JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 26 No. 3, pp. 111-117, March 2021
https://doi.org/10.9708/jksci.2021.26.03.111

Optimal Solution Algorithm for Delivery Problem on Graphs

Kwang-Eui Lee*

*Professor, Dept. of Applied Software Engineering, Dong-eui University, Busan, Korea

[Abstract]

The delivery problem on a graph is that of minimizing the object delivery time from one vertex to another vertex on a graph with m vertices using n various speed robot agents. In this paper, we propose two optimal solution algorithms for the delivery problem on a graph with time complexity of $O(m^3n)$ and $O(m^3)$. After preprocessing to obtain the shortest path for all pairs of the graph, our algorithm processed by obtaining the shortest delivery path in the order of the vertices with the least delivery time. Assuming that the graph reflects the terrain on which to solve the problem, our $O(m^3)$ algorithm actually has a time complexity of $O(m^2n)$ as only one preprocessing is required for the various deployment of n robot agents.

▸ **Key words**: Delivery problem, Path planning, Graph, Robot agent, handover

[요 약]

그래프에서의 배달문제는 m개의 정점으로 구성된 그래프에서 n개의 서로 다른 속도를 갖는 로봇 에이전트들을 이용하여 배달물을 그래프의 한 노드에서 다른 노드로 배달하는 최소 배달순서를 구하는 문제이다. 본 논문에서는 그래프에서의 배달문제에 대하여 최적해를 계산하는 $O(m^3n)$과 $O(m^3)$시간복잡도를 갖는 두 개의 알고리즘을 제안한다. 알고리즘은 그래프의 모든 쌍에 대한 최단경로를 구하는 전처리를 한 후, 최소배달시간이 작은 정점의 순으로 최단배달경로를 구하는 방법으로 개발하였다. 이 문제에서 그래프가 문제를 해결하고자 하는 지형을 반영하고 있다고 하면, 다양한 로봇 에이전트의 배치에 대하여 전처리를 1회만 실행되면 되므로 $O(m^3)$알고리즘은 실제로 $O(m^2n)$의 시간복잡도를 갖는다고 할 수 있다.

▸ **주제어**: 배달문제, 경로계획, 그래프, 로봇에이전트, 핸드오버

• First Author: Kwang-Eui Lee, Corresponding Author: Kwang-Eui Lee
*Kwang-Eui Lee (kelee@deu.ac.kr), Dept. of Applied Software Engineering, Dong-eui University
• Received: 2021. 01. 14, Revised: 2021. 02. 09, Accepted: 2021. 02. 09.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

배달문제(delivery problem)는 m차원 유클리드 공간에서 임의의 위치에 주어진 배달물을 목적지까지 다수의 로봇 에이전트를 이용하여 최소시간에 배달하는 방법을 찾는 문제이다[1]. 이때 배달에 참여하는 로봇 에이전트들은 각각 다른 초기위치와 이동속도를 갖는다. 각각의 로봇은 다양한 초기위치와 이동속도를 갖고 있기 때문에 최소시간에 배달하는 배달과정 중에 (하나의 로봇이 다른 로봇에게 배달물을 넘겨주는) 핸드오버가 여러 번 발생할 수 있다. 따라서 배달문제는 그래프에서 배달물이 배달되는 경로를 기본으로 하여 핸드오버의 위치와 각각의 핸드오버에 참여하는 두 개의 로봇들을 쌍으로 구성된 순서열의 형태로 해를 표현할 수 있게 된다.

배달문제는 경로계획문제의 한가지 유형으로 볼 수 있다[2]. 경로계획과 관련된 초기의 연구결과를 보면 주로 하나의 로봇에 대한 경로계획 문제가 연구되었으며[3][4], 이후 다수의 로봇에 대한 경로계획 문제가 연구되었다[5][6][7][8]. 하지만, 대부분의 경우 로봇 자원과 평균대기시간의 최적화에 초점이 맞추어져 있었다. 또한 이러한 문제들에 대하여 시간복잡도 개선을 위하여 다양한 휴리스틱 알고리즘들도 제안되었다[9]. 본 논문에서 다루는 최소배달시간에 관한 연구로는 2009년에 문제의 정의와 함께 m차원 유클리드공간에서의 배달문제를 해결하기 위한 유전자 알고리즘이 최초로 제시되었으며[1], 이후 2011년에 m차원의 유클리드 공간에서 핸드오버가 가능한 장소를 제한한 경우의 알고리즘이 제시되었다[10]. 2014년에는 좀 더 현실적인 유용성이 있는 트리에서의 배달문제에 대한 최적 알고리즘이 제안되었는데[11], 트리에서의 배달문제는 좀 더 일반적이라 할 수 있는 그래프에서의 배달문제의 제한된 형태로 연구되었으며, 주요 아이디어는 트리에서는 하나의 노드에서 다른 노드로의 이동 경로가 유일함을 이용하여 문제를 해결하였다.

본 논문에서는 그래프에서의 배달문제를 해결하는 2개의 알고리즘은 제안한다. 각각은 $O(m^3n)$와 $O(m^3)$의 시간복잡도를 갖는다. 논문의 구성은 2절에서는 배달문제를 해결하는 데 필요한 용어들을 정의하고, 알고리즘의 정확성을 증명하는데 필요한 몇몇 보조정리 들을 정리한다. 3절에서는 두 개의 알고리즘을 제시하고, 바르게 동작함을 증명한다. 마지막으로 4절에서는 결론 및 향후 연구 방향에 대하여 기술한다.

II. Preliminaries

배달문제는 배달될 물건의 시작 위치와 배달되어야 하는 도착 위치 그리고 배달에 관여하는 n개의 로봇 에이전트들의 초기위치와 각각 로봇 에이전트들의 정해진 이동속도로 구성된다. 이때 최소시간의 배달을 위하여 배달 중간에 여러 번의 핸드오버가 발생하게 된다. 실제로 배달문제의 해인 배달순서는 배달물이 배달되는 경로를 기반으로 하여 배달에 참여하는 로봇의 순서와 이 로봇들 간의 핸드오버가 발생하는 위치의 순서열로 표현할 수 있게 된다.

본 논문에서 다루는 그래프에서의 배달문제에서는 각각의 로봇 에이전트가 초기에 그래프의 정점에 위치하고, 그래프의 간선을 따라 이동한다. 또한, 배달될 물건의 시작위치와 도착 위치도 그래프의 정점이 되며, 핸드오버도 각각의 정점에서만 가능한 경우를 다룬다.

문제를 단순화하기 위하여 로봇 에이전트들이 배달물을 집고, 핸드오버하고, 목적지에 놓기 위한 시간은 걸리지 않는다고 가정한다. 이제 배달문제를 해결하기 위하여 다음의 용어들과 변수들을 정의한다. 다음의 내용은 기본적으로 앞의 논문들에서와 동일하나, 그래프에 적용하기 위하여 부분 수정하였으며, 논문의 완전성을 위하여 이전 논문에 기술된 내용을 포함하여 다시 기술한다.

$V = \{v_1, v_2, ..., v_m\}$: 그래프의 정점 집합. 단, 이후 표현의 간편성을 위하여 v_i를 필요에 따라 $v(i)$ 또는 $v[i]$로 표기하며 이러한 표기법의 확장은 아래의 변수들에도 적용한다.

v_s : 배달 시작 위치

v_d : 배달 도착 위치

$w(x,y), 1 \le x,y \le m$: 정점 v_x와 v_y를 연결하는 간선의 길이

$d(x,y), 1 \le x,y \le m$: 정점 v_x와 v_y를 연결하는 최단경로의 길이

$r_i, 1 \le i \le n$: i번째 로봇에이전트

$p_i, 1 \le i \le n$: r_i의 초기위치

$s_i, 1 \le i \le n$: r_i의 이동속도. 이때, 모든 로봇에이전트의 이동속도가 상이할 필요는 없다.

위와 같은 정의하에 길이 k의 v_x에서 v_y로의 배달순서를 다음과 같이 기술한다.

$v_x = u_0 b_1 u_1 ... b_i u_i ... b_k u_k = v_y$

여기서, u_i는 정점의 색인을 의미하며, b_i는 정점 u_{i-1}에서 정점 u_i로의 배달을 담당하는 로봇의 색인을 의미한다. 이때, 배달순서를 그래프의 정점을 기준으로 작성하며, 이에 따라 b_i와 b_{i+1}은 같은 로봇 에이전트 일수 있으며, 이 경우 실제로 핸드오버는 발생하지 않는다. 이 정의에 따라 두 개의 정점 v_x, v_y에 대하여 v_x에서 v_y로의 배달순서는 여러 개 존재할 수 있음을 알 수 있다. 이제 주어진 배달순서 $v_x = u_0 b_1 u_1 ... b_i u_i ... b_k u_k = v_y$에 대하여 $T(u_0 b_1 u_1 ... b_i u_i ... b_k u_k)$를 $u_0 b_1 u_1 ... b_i u_i ... b_k u_k$의 순서로 배달할 때 가장 짧은 배달시간이라 하면, v_x에서 v_y로의 최단배달순서는 v_x에서 v_y로의 배달순서 중 $T(v_x ... v_y)$가 가장 작은 배달순서가 된다. 주어진 v_x에서 v_y로의 최단배달순서도 여러 개 존재할 수 있다. 또한 이후의 논의에서 $T(v_x)$와 같이 T에 배달순서가 아닌 하나의 정점만 주어지는 경우 시작노드 v_s에서 임의의 노드 v_x로의 최단배달시간의 표기로 사용한다. 이제 배달문제는 v_s에서 v_d로의 최단배달순서를 구하는 문제로 정의할 수 있다.

이러한 정의에 따라 v_x에서 v_y까지의 최단배달순서 $v_x = u_0 b_1 u_1 ... b_i u_i ... b_k u_k = v_y$가 주어지면 이러한 최단배달순서로 배달하는 최소시간은 다음의 단계들로 구할 수 있다.

단계 1. 먼저, 로봇 에이전트 b_1이 정점 u_0에 도착한 후 u_0에 있는 배달물을 u_1에 배달한다. 이때 소요되는 시간은 먼저 로봇 에이전트 b_1이 u_0에 도달하여야 하므로 $d(p(b_1), u_0)/s_1$의 시간이 필요하며, 다시 배달물을 u_0에서 u_1으로 배달하여야 하므로 $w(u_0, u_1)/s_1$의 시간이 필요하다. u_i까지 배달에 필요한 최소시간을 dt_i라 하면, $dt_1 = d(p(b_1), u_0)/s(b_1) + w(u_0, u_1)/s(b_1)$이 된다.

단계 2. 다음, 로봇 b_2가 정점 u_1에 도착한 후 u_1에 있는 배달물을 u_2에 배달한다. 이때 배달물도 u_1에 존재해야 한다. 이때 소요되는 시간은 먼저 배달물과 b_2가 모두 u_1에 도착한 후에 b_2가 배달물을 u_1에서 u_2로 배달하게 되므로 배달물과 b_2가 모두 u_1에 도착할 때까지 소요되는 시간은 $\max(dt_1, d(p(b_2), u_1)/s(b_2))$이 되고, 배달물이 b_2에 의하여 u_1에서 u_2로 배달되는 데 $w(u_1, u_2)/s(b_2)$의 시간이 필요하므로 $dt_2 = \max(dt_1, d(p(b_2), u_1)/s(b_2)) + w(u_1, u_2)/s(b_2)$가 된다.

이후, $u_2 b_3 u_3 ... b_k u_k = v_y$에 대하여 위의 단계 2와 동일한 과정을 반복하면 주어진 배달순서에 대하여 최소배달시간을 구할 수 있다. 이를 알고리즘을 표현하면 다음 표1의 알고리즘과 같다.

Table 1. Minimum Delivery Time Algorithm

알고리즘 1. 주어진 배달순서의 최소배달시간 계산
```
input : ds = u[0]b[1]u[1]...b[i]u[i]...b[k]u[k]
dt=0; // dt[i]를 저장하는 변수
for (i=1; i<=k; i++) {
   dt = max(dt,d(p[b[i]],u[i-1])/s[b[i]]))
   dt = dt+d(u[i-1],u[i])/s[b[i]]
}
```

본격적으로 최단배달순서 알고리즘을 기술하기에 앞서 문제의 특성을 이해하는데 필요한 내용들을 다음의 보조정리들로 증명한다.

보조정리 1. v_x에서 v_y까지의 최단배달순서가 $v_x = u_0 b_1 u_1 ... b_k u_k = v_y$라 하자. 이때, b_i와 b_{i+1}이 서로 다르면, $s(b_i) < s(b_{i+1})$이 되는 최단배달순서가 존재한다. 즉, 핸드오버는 항상 더 빠른 속도를 갖는 로봇 에이전트에게 배달물을 넘기는 형태로 발생한다.

증명: b_i와 b_{i+1}이 다르다는 것은 s_i에서 핸드오버가 발생하였음을 의미한다. 그러나 $s(b_i) \geq s(b_{i+1})$인 경우 $r(b_i)$가 $r(b_{i+1})$에게 넘겨주지 않고 $v(u_i)$부터 $v(u_{i+1})$까지 직접 배달함으로서 시간을 늘리지 않고 배달할 수 있다. 이 논의는 반복적으로 적용 가능하며, 따라서 보조정리가 성립한다. □

보조정리 1은 항상 $s(b_i) \leq s(b_{i+1}), 1 \leq i \leq n-1$인 최단배달순서가 존재함을 의미하며, r_i가 배달물을 배달 중이라면, r_i보다 늦은 속도를 갖는 로봇 에이전트들은 더 이상 고려할 필요가 없음을 의미한다.

보조정리 2. 그래프의 임의의 정점 v_x에 대하여, $v_s b_1 u_1 ... b_k u_k b_{k+1} u_{k+1} ... v_x$가 v_s에서 v_x까지의 최단배달순서라 하자. 또한, $v_s g_1 h_1 ... g_j h_j = u_k$가 v_s에서 u_k까지의 최단배달순서라 하자. 그러면, $v_s g_1 h_1 ... g_j h_j b_{k+1} u_{k+1} ... v_x$는 v_s에서 v_x까지의 최단배달순서 중 하나가 된다.

증명: $v_s g_1 h_1 ... g_j h_j = u_k$는 v_s에서 v_k까지의 최단배달순서이므로 $T(v_s g_1 h_1 ... g_k h_k) \leq T(v_s b_1 u_1 ... b_k u_k)$가 된

다. 즉, 최소배달시간을 계산하는 알고리즘 1에서 dt_k 값이 더 작아지며, 이는 알고리즘의 절차에 따라 최소배달시간을 늘리지 않는다. 따라서 보조정리가 성립한다. □

III. The Proposed Scheme

1. An $O(m^3n)$ algorithm for delivery problem

이 장에서는 그래프에서의 배달문제에 대한 최적해 알고리즘을 제안한다. 이 알고리즘의 기본적인 구성은 다익스트라 알고리즘과 유사하다[11]. 대략적인 동작방법을 보면 다음과 같다. 먼저 v_s에서 그래프의 다른 모든 노드로의 최소배달시간은 존재하므로 이러한 최소배달시간 순서로 정점들을 나열한 순서를 $o_1, ..., o_m$ 이라 하자. 이때, $v_s = o_1$이 되며, 알고리즘의 진행과정에서 $o_1, ..., o_m$을 순서대로 찾아간다.

이제 알고리즘은 제일 먼저 v_s에서 o_1까지의 최소배달시간과 약간의 추가적인 정보를 구한다. 다음 단계에서 v_s에서 o_2까지의 최소배달시간과 약간의 추가적인 정보를 구한다. 이런 방법을 계속 적용하여 순차적으로 v_d까지의 최소배달시간과 약간의 추가적인 정보를 구하고, 이를 기반으로 v_s에서 v_d까지의 최단배달순서를 구하게 된다.

다음에서 전체적인 방법을 설명하고, 알고리즘을 기술한 후 증명하고 시간복잡도를 계산한다. 이때 이후 알고리즘의 진행상에서 시간의 효율성을 위하여 전처리 단계로 모든 정점간의 최단경로의 길이를 미리 구한다.

[전처리 단계] 이후의 활용을 위하여 그래프상의 정점들의 모든 쌍에 대하여 최단경로길이를 구한다. 이 과정은 플로이드의 알고리즘을 이용하여 $O(m^3)$ 시간에 계산가능하다[12].

이후의 단계들에서는 그래프의 모든 $o_1, ..., o_m$의 순서대로 각각의 v_s에서 o_i까지의 최소배달시간과 추가적인 정보를 구한다. 이때 추가적인 정보는 다음과 같다.

$T(o_i)$: v_s에서 o_i까지의 최소배달시간

$P(o_i)$: 위의 최소배달시간을 보장하는 배달순서에서 o_i바로 전 정점

$R(o_i)$: 위의 배달순서에서 배달물을 $P(o_i)$에서 o_i로 배달하는 로봇 에이전트의 색인

[단계 1] v_s에서 o_1까지의 최소배달시간과 경로 계산

이때, $v_s = o_1$이므로, $T(o_1) = 0$이 되며, $P(o_1) = 0$, $R(o_1) = 0$으로 특별한 계산은 필요하지 않다. 이때, $P(o_1) = 0$, $R(o_1) = 0$의 의미는 이전 정점과 해당하는 로봇 에이전트가 존재하지 않음을 의미한다.

[단계 2] v_s에서 o_2까지의 최소배달시간과 경로 계산

$v_s (= o_1)$에서 o_2까지의 최단배달순서에는 v_s와 o_2가 서로 인접한 정점이므로 핸드오버가 발생하지 않는다. 즉, 하나의 로봇이 자신의 위치에서 v_s로 이동한 후 배달물을 집고, o_2로 이동하게 된다. 단지 우리는 어떤 로봇이 어떤 정점으로 이동하는가를 모를 뿐이다. 따라서 그래프에서 이미 계산이 완료된 o_1을 제외한 모든 정점 v_x와 모든 로봇 r_i의 쌍에 대해서 r_i가 배달물을 v_s에서 v_x로 배달하는 시간을 구하고 이중 최솟값을 갖는 정점 v_x가 o_2이 됨을 알 수 있다. 이때 r_i가 배달물을 v_s에서 v_x로 배달하는 최소배달시간은 다음과 같이 구할 수 있다.

$d(p_i, v_s)/s_i + w(v_s, v_x)/s_i$

수식 구성의 세부적인 설명은 알고리즘 1의 설명에서 확인 가능하다. 전처리 단계에서 $d(p_i, v_s)$를 미리 계산하였고, 그 외의 모든 값들은 문제에서 주어진다. 여기서 v_x와 r_i의 쌍은 모두 $(m-1) \times n$개 존재하므로, 단계 2의 과정은 $O(mn)$의 시간복잡도로 해결할 수 있다. 무론 보조정리 1에 의하여 o_1보다 속도가 늦은 로봇을 제외시킬 수 있으나, 이 경우 논의가 복잡해지며, 시간복잡도에서 이득을 볼 수 없으므로 모든 로봇 에이전트에 대해서 계산한다.

[단계 3] v_s에서 o_3까지의 최소배달시간과 경로 계산

$v_s = o_1$에서 o_3까지의 최소배달경로는 $v_s o_3$가 되거나, $v_s o_2 o_3$이 된다. 따라서 그래프의 모든 정점 (이미 계산이 완료된 o_1과 o_2는 제외) v_x에 대하여 배달경로 $v_s v_x$로 배달되는 최소시간과 $v_s o_2 v_x$로 배달되는 최소시간을 계산한 후 이중 최솟값을 갖는 v_x가 o_3라고 할 수 있다. 이때, $v_s v_x$는 [단계 1]에서와 같은 방법으로 $O(mn)$시간에 계산 가능하다. 이제 $v_s o_2 v_x$의 경우를 보면, 보조정리2에 따라 $R(o_2)$가 배달물을 v_s에서 o_2까지 배달함을 가정할 수 있다. 이제 $T_{\max}(i) = \max(T(v_s o_2), d(p_i, o_2)/s_i)$라 하면, $T(v_s o_2 v_x) = \min\{T_{\max}(i) + d(o_2, v_x)/s_i \mid 1 \le i \le n\}$와 같이 구할 수 있다. [단계 1]에서와 유사하게 v_x와 r_i의

쌍은 모두 $(m-2)\times n$개 존재하므로, 단계 3의 과정은 $O(mn)$의 시간복잡도로 해결할 수 있다.

이후의 단계들을 진행하기 전에 다음의 보조정리가 필요하다.

보조정리 3. v_s에서 o_k까지의 최소배달순서는 v_s에서 출발하여 $\{o_1, o_2, ..., o_{k-1}\}$의 정점만을 거친 뒤 o_k에 도달하는 경로로 구성된다.

증명. v_s에서 o_k까지의 최소배달순서에 $V-\{o_1, o_2, ..., o_{k-1}\}$의 원소 v_y가 있다고 하자. 그러면, 배달물은 o_k에 배달되기전에 v_y에 배달되어야 하는데, 이때까지 걸리는 시간은 가정에 따라 o_k에 배달되는데 걸리는 시간보다 크다. $o_1, ..., o_m$의 정의에 따라 이러한 상황은 발생할 수 없다. 따라서 보조정리가 성립한다. □

[단계 k] v_s에서 o_k까지의 최소배달시간과 경로 계산

위와 같은 방법으로 $\{o_1, o_2, ..., o_{k-1}\}$에 대한 최소배달시간과 각각의 정점에 대한 $T(o_i)$, $P(o_i)$, $R(o_i)$이 구해졌다고 하자. 이제 보조정리 3.에 따라 다음과 같이 o_k에 대한 최소배달시간을 구할 수 있다.

아직 최소배달시간이 구해지지 않은 정점들의 집합 $V-\{o_1, o_2, ..., o_{k-1}\}$의 각각의 정점 v_x에 대하여 다음을 실행한다.

다음: 집합 $\{o_1, o_2, ..., o_{k-1}\}$의 임의의 정점 o_v와 r_i에 대하여 $\max(T(o_v), d(p_i, o_v)/s_i) + w(o_v, v_x)/s_i$를 최소로 하는 값을 구한다.

이제 이렇게 계산된 값 중 최솟값을 갖는 v_x가 o_k가 되며, 앞의 단계들과 마찬가지로 이때 해당하는 r_i가 $R(o_k)$가 된다. 시간복잡도는 o_v가 최대 m개이며, v_x가 최대 m개이고, r_i가 최대 n개이므로 $O(m^2n)$이 됨을 알 수 있다.

이 단계는 보조정리 1에 따라 $R(o_k)$보다 속도가 늦은 로봇에이전트를 제외함으로써 실행시간을 단축할 수 있다.

이와 같이 반복하여 그래프의 모든 정점 v_x에 대하여 $T(v_x)$, $P(v_x)$, $R(v_x)$를 구할 수 있다. 이때, $P(v_x)$는 각 정점 v_x에서의 최단배달순서상의 이전 정점을 가르키고 있으므로, 일단 그래프의 모든 정점 v_x에 대하여 $T(v_x)$, $P(v_x)$, $R(v_x)$이 계산되면, v_s에서 v_d까지의 최단배달순서는 각각의 정점 v_x의 $P(v_x)$ 체인을 이용하여 계산할 수 있다. 정점v_d에서 시작하여 순서대로 $P(v_x)$를 따라가면, v_s에 도달하는 최단배달순서가 구성된다. 이를 좀 더 형식적으로 표현하면 다음과 같다.

Table 2. Shortest Delivery Path Algorithm

알고리즘 2. 최단배달순서생성

```
ds = "";
vc = v[d];
while (vc!=v[s]) {
  ds = P[v[c]] + R[v[c]] + ds;
  vc = p[v[c]];
}
return ds;
```

정리 1. 표2의 알고리즘 2에 의하여 생성된 배달순서는 v_s에서 v_d까지의 최단배달순서이다.

증명. 알고리즘 2에 의하여 생성된 v_s에서 v_d까지의 최단배달순서를 $v_s b_1 u_1 ... b_k u_k = v_d$라 하자. 이제 이 배달순서보다 배달시간이 짧은 다른 배달순서 $v_s g_1 h_1 ... g_q h_q = v_d$가 있다고 하자. 이제, $u_k = v_d = h_q$이므로 두 개의 배달순서를 오른쪽에서 왼쪽으로 비교 했을 때, 최초로 달라지는 u_i가 존재하게 된다. 즉, 두개의 배달순서는 각각 $v_s b_1 u_1 ... b_i u_i b_{i+1} u_{i+1} b_k u_k = v_d$와 $v_s g_1 h_1 ... g_j h_j b_{i+1} u_{i+1} b_k u_k = v_d$이 된다. 그런데 알고리즘의 선택 과정에서 u_i가 선택되는 순간은 v_s에서 u_{i+1}까지의 최단배달순서의 이전 정점을 선택하는 단계에 해당하며, 이때, 선택가능한 이전 정점 중 배달시간이 가장 빠른 정점을 선택하였으므로 $T(v_s b_1 u_1 ... b_i u_i b_{i+1} u_{i+1})$는 $T(v_s g_1 h_1 ... g_j h_j b_{i+1} u_{i+1})$보다 작거나 같아진다. 또한 이후의 배단순서가 동일하므로 모순이 발생한다. 그러므로 배달순서 $v_s b_1 u_1 ... b_k u_k = v_d$는 최단배달순서이다. □

전처리과정에서 $O(m^3)$의 시간이 소요되고, [단계 1] [단계 m]까지 $O(m^3 n)$의 시간이 소요되며, 최단배달순서를 출력하는데, $O(m)$의 시간이 소요되므로 전체 알고리즘의 시간복잡도는 $O(m^3 n)$이 된다.

2. An $O(m^3)$ algorithm for delivery problem

이제 앞의 알고리즘의 시간복잡도를 개선하기 위하여 알고리즘의 각 단계에서 새롭게 추가되는 o_i에 대해서만 계산하던 $T(v_x)$, $P(v_x)$, $R(v_x)$를 각 단계에서 모든 정점에 대해서 유지하는 방법을 도입한다. 이제 $T(v_x)$, $P(v_x)$, $R(v_x)$의 정의에 약간의 확장이 필요하다. 좀 더

정확하게 다음의 단계 k의 종료 시점에서 $T(v_x)$의 의미는 $\{o_1, o_2, ..., o_k\}$의 정점만을 거쳐 v_x로 오는 최단배달순서의 최소배달시간이 되며, $P(v_x)$는 현재의 배달순서상에서 v_x바로 전 정점이 되고, $R(v_x)$는 현재의 배달순서 상에서 $P(v_x)$에서 v_x로 배달하는 로봇 에이전트가 된다. 각각의 단계에서 $T(v_x)$, $P(v_x)$, $R(v_x)$는 값을 가지고 있지 않을 수도 있다, 그러나 이 값들은 지속적으로 갱신되며, v_x가 o_i로 선정되는 순간 바른 값을 갖게 된다. 이제 제시되는 알고리즘의 각각 단계는 다음과 같다.

[전처리 단계] 이후의 활용을 위하여 그래프 정점들의 모든 쌍에 대하여 최단경로길이를 구한다.

[단계 1] v_s에서 o_1까지의 최소배달시간과 경로 계산

앞의 알고리즘과 마찬가지로 $v_s = o_1$이므로 o_1을 구하기 위한 특별한 계산은 필요하지 않다. 하지만, $T(v_x)$, $P(v_x)$, $R(v_x)$의 새로운 정의에 따라 각각의 정점 v_x에 대해서 $T(v_x)$, $P(v_x)$, $R(v_x)$를 다음과 같이 구한다.

$T(v_x) = \min\{d(p_i,o_1)/s_i + w(o_1,v_x)/s_i, 1 \le i \le n\}$

$P(v_x) = o_1$

$R(v_x) = 0$, 이전 로봇은 없음을 의미함.

이때, v_s와 v_x가 직접적인 연결이 없다면 $T(v_x) = \infty$가 된다. 단계 1의 과정은 $O(m)$개의 정점에 대하여 $O(n)$의 계산이 필요하므로 $O(mn)$의 시간복잡도로 해결할 수 있다.

[단계 2] v_s에서 o_2까지의 최소배달시간과 경로 계산

$T(v_x)$, $P(v_x)$, $R(v_x)$의 확장된 정의로부터 o_2는 o_1이외의 모든 정점 v_x중 $T(v_x)$를 최소로 하는 v_x가 됨을 알 수 있다. 이때 $T(v_x)$는 이미 계산되어 있다. 따라서 o_2를 결정하기 위해 $T(v_x)$중 최솟값을 구해야 하며, $O(m)$의 계산이 필요하다. 또한 o_2가 결정되면, $P(o_2)$, $R(o_2)$도 바로 결정된다. 이제 다음 단계를 위하여 $V - \{o_1, o_2\}$의 정점들에 대하여 $T(v_x)$, $P(v_x)$, $R(v_x)$의 값을 갱신할 필요가 있다. 그런데, 이때 $v_s (= o_1)$에서 v_x로의 가능한 경로는 o_1v_x와 $o_1o_2v_x$만 존재한다. 그리고 $T(o_1v_x)$는 이미 계산하여 알고 있으므로 $T(o_1o_2v_x)$만 계산하여 두 값 중 최솟값을 취하면 된다. 이제 $T(o_1o_2v_x)$는 $T_{\max}(i) = \max(T(v_1o_2), d(p_i,o_2)/s_i)$라 할 때, $T(v_1o_2v_x) = \min\{T_{\max}(i) + d(o_2,v_x)/s_i \mid 1 \le i \le n\}$와 같이 구할 수 있다. v_x와 r_i의 쌍은 모두 $(m-2)\times n$개 존재하므로, 단계 2의 과정은 $O(mn)$의 시간복잡도로 해결할 수 있다. 이 과정에서 $P(v_x)$, $R(v_x)$를 계산하는 것은 단순하다.

이제 좀더 일반적인 상황으로 $o_1, ..., o_{k-1}$까지 계산된 상황에서 o_k를 계산하는 상황을 보면 다음과 같다.

[단계 k] v_s에서 o_k까지의 최소배달시간과 경로 계산

$T(v_x)$, $P(v_x)$, $R(v_x)$의 확장된 정의로부터 o_k는 $V - \{o_1, o_2, ..., o_{k-1}\}$의 모든 정점 v_x중 $T(v_x)$를 최소로 하는 v_x가 됨을 알 수 있다. 그런데 우리는 단계 (k-1)에서 이미 필요한 $T(v_x)$를 유지하고 있다. 따라서 o_k를 결정하기 위해 $O(m)$의 계산이 필요하다. 또한 o_k결정됨에 따라 $P(o_k)$, $R(o_k)$는 바로 결정된다. 이제 $V - \{o_1, o_2, ..., o_k\}$의 정점 v_x들에 대하여 $T(v_x)$, $P(v_x)$, $R(v_x)$의 값을 갱신하여야 한다.

이제 v_x를 기준으로 $T(v_x)$ 갱신하는 과정은 다음과 같다. 정의에 따라 $T(v_x)$는 $\{o_1, o_2, ..., o_k\}$의 정점만을 중간정점으로 v_x에 도달하는 최소배달시간이다. 이를 계산하기 위해서는 이러한 배달경로들은 $\{o_1, o_2, ..., o_k\}$ 중의 하나를 v_x바로 이전 정점으로 갖게 된다. 그런데 우리는 이미 $\{o_1, o_2, ..., o_{k-1}\}$의 정점들에 대해서 이 정점들을 v_x바로 이전 정점으로 갖는 경우의 최소배달시간을 이전 단계의 $T(v_x)$로 알고 있다. 따라서 o_k를 v_x바로 이전 정점으로 갖는 경우의 최소배달시간을 계산하여 이 값이 기존의 $T(v_x)$보다 작은 경우, $T(v_x)$, $P(v_x)$, $R(v_x)$를 갱신하면 된다. 새롭게 계산된 최소배달시간이 기존의 $T(v_x)$보다 큰 경우 갱신은 일어나지 않는다.

이 과정을 다시 보면, 이미 $T(v_x)$는 계산되어 있고, $P(v_x)$를 o_k로 했을 때의 최소배달시간은 $T_{\max}(i) = \max(T(o_k), d(p_i,o_k)/s_i)$라 할 때, $T(v_x) = \min\{T_{\max}(i) + d(o_k,v_x)/s_i \mid 1 \le i \le n\}$와 같이 구할 수 있다. 이때, v_x와 r_i의 쌍은 모두 $(m-k)\times n$개 존재하므로, 단계 k의 과정은 역시 $O(mn)$의 시간복잡도로 해결할 수 있다.

제시된 방법의 정확성은 3.1절 에서와 동일하게 증명가능하며, 시간복잡도를 보면, 전처리과정에서 $O(m^3)$의 시간이 소요되고, [단계 1]에서 [단계 m]까지 $O(m^2n)$의 시

간이 소요되며, 최단배달순서를 출력하는데, $O(m)$의 시간이 소요되므로 전체 알고리즘의 시간복잡도는 $O(m^3)$이 된다.

이 문제에서 그래프는 지형을 반영하므로 동일한 지형에 대하여 배달 시작 위치와 배달 도착 위치, 그리고 로봇 에이전트의 초기위치와 속도가 바뀌는 다양한 반복적인 실행에 대해서 전처리를 1회만 실행한다고 하면, 이 알고리즘은 $O(m^2n)$의 시간복잡도를 갖는다고 할 수 있다.

IV. Conclusions

본 논문에서는 그래프에서의 배달문제에 대하여 $O(m^3n)$ 시간 복잡도의 최적해 알고리즘을 제안하고 이를 개선하여 $O(m^3)$시간복잡도의 최적해 알고리즘을 제안하였다. 제안된 그래프에서의 배달문제는 그동안 제안되온 배달문제의 가장 일반적인 형태라고 할 수 있으며, 알고리즘의 구조도 간단한 형태로 구성되어 이후 다른 연구에 도움을 줄 수 있을 것으로 기대한다.

앞으로의 연구에서는 가장 일반적인 형태라 할 수 있는 그래프에서의 배달문제에 대한 시간복잡도 하한을 구하는 연구가 필요하며, 배달물이 두 개 이상인 경우에 대한 연구와 로봇에이전트들이 동시에 두 개 이상의 배달물을 가지고 이동할 수 있는 경우에 대한 연구가 진행되어야 할 것이다.

ACKNOWLEDGEMENT

This Work was supported by Dong-eui University Foundation Grant (2014)

REFERENCES

[1] KwangEui Lee and JiHong Kim, “Genetic Algorithm for Delivery Problem,” IJCSNS, V9, N2, pp. 248-251, February 2009.

[2] Motion planning, *http://en.wikipedia.org/wiki/Motion_planning.*

[3] S. M. Lavalle, “*Planning Algorithms*,” Cambridge University Press, 2006.

[4] Hyungil Kim, “Path Planning for Cleaning Robots Using Virtual Map,” Journal of The Korea Society of Computer and Information, V14, N11, pp. 31-40, November 2009.

[5] D.K. Liu, D. Wang, G. Dissanayake, “A force field method based multi-robot collaboration,” Proc. IEEE Int. Conf. on Robotics, Automation and Mechatronics, Bangkok, Thailand, pp. 662-667, June 2006.

[6] Y. Guo, L.E. Parker, “A distributed and optimal motion planning approach for multiple mobile robots,” IEEE Int. Conf. on Robotics Automation, pp. 2612-2619, May 2002.

[7] K. Azarm and G. Schmidt, “Conflict-Free Motion of Multiple Mobile Robots Based on Decentralized Motion Planning and Negotiation,” IEEE Int. Conf. on Robotics and Automation, pp 3526-3533, vol. 4, April 1997.

[8] Kyeonah Yu and Su-Jin Cho, “Path-Planning for Group Movement in Dynamic Environments,” Journal of The Korea Society of Computer and Information, V18, N2, pp. 117-126, February 2013.

[9] K. Rajchandara, Dr. R. Baskaranb, Mr. K. Padmanabhan Panchuc, “Multiple Robot Path Planning Algorithms for Static Environment and Dynamic Environment: A Review,” International Journal of Pure and Applied Mathematics, V119, N10, pp. 1273-1290, 2018.

[10] KwangEui Lee, “Near Optimal Algorithm for Delivery Problem,” IJCSNS, V11, N12, pp. 25-28, December 2011.

[11] KwangEui Lee, “Optimal Solution Algorithm for Delivery Problem on trees,” Journal of The Korea Society of Computer and Information, Vol. 19, No. 2, pp 143-150, Feb. 2014.

[12] R. Neapolitan and K. Naimipour, “*Foundations of Algorithms Using C++ Pseudocode, 3rd Ed.*,” Addison Wesley, 2003.

Authors

Kwang-Eui Lee received the B.S., M.S. and Ph.D. degrees in Computer Science from Sogang University, Korea, in 1990, 1992 and 1997, respectively. Dr. Lee joined the faculty of the Department of Multimedia Engineering at Dong-eui University, Busan, Korea, in 2001. He is currently a Professor in the Department of Applied Software Engineering, Dong-eui University. He is interested in Computational Theory, Graph Algorithms, Big Data, and Data Visualization.