계층 군집을 이용한 효율적인 이동 집단의 소요 이동 시간 알고리즘 설계 및 구현

송예지(YeJi Song)*, 전태현(TaeHyeon Jeon)*, 안진현(Jinhyun Ahn)**,
임동혁(Dong-Hyuk Im)***
*호서대학교 컴퓨터공학과
*e-mail:dldfma0105@gmail.com, xoqus294@gmail.com
**제주대학교 경영정보학과
**e-mail:jha@jejunu.ac.kr
***광운대학교 정보융합학부
***e-mail:dhim@kw.ac.kr

Design and Implementation Algorithm of Efficient Travel Time for Mobility Groups Using Hierarchy Clusters

YeJi Song*, TaeHyeon Jeon*, Jinhyun Ahn**, Dong-Hyuk Im***
*Dept. of Computer Engineering, Hoseo University
**Dept. of Management Information System, Jeju National University
***School of Information Convergence, Kwangwoon University

요 약

본 논문은 동일 목적지를 가진 다수의 사람들이 동일 이동 수단을 탑승하여 이동할 경우 발생하는 소요 시간의 감소를 위한 거점 군집에 대해 다루려 한다. 실제 도로망에서는 노드에 해당하는 사람들의 위치가 유동적으로 변화 하고 사람 간의 거리 값 또한 변화하게 된다. 따라서 동일한 목적지를 가진 다수의 사람들이 한 대의 차량을 탑승하여 이동한다고 가정할 때 차량의 최단 경로 및 소요 시간 감소를 목적을 기반으로 설계하였다. 차량의 최단 경로를 감소시키 위해서 사람들을 군집화 하는 방법을 사용하였고 그 결과 경유지가 감소 되어 차량의 이동 경로 값은 감소되었다. 또한, 전체 이동 소요 시간 역시 군집을 통해 감소시킬 수 있다. 본 논문에서는 이와 같이 군집을 이용한 최단 경로의 감소와 전체 이동시간 감소에 대한 알고리즘을 설계하고 구현하였다.

1. 서론

동일한 목적지를 가진 다수의 사람들이 동일한 이동 수단을 탑승하게 되는 경우는 주변에서 쉽게 찾을 수 있다. 각기 다른 위치에 거주하는 학생들이 학원에 등원하기 위하여 학원 차량을 탑승하는 경우, 회사에 출근하기 위해 통근 차량을 탑승하는 경우 등이 존재한다. 보통의 경우 차량을 운행하는 단체에서 사람들의 분포와 위치를 고려하여 노선을 만들고 적당한 위치에 정류소를 지정한다. 이는 결국 차량의 이동 거리와 소요 시간을 감소시키기 위함이 목적이다. 하지만 단순히 차량의 이동 비용을 감소시키기 위하여 사람 수에 비해 적은 양의 정류소를 세우는 경우 사람들의 이동 비용이 커지게 되므로 차량 운영 단체에 불만을 표하거나 해당 단체를 이용하지 않을 수도 있다. 따라서 차량의 이동 비용과 사람들의 이동 비용을 감소시키는 적절한 정류소의 수와 위치가 중요하다.

이 문제는 동일하게 그래프상의 최단 경로 탐색에도 적용할 수 있다. 그래프상의 각 노드를 이동 가능하도록 하고 군집화 하면 출발 노드부터 도착 노드까지의 전체 이동 소요 시간을 단축시킬 수 있다. 최단 경로의 전체 소요 시간을 단축시키기 위해서 노드들을 군집화하고 각 군집의 중심으로 노드들을 이동시킨 후 해당 위치를 이용해 최단 경로를 탐색하는 방법을 이용한다.

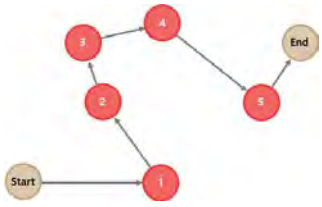
(그림 1)은 노드가 5개일 때 최단 경로의 예시이다. 출발, 도착 노드를 포함한 모든 노드간의 거리 계산은 <표 1>과 같다. <표 1>의 좌 하단 영역에 표시된 것과 같이 (그림 1)의 선택된 경로의 거리 총합은 14.64이다.

노드 하나의 최대 이동 가능한 거리를 2.5라고 한다면 (그림 2)와 같이 노드 2, 3, 4는 군집화할 수 있

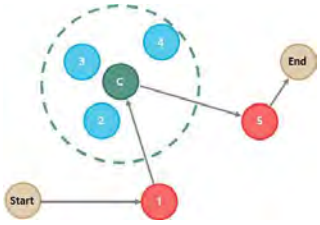
다. 이때 중심 위치를 C로 두어 다른 노드와의 거리를 모두 구해 table에 추가할 수 있다. (그림 2)에서는 노드 2, 3, 4의 위치가 C로 이동했다고 가정하여 계산된 최단 경로이며 이때의 거리 값은 <표 1>의 우 상단 영역 표시된 것과 같이 12.1이다. (그림 2)처럼 군집화되어 이동된 노드는 이동하며 소요된 시간 값이 존재하므로 가장 멀리 이동한 노드의 값을 전체 값에 더하여준다. 군집C에서 가장 멀리 이동한 노드는 4이며 C와 4의 거리는 1.41이므로 최종 값은 13.51이 된다. (그림 1)과 (그림 2)의 거리 값의 차는 1.13으로 감소했다는 것을 알 수 있다.

<표 1> 노드간 거리 Table

	Start	1	2	3	4	5	C	End
Start		3.5	2.83	3.81	5.32	6.32	3.91	7.83
1	3.5		2.5	4.03	4	3.2	3.16	4.95
2	2.83	2.5		1.58	2.5	4	1.12	5.22
3	3.81	4.03	1.58		2.06	4.74	1.12	5.5
4	5.32	4	2.5	2.06		3.2	1.41	3.54
5	6.32	3.2	4	4.74	3.2		3.64	1.8
C	3.91	3.16	1.12	1.12	1.41	3.64		4.53
End	7.83	4.95	5.22	5.5	3.54	1.8	4.53	



(그림 1) 최단 경로.



(그림 2) 군집 후 경로.

2. 관련 연구

본 논문에서 다루고자 하는 소요 시간 감소 최단 경로는 전체 노드를 일정 거리 값을 가진 노드들끼리 군집시키는 방법을 사용한다. 이는 다시 하나의 군집 내에서 최적의 미팅 장소를 탐색하는 문제로 이해할 수 있다.

Dijkstra 알고리즘[1]의 경우 주어진 그래프 내에서 출발 노드와 도착 노드 사이의 최단이 되는 경로를 탐색하게 된다. 본 논문에서 다루는 최단 경로의 그래프는 모든 노드를 빠짐없이 방문해야 하며 각 노드를 연결하는 간선은 모든 노드간 이어진 형태로 그려진다. 따라서 전체 노드를 방문해야 하는 경우에는 적용하기 어렵다.

해밀턴 경로 알고리즘은 그래프상의 전체 노드를 모두 경유하는 알고리즘이다[2]. 이는 본 논문에서 다루고자 하는 경유지를 모두 거치는 최단 경로와 가장 유사한 알고리즘이지만 주어진 노드를 1번씩만 거치는 것에 중점을 두고 있으므로 최단 경로를 보장하지 않는다. 따라서 해당 알고리즘을 기초로 모든 노드를 탐색하도록 하지만 간선의 가중치를 이용해 그 경로 값이 최소가 되도록 하는 알고리즘을 구상하여야 한다.

단일 그룹 콜렉티브 여행 질의 처리[3] 방법은 본 논문에서 제시하는 방법의 일부와 같은 해결 방안을 제시하고 있다. 여러명의 사용자들이 특정 지점에 모여 하나의 운송 수단을 통해 도착지로 이동하며 발생하는 전체 비용을 최소로 하는 지점을 찾는 문제이다[3]. 하지만 해당 방법은 하나의 그룹 단위가 모이게 되는 특정 지점을 찾아내는 것이므로 전체 노드를 여러 그룹으로 나누어 찾는 문제와 완벽히 일치하지는 않는다.

3. 군집화 설계

각 노드는 이동할 수 있는 범위가 존재하며 이를 이용하여 군집하기 위해 계층 군집[4]을 사용하였다. moblie group clustering 알고리즘은 해당 값을 *m*으로 하는 Input data를 전달받는다. 또한, 출발 노드와 도착 노드를 제외한 모든 노드는 *NS*(all nodes set)라는 노드 집합으로 전달받는다.

알고리즘1은 계층 군집[4]과 유사하게 작동한다. Input data로 전달받은 *NS*와 군집화된 노드들을 저장할 *CL*(clusters list)을 합하여 *NC*(nodes and clusters list)를 생성하고 *NC*내의 원소의 수가 1인 경우, 다시 말해 전체 노드들이 모두 군집화되어 1개의 군집이 될 때까지 이하 작업을 반복한다. *NC* 내에 존재하는 모든 원소들은 원소끼리의 거리를 모두 계산하여 *table*에 저장한다. 이는 계층 군집[4]을 위한 모든 노드들의 거리 값을 항상 보유하기 위함이다. *table*이 업데이트되면 *table*내에서 원소간의 거리가 짧은 것을 찾아 그 거리와 해당하는 원소들을 각각 *shortest*, *nodes*에 저장한다. 이때 찾아낸 최소 거리 값이 *m*보다 큰 경우 모든 원소간의 거리는 하나의 노드가 이동 가능한 범위 값을 초과하기 때문에 더이상 군집을 진행하지 않게 된다. 만약 *shortest* 값이 *m*보다 작거나 같다면 군집이 가능하므로 군집을 진행한다. 찾은 *nodes*가 군집되지 않은

노드인지 군집인지 검사하여 해당하는 집합에서 삭제하고 *nodes*를 하나의 군집으로 군집화하여 새롭게 *CL*에 추가한다.

군집 단계가 종료되면 전체 노드 집합은 n개의 군집과 군집 되지 않은 노이즈 노드로 나뉘어 진다. 노이즈는 *NS*에 남아있으며 군집들은 *CL*에 존재하게 된다. *NS* 집합에 속한 노드들은 이동하지 않으므로 변화 없이 반환하며 군집들은 각 군집의 중앙 위치와 거리 값을 계산한 뒤 반환하게 된다. *CL*내의 군집 하나를 *cluster*라고 한다면 해당 *cluster*의 중앙 위치를 구해 *center*에 저장한다. *cluster*의 *center*위치와 *cluster*내의 각 노드들은 거리를 계산하여 *center*와 가장 멀리 떨어진 노드의 거리 값을 찾아 *t*(travel time)에 저장한다(15 ~ 18). 하나의 *cluster* 연산이 완료될 때마다 *cluster*, *center*, *t*의 값을 묶어 군집 테이블인 *CT*(cluster table)에 저장한다.

Algorithm1 moblie group clustering

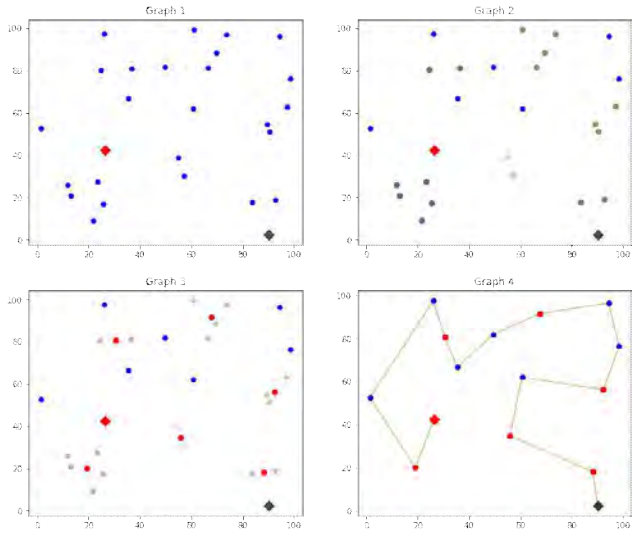
Input data: ***NS*** *(all node set),* ***m*** *(distance at which one node can move)*

1. *NC* = *NS* union *CL*
2. WHILE size of *NC* >= 1
3. *table* = CALCULATE distance between elements from *NC*
4. *shortest*, *nodes* = smallest distance and nodes with smallest distance from *table*
5. IF *shortest* > *m*
6. BREAK
7. ELSE
8. IF *nodes* are there *CL*
9. REMOVE *nodes* from *CL*
10. ELSE
11. REMOVE *nodes* from *NS*
12. *cluster* = group up nodes
13. add *cluster* to *CL*
14. *NC* = *NS* union *CL*
15. FOR *i* = 0 to *i* < size of *CL* do
16. *cluster* = get cluster from *CL*[*i*]
17. *center* = CALCULATE center position of *cluster*
18. *t* = find farthest element within *center* from *cluster*
19. add group up *cluster*, *center*, *t* to *CT*
20. RETURN *NS*, *CT*

알고리즘1은 노이즈만 남게 된 *NS*집합과 군집들의 정보가 저장된 *CT*를 함께 반환하며 종료한다.

4. 구현

군집화 및 최단 경로 알고리즘의 구현 모습은 (그림 3)과 같다. 25개의 파란색 원형 노드와 빨간색 마름모 도형인 출발 노드, 검정색 마름모 도형인 도착 노드로 구성되어있다. 출발, 도착 노드를 포함한 Graph 1상에 존재하는 모든 노드는 2차원 좌표상의 랜덤한 위치에 생성된다. Graph 2와 Graph 3에 걸쳐 모든 노드는 일정 범위 내의 노드들과 군집화 하여 각 군집의 중심으로 좌표로 이동한다. Graph 2에서 각 군집은 서로 다른 색으로 표현되며 군집에 속하지 못한 노이즈 노드들은 기존과 동일한 파란색으로 표시된다. Graph 3에서 각 군집들은 중심 좌표를 계산하여 빨간색 원형 도형을 생성하여 표현한다. Graph 4에서는 노이즈와 군집의 중심점인 파랑, 빨강 원형 도형만을 고려하여 최단 경로를 탐색해 결과를 그래프에 선으로 표현한다.



(그림 3) 군집화 및 최단 경로 탐색.

5. 결론

이동 가능한 노드들의 군집을 이용한 최단 경로 알고리즘의 이동 시간 단축에 대하여 설계하였다. 그래프상의 이동 경로가 아닌 실제 도로망의 실시간 소요 시간 변경에 대하여 각 경유 위치의 군집화는 전체 이동 시간에 영향을 주게 된다.

향후 연구과제로는 군집화되지 않은 집합과의 전체 소요 비용과의 비교 실험을 진행하여 효율성에 대한 증명을 진행할 것이다. 또한, 출발 노드부터 도착 노드까지의 경로 탐색에서 모든 노드를 방문해야 하는

경로에 대한 최단 경로의 연산 방법 역시 ALT 알고리즘[5]을 적용하여 효율성을 높이는 방법을 적용할 계획이다.

Acknowledgement

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터지원사업의 연구결과로 수행되었음 (IITP-2020-2018-0-01417).
또한, 2018년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2018R1D1A1B07048380)

참고문헌

[1] E.W. Dijkstra “A note on two problems in connexion with graphs” Numerische mathematik 1(1959) p 269-271 1959-01-01
[2] B. BOLLOBAS, T.I. FENNER and A.M. FRIEZE “an algorithm for finding hamilton paths and cycles in random graphs” combinatorica 7(4) p 327-341 1986-9-19
[3] 이준규(Junkyu Lee), 박석(Seog Park) “도로 교통망 환경에서 G-트리 구조를 이용한 단일 그룹 콜렉티브 여행 질의 처리” 정보과학회논문지 vol.47, no.5, pp.513-525 2020
[4] Corpet F. “Multiple sequence alignment with hierarchical clustering. Nucleic” Acids Res. 16(22) : 10881-90. 1988 Nov 25
[5] 이용현, 임동혁, 구해모, 김형주 “관계형 데이터베이스 상에서 사용하는 최단 경로 탐색 알고리즘(ALT) 설계와 구현” 정보과학회 컴퓨팅의 실제 논문지 제25권 제4호 215-222(8 pages) 2019.04