 수 있다. 실험은 비용을 다르게 하면서 10번씩 수행하여 얻어진 결과값들의 평균을 구한 것이고 마스터와 슬래이브간에 후보해 전송, 분기값 전송 및 근사값 전송은 PVM을 이용하였다. 그리고 최근접 휴리스틱 알고리즘을 수행하는 과정에서 유전 알고리즘에서 구한 해가 최적해에 근접한 근사해일수록 더욱 수행 시간을 줄일 수 있음을 알 수 있다. 비록 시간 복잡도(time complexity)는 표준 최근접 휴리스틱 알고리즘과 동일하지만 실제 실험 결과 수행시간이 적게 걸림을 알 수 있다.

그림 4는 한 개의 마스터와 슬래이브의 수를 2개 3개로 점차 늘리면서 본 논문에서 제안한 분산 최근접 휴리스틱 알고리즘을 실험한 결과를 보인 것이다. 실험은

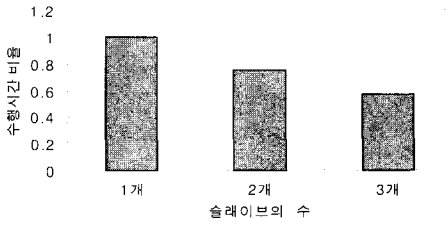

그림 4 슬래이브 수에 따른 수행시간 비율

각 도시에 대해 슬래이브의 수를 2개로 하고 실험을 10번 반복하여 얻은 결과들을 평균하였으며, 슬래이브의 수를 3개로 늘려서도 같은 방법으로 실험하였다. 그림 4를 보면 슬래이브 수가 1개 일 때와 비교해 3개로 늘려 실험한 결과 약 40% 정도 수행시간이 적게 걸림을 알 수 있다. 수행시간이 슬래이브 수에 비례해서 감소하지 않은 이유는 마스터에서 후보해를 생성할 때 걸린 시간이나, PVM을 이용해서 각 마스터에게 후보해와 분기값을 전송 할 때 가공하는 시간등이 슬래이브 수에 비례해서 증가하기 때문이다.

6. 결론 및 향후 연구과제

통신망으로 연결된 혼합 기종 컴퓨터들의 집합을 동시 계산 자원으로 다룰 수 있는 단일화된 시각을 제시하는 PVM은 프로그래머가 부가적인 부담 없이 프로그래밍에 전념할 수 있게 한다. 이러한 환경에 힘입어 계산 복잡도가 지수적인 문제들을 위한 효율적인 분산 알고리즘의 개발은 더욱 중요하다.

외판원 문제는 주어진 n개의 도시들과 그 도시들간의 거리 비용이 주어졌을 때, 처음 출발도시에서부터 정확히 한 도시는 한 번씩만 방문하여 다시 출발도시로 돌아오면서 방문한 도시들을 연결하는 최소의 비용이 드는 경로를 찾는 문제로 최적해를 구하는 것은 전형적인 NP-완전 문제중의 하나이다. 본 논문에서는 외판원 문제의 최적의 해를 반드시 구할 수 있는 분기 함수를 적용한 효율적인 분산 최근접 휴리스틱 알고리즘을 설계하고 구현하였다. 먼저 최근접 휴리스틱 알고리즘을 수행하면서 유전 알고리즘으로 얻은 근사해를 초기 분기 함수값으로 사용하여 수행 시간을 줄였으며, 특히 유전 연산자인 돌연변이를 변형하여 유전 알고리즘을 수행하므로써 더욱 좋은 근사해를 얻을 수 있었다. 그리고 LAN으로 연결된 분산 시스템의 PVM 환경에서 마스터/슬래이브 모델을 사용하여 분산시켜 수행함으로써 더욱 빠른 수행 시간을 얻을 수 있었다.

앞으로의 연구과제는 최근접 휴리스틱 알고리즘을 수행하는 도중 생성되는 경로의 정보를 이용하여 좀 더 빨리 최적해를 구하는 방법을 연구하는 일과, 일반적인 NP-완전 문제의 최적해를 빨리 구할 수 있는 효율적인 분산 알고리즘을 설계하는 일이다.

참 고 문 헌

[1] C. Cotta, J.F. Aldana, A.J. Nebro, J.M. Troya, "Hybridizing Genetic Algorithms with Branch and Bound Techniques for the Resolution of the TSP," Springer-Verlag, Artificial Neural Nets and genetic Algorithms, pp. 277-279, 1995.

[2] E. Horowitz, S. Sahni, Computer Algorithms, *Computer Science,* pp. 370-421, 1978.

[3] T.C. Sapountzis, "The Traveling Salesman Problem," *IEEE Computing Futures,* pp. 60-64, Spring, 1991.

[4] M. Gen, R. Cheng, Genetic Algorithms and Engineering Design, pp. 118-132, 1997.

[5] D. E. Goldberg, Genetic Algorithms : in Search and Optimization, *Addison-Wesley,* pp. 1-125, 1989.

[6] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, *Springer-verlag,* pp. 209-237, 1995.

[7] G. Reinelt, The Traveling Salesman Computational Solutions for TSP Applications, *Springer-Verlag,* 1994.

[8] J. Kim, Y. Hong, "A Distributed Hybrid Algorithm for the Traveling Salesman Problem," *IASTED, AI'99,* Feb. 1999.

[9] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, V. Sunderam, "PVM : Parallel Virtual Machine, A User's Guide and Tutorial for Networked Parallel Computing," *The MIT Press,* 1992

[10] TSPLIB, *http://www.iwr.uni-heidelberg.de/iwr/comopt/ soft/TSPLIB95/ATSP.html.*

김 정 숙

1993년 2월 동국대학교 컴퓨터공학과 졸업(공학사). 1995년 2월 동국대학교 대학원 컴퓨터공학과 졸업(공학석사). 1999년 8월 동국대학교 대학원 컴퓨터공학과 졸업(공학박사). 2000년 3월 ~ 현재 김포대학 컴퓨터 계열 조교수