최단경로 탐색을 위한 ACO 알고리즘의 비교 분석

최경미*, 박영호*
*숙명여자대학교 멀티미디어과학과
e-mail: {kmchoi0990, yhpark}@sookmyung.ac.kr

Analysis on ACO Algorithm for Searching Shortest Path

Kyung-Mi Choi*, Young-Ho Park*
*Dept. of Multimedia Science, Sookmyung Women's University

요 약

최근 ITS(Intelligent Transportation Systems)의 개발과 함께 차량용 내비게이션의 사용이 급증하면서 경로탐색의 중요성이 더욱 가속화되고 있다. 현재 차량용 내비게이션은 멀티미디어 및 정보통신 기술의 결합과 함께 다양한 기능 및 정보를 사용자에게 제공하고 있으며 이러한 기능과 정보를 사용해서 목적지점까지의 최단경로를 탐색하는 것이 내비게이션 시스템의 핵심기능이다. 이러한 경로탐색 알고리즘은 교통시스템, 통신 네트워크, 운송 시스템은 물론 이동 로봇의 경로 설정 등 다양한 분야에 사용되고 있다. 개미 집단 최적화(Ant Colony Optimization, ACO) 알고리즘은 메타 휴리스틱 탐색 방법으로 그리디 탐색(Greedy Search)뿐만 아니라 긍정적 반응의 탐색을 사용한 모집단에 근거한 접근법으로 순환 판매원 문제(Traveling Salesman Problem, TSP)를 풀기 위해 처음으로 제안되었다. 본 논문에서는 개미 집단 최적화(ACO) 알고리즘이 기존의 경로 탐색 알고리즘으로 알려진 Dijkstra 보다 최단경로 탐색에 있어서 더 적합한 알고리즘이라는 것을 설명하고자 한다.

1. 서론

개미 집단 최적화(Ant Colony Optimization, ACO) 알고리즘은 조합 최적화 문제를 해결하기 위한 메타 휴리스틱(Meta Heuristics) 탐색 방법으로, 그리디 탐색(Greedy Search) 뿐만 아니라 긍정적 반응의 탐색을 사용한 모든 집단에 근거한 접근법이다. 메타 휴리스틱은 지역 최적화를 피하기 위한 알고리즘으로 Colorni, Dorigo, Maniezzo가 처음으로 개미 집단 알고리즘을 제안하였으며, 지역 갱신과 전역갱신을 통해 최적의 해답을 찾는 알고리즘이다[3][4]. 조합최적화 문제인 순회 판매원 문제(Traveling Salesman Problem), 순서 문제(Sequential Ordering Problem), 차량 경로 문제(Vehicle Routing Problem), 이차 배정 문제(Quadratic Assignment Problem), 일정 계획 문제(Job-shop Scheduling Problem), 그래프 착색 문제(Graph Coloring Problem), Telecommunications Networks 등에서 최적의 해를 구하기 위해 사용되는 메타 휴리스틱방법으로 유전자 알고리즘(Genetic Algorithm, GA), 타부서치(Tabu Search, TA), 시뮬레이티드 어닐링(Simulated Annealing, SA), Lin-Kernighan(LK) 알고리즘 등이 있다[1]. 최근 ITS(Intelligent Transportation Systems)의 개발과 함께 차량용 내비게이션의 사용이 급증하면서 경로 탐색의 중요성이 더욱 가속화되고 있으며, 이러한 경로 탐색 알고리즘은 교통시스템, 통신 네트워크, 운송시스템은 물론 이동 로봇의 경로 설정 등 다양한 분야에 사용되고 있다. 이에 본 논문에서는 개미 집단 최적화(ACO) 알고리즘이 기존 경로탐색 알고리즘보다 적합한 알고리즘이라는 것을 설명하고자 한다. 본 논문의 구성은 2장에서는 경로 탐색 알고리즘에 대해서 설명하고, 3장에서는 개미 집단 최적화 알고리즘 적용 사례에 대해서 설명한다. 4장에서는 기존의 경로 탐색 알고리즘과 비교, 분석한다. 마지막으로 5장에서는 본 논문의 결론을 맺도록 한다.

2. 경로 탐색 알고리즘

본 장에서는 경로 탐색 알고리즘에 대해서 설명한다. 2.1절에서는 개미 시스템(AS) 알고리즘에 대해서 설명하고, 2.2절에서는 개미 집단 시스템(ACS) 설명 및 수행방법, ACO 알고리즘에 대해서 살펴본다.

2.1 개미 시스템(Ant System, AS)

개미 시스템(AS)은 실제 개미들이 먹이에서 둥지까지 가장 짧은 경로로 찾는 능력을 모방한 메타 휴리스틱 탐색 기법으로 개미들이 목적지를 향해 나아

가는 동안 각 경로에 페로몬을 분비하고, 이후에 지나가는 개미들은 그 경로에 쌓여있는 페로몬 정보를 이용해 다음 경로를 선택하는 원리를 휴리스틱 탐색에 적용시킨 시스템이 개미 시스템(AS)이다.

2.2 개미 집단 시스템(Ant Colony System, ACS)

개미 집단 시스템(ACS)는 개미 시스템(AS)의 성능을 향상시키기 위해 Dorigo와 Gambardella[4]에 의해 소개되었다. 개미 시스템(AS)는 짧은 경로가 있으면 그것만을 선택하고자 하는 성질로 인하여 국부 최적에 빠질 확률이 높아지기 때문에 이러한 문제를 쉽게 해결하기 위해 확률 분포를 이용해서 다음 노드를 선택하는 과정을 추가한 개미 집단 시스템(ACS)이 연구되었다.

개미 집단 시스템(ACS)의 수행방법은 다음과 같다. 먼저 개미들이 초기화 규칙에 따라 무작위로 노드를 선택한 다음, 각 개미들은 상태전이(State Transition Rule)에 따라 다음에 방문할 노드를 선택하고 계속해서 탐색과정을 거친다. 이러한 과정을 거치는 동안 개미들은 지역 갱신 규칙(Local Updating Rule)에 따라 방문한 각 노드(Node)에 페로몬 양을 갱신하게 된다. 이러한 방법을 반복한 후 모든 개미들이 탐색과정을 마치게 되면 전역 갱신 규칙(Global Updating Rule)에 따라 다시 페로몬 양을 갱신하게 된다. 결국, 각 개미들은 짧은 노드를 선택하려는 휴리스틱 정보와 많은 양의 페로몬을 가진 노드를 선택하려는 페로몬 정보에 따라 탐색 경로를 완성하게 된다. (그림 1)는 개미 집단 최적화 알고리즘의 Pseudocode는 다음과 같다[5].

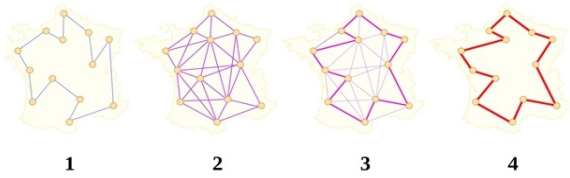
```
Procedure: ACO algorithm

    Set parameters, initialize pheromone trails;
    While (termination condition not met) Do
      ConstructAntsSolutions;
      UpdatePheromones;
      DaemonActions; (optional)
    End
End
```

(그림 1) Ant Colony Optimization 알고리즘

3. 개미 집단 최적화 알고리즘 적용 사례

순회 판매원 문제(Traveling Salesman Problem, TSP)는 NP-hard 조합 문제로 이 문제를 개미 집단 최적화 알고리즘을 이용하여 해결할 수 있다. (그림 2)는 순회 판매원 문제(TSP)는 판매원이 집에서 출발하여 모든 고객들의 도시에 방문하고 다시 집으로 돌아오는 최단경로를 찾는 문제이다. 이 문제에서는 다음과 같은 전제 조건이 있다[6].



① 매판원의 집도 도시로 취급한다.
② 모든 도시들은 서로 연결되어 있다.
③ 모든 도시들은 한 번씩만 거쳐야 한다.
④ 모든 도시들은 거친 뒤 다시 매판원의 집으로 돌아와야 한다.

(그림 2) Traveling Salesman Problem

TSP는 그래프 G=(N, A)로 나타낼 수 있으며, N은 n=|N|이다. A는 Arc를 의미하고, 각 Arc (i, j)∈A에는 weight로서 i, j의 distance(두 도시간의 거리) d_{ij}가 할당된다.

(그림 3)은 ACO를 적용한 알고리즘의 TSP를 구하는 Pseudocode는 다음과 같다.

개미는 처음에 랜덤하게 시작 노드를 정해 위치시킨다. 페르몬을 업데이트 할 때는 짧은 경로일수록 가중치를 주기 위해서 d_{ij}에 반비례하게 한다. 모든 개미가 모든 노드를 거치게 되면 프로그램이 종료하게 된다. TSP의 결과는 프로그램이 종료된 후 그래프의 각 Arc(경로)에 남아 있는 페로몬의 수치가 높은 순으로 선택하면 하나의 루프가 형성된다. 그것이 TSP의 최적의 해가 된다.

```
Start Node : Each ant is initially put on a
ramdomly chosen start city

While(all ants don't pass by every nodes)
   while(an ant don't pass by every nodes)
      ant ← getNextMovableNode()
      ant ← chooseNextNode()
   end-while
   Arc ← updatePheromone(ant's path)
End-while
```

(그림 3) ACO 알고리즘을 적용한 TSP

4. 알고리즘과 비교 및 분석

본 장에서는 기존의 경로 탐색하는 방법으로 알려진 Dijkstra 알고리즘과 ACO 알고리즘을 비교 및 분석한다. Dijkstra 알고리즘은 최단경로를 탐색하는 알고리즘으로 철도건설, 통신네트워크, 항공기 운항 계획 등 목적지에 이르는 경로를 찾아야 하는 분야에 사용되었다. Dijkstra 알고리즘에 대해서 설명하면, 특정 시작점에서 각 목적지까지 이르는 최단 경로를 구하는 알고리즘으로 각 정점에 이르는 최단 경로를 시작점의 주변에서부터 하나씩 확장해가면서 서서히 범위를 넓힌 후 최종적으로 모든 정점에 이르는 최단 경로를 구한다. 하지만, Dijkstra 등과 같은 Polynomial Time의 시간 복잡도를 가지는 최단 경로 알고리즘은 노드의 수가 증가함에 따라 경로 탐색 시간이 오래 걸리는 단점이 있다[2]. 이에 본 논문에서는 최단경로 탐색 문제를 NP-complete 문제로 보고 노드의 수가 증가하더라도 빠른 시간 내에 적절한 경로를 탐색하는 알고리즘인 개미 집단 최적화(ACO) 알고리즘을 사용함으로써 노드의 수가 증가하더라도 빠른 시간 내에 최단경로을 탐색한다.

5. 결론 및 향후 연구

본 논문에서는 최단 경로 탐색을 위한 개미 집단 최적화(Ant Colony Optimization, ACO) 알고리즘에 대해서 설명하였고, 기존의 최단경로 탐색방법으로 알려진 Dijkstra 알고리즘과 비교, 분석하였다. 개미 집단 최적화(ACO) 알고리즘은 다양한 분야에서 사용되는 알고리즘으로 교통시스템, 통신 네트워크, 운송 시스템은 물론 이동 로봇의 경로 설정 등에 사용되고 있으며, 앞으로 ACO 알고리즘을 사용한 많은 어려운 문제를 해결할 수 있는 연구가 필요하겠다.

이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(2011-0002707).

참고문헌

[1] 이승관, 최진혁, “개미 집단 최적화에서 강화와 다양화의 조화, ” 한국콘텐츠학회논문지 제 11권 제 3 호, 2011.

[2] 옥승호, 안진호, 강성호, 문병인, “선호도 기반 최단경로 탐색을 위한 휴리스틱 융합 알고리즘,” 전자공학회 논문지 제47권 TC 편 제 8 호, 2010.

[3] A. Colorni, M. Dorigo, and V. Maniezzo, “An investigation of some properties of an ant algorithm,” Proceediings of the Parallel Parallel Problem Solving from Nature Conference(PPSn 92), R. Manner and B. Manderick(Eds.), Elsevier Publishing, pp.509-520, 1992.

[4] L.M. Gambardella and M. Dorigo, “Ant Colony System: A Cooperative Learning approach to the Traveling Salesman Problem” IEEE Transactions on Evolutionary Computation, Vol.1, No1, 1997.

[5] Seung-Ho, Woo-Jin Seo, Jin-Ho Ahn, Sungho Kang and Byungin Moon, "An ant colony optimization approach for the preference-based shortest path search," Journal of the Chinese Institute of Engineers, Vol.34, No.2, 2011.

[6] http://en.wikipedia.org/wiki/Ant_colony_optimization