환승 저항을 고려한 운행시간표 기반 대중교통 다중 경로 탐색 알고리즘

A schedule-based Public Transit Routing Algorithm for Finding K-shortest Paths Considering Transfer Penalties

전 인 우* · 남 현 우** · 전 철 민***

* 주저자 : 서울시립대학교 공간정보공학과 석사과정
** 공저자 : 서울시립대학교 공간정보공학과 박사과정
*** 교신저자 : 서울시립대학교 공간정보공학과 교수

Inwoo Jeon* · Hyunwoo Nam** · Chulmin Jun***

* Dept. of Geoinformatics, Univ. of Seoul
** Dept. of Geoinformatics, Univ. of Seoul
*** Dept. of Geoinformatics, Univ. of Seoul
† Corresponding author : Chulmin Jun, cmjun@uos.ac.kr

Vol.17 No.3(2018)
June, 2018
pp.72~86

ISSN 1738-0774(Print)
ISSN 2384-1729(On-line)
https://doi.org/10.12815/kits.2018.17.3.72

Received 17 May 2018
Revised 8 June 2018
Accepted 26 June 2018

요 약

운행시간표 기반 대중교통 경로 탐색 알고리즘은 운행계획에 따른 정류장별 출·도착 시각을 이용하여 최소 이동 시간이 소요되는 단일 경로를 산출한다. 다만, 경로 계산 과정에서 환승 저항, 대안 경로 선택 등의 추가 요소들을 반영하는데 한계가 있다. 본 연구는 환승 저항 및 다중 경로 탐색이 반영된 개선된 RAPTOR 알고리즘을 제안한다. 환승 저항은 환승 시점에 적용되며, 교통수단 유형을 구분하여 적용하였다. 본 연구에서는 수도권 대중교통 이용 승객의 실제 이동 경로를 기준으로 개선 전·후의 알고리즘 결과를 분석하였다. 이를 통해 제시한 알고리즘이 승객의 다양한 경로 선택 기준을 반영한다는 것을 확인하였다.

핵심어 : 대중교통 경로 탐색, 운행시간표, 환승 저항, 다중 경로

ABSTRACT

Schedule-based public transit routing algorithm computes a single route that calculated minimum travel time using the departure and arrival times for each stop according to vehicle operation plan. However, additional factors such as transfer resistance and alternative route choice are not reflected in the path finding process. Therefore, this paper proposes a improved RAPTOR that reflected transfer resistance and multi-path searching. Transfer resistance is applied at the time of transfer and different values can be set according to type of transit mode. In this study, we analyzed the algorithm's before and after results compared with actual route of passengers. It is confirmed that the proposed algorithm reflects the various route selection criteria of passengers.

Key words : Public transit routing, Timetable, Transfer penalty, Multi-path

Ⅰ. 서 론

1. 개 요

대중교통 최단 경로 탐색 알고리즘들은 일반적으로 그래프 이론을 이용하여 경로를 탐색한다(Pyrga et al., 2008; Cionini et al., 2014; Wang et al., 2016). 데이터 구조로는 정류장을 나타내는 노드와 노드 간 연결을 나타내는 링크로 구성된 네트워크를 이용하며, 각 링크에는 이동 시간, 이동거리 등의 비용이 저장된다. 또한 경로선택에 영향을 주는 환승 저항 등도 링크의 비용으로 이용된다(Kim et al., 2017).

최근에는 그래프 이론 기반의 알고리즘뿐만 아니라 대중교통 운행시간표를 이용하는 알고리즘들이 개발되고 있다. 운행시간표 기반의 알고리즘들은 기존 노드-링크 방식의 데이터 구조가 아니라 노선별 운행시간표를 저장하는 고유의 데이터 형식을 이용하고, 그래프 이론의 경로 탐색 방식보다 연산속도가 빠르다는 장점이 있다(Witt, 2015; Strasser and Wagner, 2014).

다만, 대부분의 운행시간표 기반 알고리즘들은 운행계획에 따른 정류장 출·도착 시각만을 경로 탐색에 이용하고 있다. 이로 인해 환승 횟수가 많더라도 물리적인 이동 시간이 최소라면, 해당 경로를 가장 빠른 도착 경로로 판단한다. 하지만 대중교통 승객들은 과도한 환승을 선호하지 않는 경향이 있으므로, 환승 저항이 고려된 최단 경로를 탐색할 필요가 있다(Park et al., 2001; Arbex and da Cunha, 2015). 또한, 다양한 교통수단이 있는 대중교통 네트워크에서는 개인의 경로 선택의 폭을 넓혀 줄 수 있도록 복수의 대안 경로를 탐색할 필요도 있다(Kim and Kim, 2009).

이에 본 연구에서는 환승 저항을 고려한 운행시간표 기반의 대중교통 다중 경로 탐색 알고리즘을 개발하였다. 운행시간표 기반 대중교통 경로 탐색 알고리즘인 RAPTOR를 개선한 것이며, 환승 저항을 경로 탐색 과정에서 고려할 수 있도록 알고리즘을 수정하였다. 환승 저항은 환승 교통수단의 유형별로 각각 부여할 수 있게 하였고, 유형은 버스 간 환승, 버스-지하철 간 환승, 지하철 간 환승으로 구분하였다. 추가로 도보 환승 시의 환승 시간을 조절하기 위한 도보 보정 계수도 적용하였고, 이를 통해 도보 이동의 선호 정도를 경로 계산에 반영할 수 있다. 또한, 출발지에서 도착지까지 중복 없는 K개의 다중 경로를 산출할 수 있도록 개선하였고, 이 과정에서 유사 경로 판단 규칙을 적용하였다.

환승 저항, 도보 보정 계수, 다중 경로 산출 등의 추가 기능이 경로 탐색 결과에 어떤 영향을 주는지 확인하기 위해 기존 RAPTOR 알고리즘과 개선된 알고리즘의 경로 산출 결과를 비교하였다. 실험은 교통카드 실적자료로부터 추출한 실제 승객들의 이동 경로와 각 알고리즘의 산출 결과의 유사도를 분석하였다. 실제 승객들의 이동 경로는 총 이동 시간, 이동 거리, 환승 저항, 도보 이동 선호도, 대안 경로 선택 등의 여러 요소가 반영된 결과라 판단하였고, 실험을 통해 개선된 알고리즘은 기존 RAPTOR 알고리즘보다 승객의 다양한 경로 선택 기준을 반영할 수 있는 점을 확인할 수 있었다.

Ⅱ. 관련 연구

1. 관련 연구 분석

본 연구와 관련된 연구는 운행시간표 기반 대중교통 경로 탐색 연구와 다중 경로 탐색 관련 연구, 환승 저항 관련 연구가 있다. 우선, 최근에 개발되고 있는 대표적인 운행시간표 기반의 대중교통 경로 탐색 알고리즘으로는 RAPTOR(Round-bAsed Public Transit Optimized Router), CSA(Connection Scanning Algorithm), Trip-based

등이 있다(Delling et al., 2015; Dibbelt et al., 2013; Madkour et al., 2017). Delling et al.(2012)은 RAPTOR 알고리즘을 제시하였으며 각 정류장에 노선의 차량 별 도착 시각을 저장하고, 각 정류장마다 지나가는 차량을 탐색하여 최종 도착정류장까지의 최소 도착 시각 및 경로를 산출하였다. 이 연구에서는 운행시간표를 이용한 대중교통 경로 탐색 알고리즘이 그래프 이론 기반의 알고리즘보다 탐색 시간이 빠른 것을 실험을 통해 증명하였다. Dibbelt et al.(2013)은 CSA를 제시하였으며 운행정보를 1차원 배열의 형태로 나타낸 connection을 탐색하여 경로를 산출하였다. 이 연구에서 connection은 정류장에 도착하는 모든 교통수단의 도착 시각을 1차원의 배열로 나타낸 데이터 구조를 의미하며, 연구 결과는 RAPTOR의 연구 결과와 같이 그래프 이론 기반의 경로 탐색 알고리즘보다 경로 탐색 속도에 장점이 있음을 확인하였다. 이처럼 운행시간표를 이용하면 기존 그래프 이론 기반 탐색 방식보다 연산복잡도가 낮아, 매우 빠른 시간에 경로를 탐색할 수 있고, 출발 시각에 따른 동적인 탐색이 가능한 특징이 있다.

대중교통 다중 경로 탐색 방법과 관련된 대부분의 연구에서는 그래프 이론 기반의 알고리즘을 이용하여 다중 경로를 탐색한다. Guo and Jia(2017)은 그래프 이론 기반의 알고리즘을 이용하여 노드 대신 링크 형태의 arcs에 출발 시각을 부여하여 운행시간표를 고려한 다중 경로 탐색 방식을 제안하였다. 연구에서 제안한 다중 경로 탐색 방식은 각 arcs를 이용하여 노드에 도착한 시각 순서대로 저장하고 이를 도착 노드까지 반복하며 진행되었다. Hu and Chiu(2015)은 그래프 이론을 이용한 경로 탐색 과정에서 다중 경로 간에 일정부분의 중복을 허용하도록 경로 유사도 값을 결정하였고, 이를 이용한 다중 경로 산출 알고리즘을 제안하였다. 연구 결과, 적절한 경로 유사도 값을 적용하면 최단 경로의 이동시간과 K번째 경로의 이동시간의 차이가 크지 않고 비정상적인 경로가 산출되지 않는 것을 확인하였다.

마지막으로, 환승 저항과 관련된 최근 연구에 대한 분석이다. 환승 저항은 환승 대기 시간이나 환승 도보 이동 시간 등 시간적 요소와 환승 편의성, 혼잡도 등 환승으로 인한 심리적 부담감인 비시간적요소를 포함하는 개념으로, 비시간적요소를 시간으로 환산하여 알고리즘에 적용된다. Yoo(2015)는 스마트카드 데이터를 이용하여 서울시 대중교통을 이용할 때 발생하는 환승 저항값을 추정하였다. 그 결과, 평균적으로 11.24분의 환승 저항값이 발생한다고 추정하였고, 출·도착지가 어떤 곳인지에 따라 환승 저항값에 차이가 있다는 것을 확인하였다. Yoo(2017)는 평면거리, 계단 수, 환승시간, 에스컬레이터 유무 등을 환승 저항의 요소로 고려하여 수직이동거리를 평면거리로 환산하여 환승 편의성을 분석하였다. 연구 결과, 환승 저항은 수직 이동거리에 의해서도 발생하므로 환승 편의성은 평면거리와 수직 이동거리를 같이 고려한 개선방안을 제시하였다. Yang(2017)은 서울시 9호선과 공항철도의 환승개찰구를 이용한 승객들을 대상으로 1회 환승 시에 발생하는 환승 저항값을 계산하였다. 연구 결과, 지하철 간 환승 저항값은 환승 1회 당 5.35분 정도로 추정되었고 이를 통해 환승 횟수에 따라 환승 저항값이 달라진다고 언급하였다. Garcia-Martinez et al.(2018)은 통근자를 대상으로 도보 이동 시간, 대기 시간 등의 요인들과 환승 저항의 상관관계를 분석하였다. 연구 결과, 환승 저항은 환승 대기시간, 환승 도보시간 등 시간적 요소와 타 교통수단으로의 환승이나 혼잡도 등 비시간적요소에 의해 영향을 받는 것을 실험을 통해 증명하였다. 이처럼 환승 저항은 동일한 교통수단으로의 환승 시에도 적용되며, 서로 다른 교통수단으로의 환승 시에는 보다 높은 저항이 적용된다(Park et al., 2012). 이를 통해, 환승 저항의 적용은 경로 탐색에서 중요한 요인으로 작용하며, 환승 교통수단의 종류에 따라 각각의 값을 부여할 수 있도록 해야 한다는 점을 확인할 수 있다.

2. 기존 연구와의 차별성

관련 연구들을 살펴본 결과, 운행시간표 기반의 경로탐색 알고리즘은 연산시간 단축에 강점이 있고, 출발

시각에 따른 가변적 경로탐색이 가능하다. 또한, 기존 연구들은 K개의 다중 경로를 탐색하기보다 환승횟수, 최소도착시간 등을 고려한 파레토 셋을 찾는 연구들이 다수이다. 다만, 일반적으로 대중교통 승객은 경로를 선택하는 상황에서는 최소 도착 시간뿐만 아니라 그 외의 대안 경로들 중 개인의 선호(최소 환승, 최소 도보 이동 등)에 따라 경로를 결정하기 때문에 다중 경로 탐색이 필요하다. 따라서 본 연구에서는 다중 경로 탐색이 가능한 운행시간표 기반의 알고리즘을 제안하고자 한다. 이 과정에서 다중 경로 탐색과 관련된 선행 연구들에서 사용하는 경로 유사도 개념을 운행시간표 기반 알고리즘에 맞게 변형하여 적용하였다. 환승 저항의 경우는 운행시간표 기반 경로 탐색 과정에서 환승 대기 시간, 환승 도보 시간 등의 시간적 요소들이 이미 반영되고 있으므로, 비시간적 요소인 심리적 저항감만 환승 저항으로 고려하였고 이를 시간으로 환산하여 경로 탐색이 가능하도록 하는 방법을 제안한다.

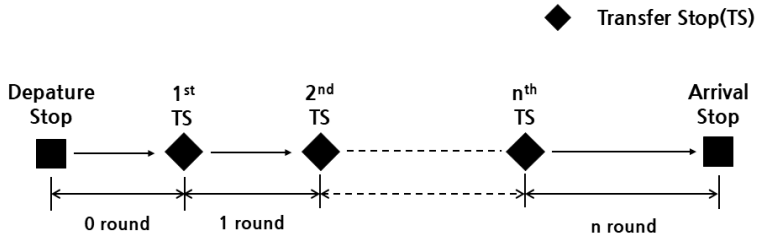
3. RAPTOR 알고리즘

본 연구에서 제안하는 알고리즘은 RAPTOR를 개선한 것이므로, 기존 RAPTOR에 대한 간략한 설명을 진행한다. 설명에 사용된 주요 변수들은 <Table 1>에 정리되어 있고 알고리즘의 데이터 구조와 이들 간 관계는 <Fig. 2>와 같다.

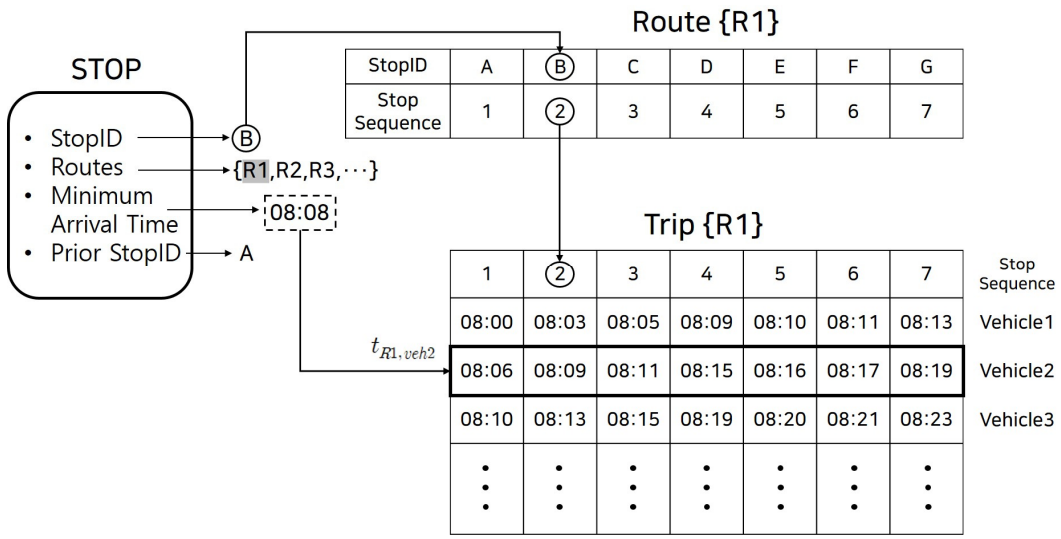
〈Table 1〉 Nomenclature

Variable	Description
$J_{s,e}$	Journey from stop $s \in S$ to stop $e \in S$
$t_{r,v}$	Driving information of vehicle v for the route $r \in R$
$\tau_{dep}^{m}(p)$	Minimum arrival time at stop $p \in S$ in m rounds
$\tau_{arr}^{m}(t,p)$	Arrival time at stop $p \in S$ using $t_{r,v} \in T$ in m rounds

RAPTOR는 기본적으로 라운드 별로 경로 탐색 과정을 진행하는데, 하나의 라운드는 환승 정류장(또는 0라운드에서는 출발지)에서 다음 환승 정류장(또는 도착지)까지의 최소 도착 시각을 갱신하는 하나의 단계를 의미한다. m라운드에서는 m번째 환승 정류장에서 m+1번째 환승 정류장까지 이동할 수 있는 모든 정류장의 최소 도착 시각을 갱신한다. 알고리즘의 시작은 0라운드부터 시작하며, <Fig. 1>를 이용하여 라운드에 대한 설명을 추가적으로 서술한다. 먼저 출발 정류장에서 교통수단 또는 도보를 이용하여 도착하는 환승 정류장(그림에서는 1st TS)들을 탐색하는 과정이 0라운드이고, 1라운드에서는 환승 정류장(1st TS)에서 교통수단 또는 도보로 이동할 수 있는 환승 정류장(그림에서는 2nd TS)를 탐색한다. 이렇게 이전 라운드의 환승 정류장에서부터 교통수단 또는 도보로 이동할 수 있는 환승 정류장(또는 도착지)을 탐색하는 과정을 라운드라고 한다.



〈Fig. 1〉 An example of rounds



〈Fig. 2〉 An example of data structure in RAPTOR algorithm

알고리즘의 데이터 구조는 몇 개의 객체들로 구성되어 있다. 이들은 정류장 정보와 m라운드에 정류장 p의 최소 도착 시각인 $\tau^m_{dep}(p)$을 저장하는 Stop S, 노선 정보를 저장하는 Route R, 노선 별 차량의 운행정보를 저장하는 Trip T, 정류장간의 도보 이동 정보를 저장하는 F을 포함한다. 하나의 Route는 어떤 노선이 지나가는 정류장 번호와 정류장 순번으로 구성된다. 하나의 Trip은 노선의 차량이 출발차고지에서 도착차고지까지 한 번의 운행을 의미하며, 정류장순번 1번부터 마지막 순번까지의 정류장별 도착 시각들을 저장한다.

$t_{r,v}$을 탐색하는 방식은 <Fig. 2>의 예시를 이용하여 설명하겠다. <Fig. 2>에서 승객이 Stop 'B' 정류장에 8:08분에 도착했다고 가정하고 이용할 수 있는 노선{R1,R2,R3}을 모두 이용하여 다른 정류장으로 이동해본다. <Fig. 2>에서는 그 중 R1노선을 이용한 경우이고 R1의 route 정보에서 Stop 'B'에 해당하는 정류장 순번 '2'을 찾고, 이와 동일한 순번을 R1의 Trip 테이블에서 찾는다. Stop 'B'에 도착한 시간인 8:08분 이후에 탑승할 수 있는 차량 중 가장 빨리 탑승할 수 있는 8:09분에 도착하는 차량($t_{R1,veh2}$)을 선택한다. 이후, 선택한 $t_{R1,veh2}$을 탑승하여 지나는 모든 정류장까지 이동하면서 최소 도착 시각을 갱신한다. 최소 도착 시각을 갱신하는 것은 정류장의 $\tau^m_{dep}(p)$보다 $\tau^m_{arr}(t,p)$이 더 빠를 경우 $\tau^m_{dep}(p)$을 변경하는 것을 의미한다. 정류장의 $\tau^m_{dep}(p)$이 갱신되면 해당 정류장을 마킹한다. 마킹된 모든 정류장은 다음 라운드에서 $t_{r,v}$을 탐색하거나 도보로 이동할 수 있는 정류장을 탐색하기 위해 사용된다.

RAPTOR 알고리즘은 입력 값으로 출발 정류장, 도착 정류장, 출발 시각을 이용한다. 경로를 탐색하기 전에 모든 정류장의 최소 도착 시각은 ∞로 초기화하고 출발 정류장의 최소 도착 시각만 출발 시각을 입력한다. 알고리즘의 수행 단계는 크게 3단계로 구분하여 진행되며 0라운드부터 시작한다.

STEP 1. 0라운드가 아닌 경우에는 모든 정류장의 m라운드 최소 도착 시각에 (m-1)라운드까지의 최소 도착 시각을 저장한다. 또한, (m-1)라운드에 마킹된 정류장에서 이용할 수 있는 모든 노선을 노선-정류장 (R,S) 쌍으로 Q라는 이름의 리스트에 저장한다. Q는 탐색을 위해 사용하는 변수이며, 마킹한 정류장에서 출발 가능한 노선을 묶음으로 저장한 리스트이다. 0라운드인 경우, 출발 정류장에서 이용할 수 있는 노선을 Q에 저장하고 STEP2를 진행한다.

STEP 2. Q에 저장된 정류장-노선 쌍을 하나씩 이용하면서 $t_{r,v}$을 탐색한다. 정류장의 최소 도착 시각 이후

에 탑승할 수 있는 노선의 $t_{r,v}$을 탐색하고 이를 이용하여 지나는 모든 정류장(p')으로 이동한다. $\tau^m_{arr}(t,p)$와 p'의 $\tau^m_{dep}(p')$을 비교하고, $\tau^m_{arr}(t,p)$이 빠를 경우 p'의 $\tau^m_{dep}(p')$을 갱신하고 해당 정류장을 마킹한다.

STEP 3. STEP 2에서 마킹된 정류장에서 도보로 이동할 수 있는 정류장(p_i)까지 이동한다. 마킹된 정류장의 $\tau^m_{dep}(p)$에 도보 이동시각을 더한 시각과 p_i의 $\tau^m_{dep}(p_i)$을 비교하고, 도보로 이동하여 도착한 시각이 빠를 경우 p_i의 $\tau^m_{dep}(p_i)$을 갱신하고 해당 정류장을 마킹한다.

STEP 4. $\tau^m_{dep}(p)$가 갱신된 정류장이 있는지 확인한다. 최소 도착 시각이 갱신된 정류장이 있을 경우 m을 1 증가시키고 STEP 1~3 과정을 반복한다. 최소 도착 시각이 갱신된 정류장이 없다면 알고리즘을 종료한다.

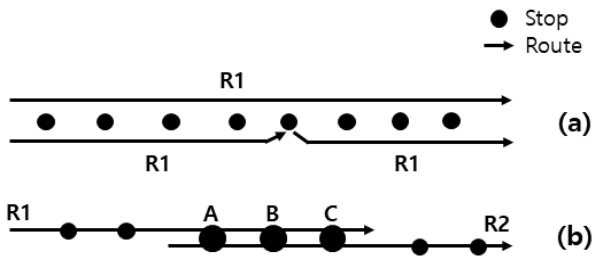
RAPTOR 알고리즘은 도착 정류장에 $\tau^m_{dep}(p)$만 저장되므로 도착 정류장까지 이동한 경로를 알 수 없다. 따라서 $\tau^m_{dep}(p)$가 갱신될 때 $t_{r,v}$을 이용하기 시작한 정류장 정보를 함께 저장한다. 이를 도착 정류장부터 정류장을 역으로 탐색하여 출발 정류장까지 반복하여 경로를 탐색한다. 예를 들어, <Fig. 2>의 'B' 정류장에 $\tau^m_{dep}(p)$을 갱신한 Prior Stop 'A'을 찾고, Stop 'A' 정류장의 $\tau^m_{dep}(p)$을 갱신한 Prior Stop을 찾는 과정을 출발 정류장까지 반복한다.

Ⅲ. 개선된 RAPTOR 알고리즘

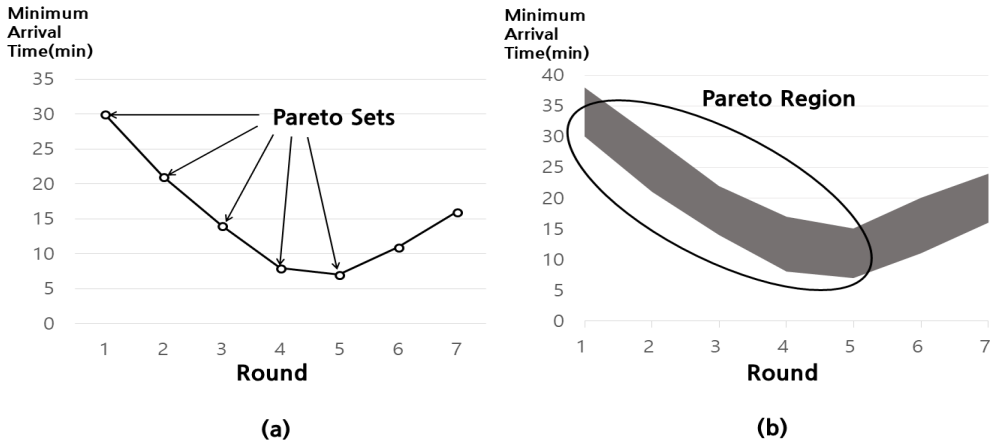
1. 유사 경로를 제외한 다중 경로 탐색

기존 RAPTOR 알고리즘은 출발지에서 도착지까지 가장 빨리 도착한 경로를 도착 정류장부터 역으로 탐색하여 산출한다. 본 연구에서 제안하는 알고리즘은 모든 정류장까지 도착하는 경로를 도착 시각 순서에 따라 K개까지 정류장에 저장한다. 이 때, K개의 경로에는 유사한 경로가 중복으로 저장되지 않도록 하였다. 다중 경로 탐색에서 탐색한 경로의 중복여부는 중요한 요소이다. 본 연구에서는 동일한 노선의 조합으로 구성된 경로 또는 같은 노선을 재탑승(지하철 제외)하는 경로는 유사한 경로로 분류하였고, 이러한 경로가 산출되지 않도록 두 가지 규칙을 적용하였다.

첫 번째 규칙은 특정 노선을 타고 하차한 정류장에서 같은 노선의 $t_{r,v}$을 탐색하지 않는 것이다. <Fig. 3>의 (a)는 왼쪽 정류장에서 R_1을 탑승하여 오른쪽 정류장으로 이동하는 상황이다. R_1에 탑승하여 한 번에 도착 정류장으로 이동하는 경로(그림 상단)와 R_1에 탑승한 후, 중간 정류장에서 하차한 후에 같은 정류장에서 R_1에 재탑승하는 경로(그림 하단)는 도착 정류장까지의 도착 시각이 다르므로 서로 다른 경로로 저장된다. 하지만, 이와 같은 경로는 실질적으로 같은 경로로 볼 수 있으므로 중복하여 저장하지 않도록 하차한 정류장에서 이전에 이용한 노선으로 다시 탑승하지 않도록 하였다.



〈Fig. 3〉 An example of similar paths



〈Fig. 4〉 Comparison of Pareto sets between RAPTOR (a) and our improved RAPTOR (b)

두 번째 규칙은 $J_{s,e}$에 탑승한 노선들의 순서를 저장하고 이후 동일한 노선 순서를 이용한 경로는 $J_{s,e}$에 추가하지 않고 최소 도착 시각만 비교하여 갱신하는 것이다. <Fig. 3>의 (b)는 R_1노선에서 R_2노선으로 환승할 때 다른 환승 정류장 A,B,C에서 환승하는 상황을 나타낸다. A,B,C 중 어디서 환승하는지에 따라 이동 소요 시간은 약간 달라질 수 있지만, 이용한 노선은 일치하는 상황이다. 이러한 경로는 유사한 경로로 판단하였고, 노선 순서가 $\mathrm{R}_1 \rightarrow \mathrm{R}_2$인 경로가 $J_{s,e}$에 이미 저장되어 있다면 저장된 경로와 현재 탐색한 경로의 도착 시각을 비교하여 빠른 도착 시각을 $J_{s,e}$에 갱신하여 동일한 노선 순서를 이용한 경로를 중복저장 하지 않는다.

<Fig. 4>은 RAPTOR 알고리즘과 본 연구에서 제시하는 개선된 RAPTOR 알고리즘의 Pareto-optimal set을 나타낸 그래프이다. RAPTOR는 도착 시각과 환승횟수를 최소화하는 다목적(bicriteria) 최적화 알고리즘으로, 최소 도착 시각이 산출되면서 환승횟수가 최소인 사전편찬식(lexicographic) 최적해를 탐색할 수 있다. 개선된 RAPTOR 알고리즘도 기존 RAPTOR와 동일한 방식으로 Pareto-optimal set을 탐색한다. 다만, 각 라운드 별로 도착 시각 순서에 따라 $n(\leq K)$개의 다중 경로를 탐색하기 때문에, <Fig. 4>과 같이 해 공간이 영역으로 나타난다.

2. 환승 저항 적용 및 도보 이동 시간 조정

RAPTOR에서는 경로 탐색 시 각 정류장의 도착 시각 이후에 탑승 가능한 $t_{r,v}$을 찾은 후, 해당 차량을 타고 다른 정류장에 내리며 하차 정류장의 도착 시각을 갱신한다. 따라서 각 정류장에 도착하는 시각이 변경되면 탑승하는 노선과 차량이 달라질 수 있고, 이로 인해 산출되는 경로가 동적으로 바뀌는 특징이 있다. 본 연구에서는 이러한 특징을 이용하여 환승하는 시점에 환승 저항을 반영한 동적인 경로 선택이 가능하도록 하였다. 정류장에서 환승을 하는 경우, 환승 저항 시간만큼 대기한 후 이후에 탑승 가능한 $t_{r,v}$을 탐색하도록 하였다. 예를 들어, 한 정류장의 최소 도착 시각이 15시 3분이고 환승 저항이 3분이라면 15시 6분 이후의 $t_{r,v}$을 검색한다. 또한, 환승 저항에는 환승 횟수 외에도 환승을 하는 수단까지의 수평적 이동과 수직적 이동이 고려되어야 한다. 버스에서 버스로 환승하는 경우에는 대부분 수평적 이동이 많은 반면, 버스에서 지하철로 환승하거나 지하철 호선 간 환승할 때 수직적 이동이 수반되므로, 환승 유형에 따라 환승 저항이 다르게 적용될 것이라고 가정한다. 따라서 환승 유형은 버스 간 환승, 버스-지하철 간 환승, 지하철 간 환승으로 구분

하였고 환승 유형에 따라 다른 환승 저항값을 적용하였다. 이를 구현하기 위해 정류장에 도착한 $t_{r,v}$의 교통수단의 종류를 저장해두고 다음에 탑승하는 $t_{r,v}$의 교통수단에 따라 환승 저항값을 다르게 부여할 수 있도록 하였다.

다음은 도보 보정 계수의 적용에 대한 설명이다. 도보 속도는 성인의 걸음걸이를 기준으로 약 $1.2m/s$ 이다. 본 연구에서는 정류장 간 도보 이동이 가능하다면, 이동 거리를 직선거리로 계산하여 사용한다. 따라서 이동 거리와 보속을 통해 도보 이동 소요 시간을 계산하고, 도보로 이동한 정류장의 도착 시각을 갱신한다. 다만, 이동 거리를 직선거리로 계산하므로 실제 보행자의 도보 이동 거리보다 짧게 계산되고, 도보 상황에 따라 횡단보도에 의한 지체도 발생할 수 있다. 이로 인해 보속을 $1.2m/s$로 지정하면 이동 소요 시간이 실제 상황보다 짧아지게 되므로 도보 이동 시간의 조정이 필요하다. 이에 본 연구에서는 실제 보행자의 환승 도보 이동 시간과 알고리즘 내의 환승 도보 이동 시간이 유사하도록 도보 보정 계수를 이용하며, 이는 개념적으로 도보 이동 속도에 해당하나 실제 보속보다는 느리게 부여한다. 이를 위해 실제 대중교통 승객이 직선거리가 아니라 출발지부터 도착지까지 직각으로 이동하는 맨해튼 거리(Manhattan distance)로 이동하는 것으로 가정한다. 만약 직선거리가 1일 때 맨해튼 거리는 $\sqrt{2}$가 되므로, 거리의 비율에 따라 속도를 계산한 결과 $1.2m/s$를 $\sqrt{2}$로 나누어 준 값인 $0.83m/s$을 보정 계수로 이용한다.

3. 개선된 RAPTOR 알고리즘을 이용한 경로 탐색 과정

개선된 RAPTOR 알고리즘의 전체적인 경로 탐색 과정은 다음과 같다. 탐색을 시작하기 전에 출발 정류장의 최소 도착 시각은 출발 시각으로 설정하고, 다른 정류장들의 최소 도착 시각은 ∞로 초기화한다. 또한, 출발 정류장을 마킹하고 0라운드부터 탐색을 시작한다. 이외에 알고리즘의 설명을 위해 m라운드에 각 정류장에 도착한 경로를 도착 시각이 빠른 순으로 정렬하여 $\{J^{m,1}, J^{m,2}, \cdots, J^{m,K}\} = J^m$으로 표기하였다. 개선된 RAPTOR의 알고리즘을 이해하기 쉽도록 기존 RAPTOR의 알고리즘과 함께 <Fig. 5>에 정리하였다.

STEP 1. 0라운드가 아닌 경우에는 각 정류장의 m라운드 최소 도착 시각과 J^m에 $(m-1)$라운드의 최소 도착 시각과 J^{m-1}을 저장한다.

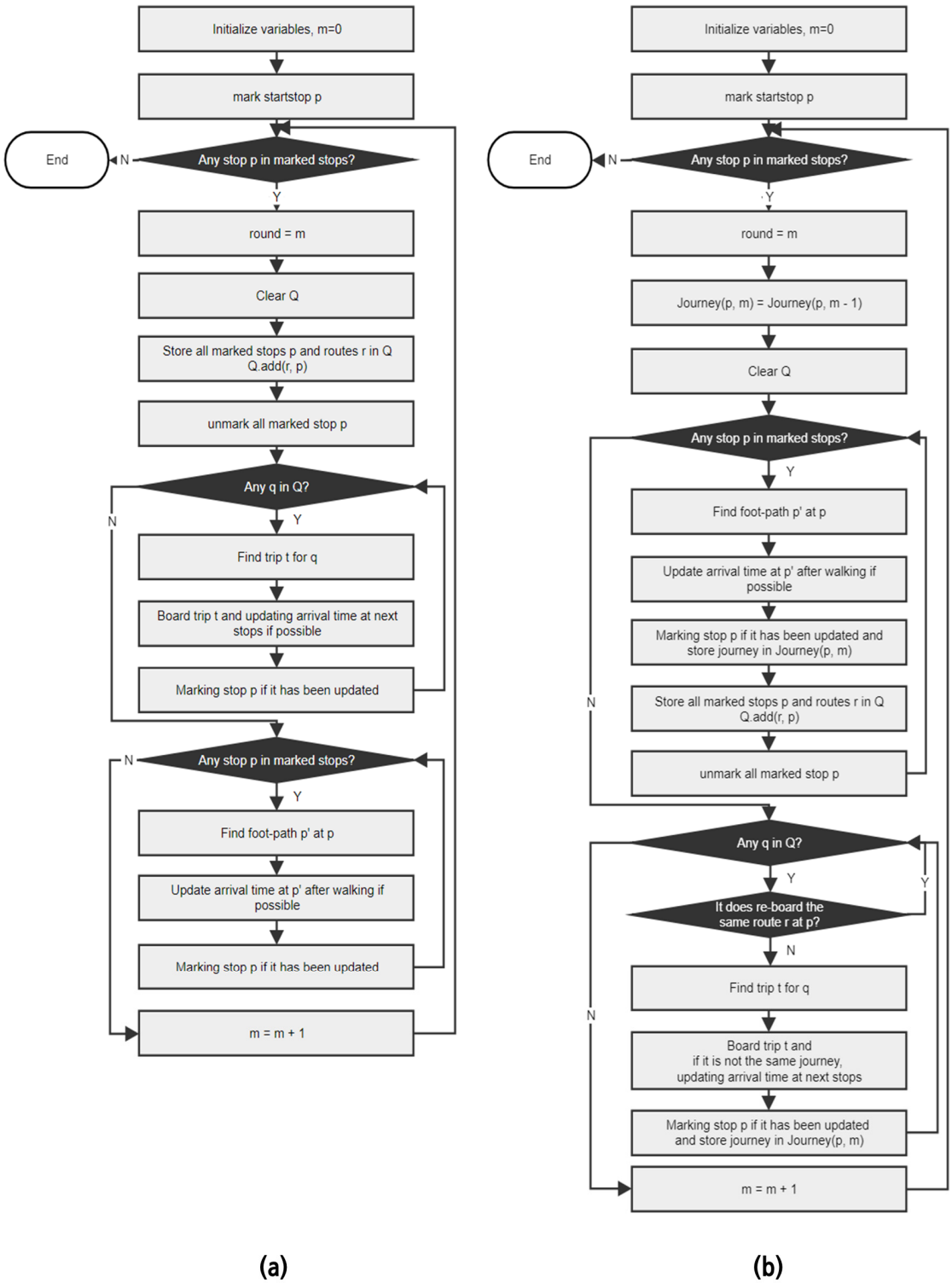
STEP 2. $(m-1)$라운드에 마킹한 정류장에서 도보로 이동할 수 있는 정류장들의 $J_{s,e}$와 $\tau^m_{dep}(p)$을 탐색하고, $\tau^m_{dep}(p)$보다 $J_{s,e}$의 도착 시각이 빠르면 정류장의 $\tau^m_{dep}(p)$을 갱신한 후 마킹한다. $n(J^m) < K$ 인 경우에는 $J_{s,e}$을 J^m에 추가한다. $n(J^m) = K$인 경우에는 $J^{m,K}$을 $J_{s,e}$로 교체한다. J^m에 변화가 있다면, J^m을 도착 시각이 빠른 순서대로 정렬한다.

STEP 3. $(m-1)$라운드에 마킹한 정류장 S에서 이용할 수 있는 모든 노선 R을 (R,S) 형태로 Q에 저장한다.

STEP 4. Q에 저장된 모든 (R,S)의 S에서 R의 $t_{r,v}$을 탐색한다. 이 때, 정류장의 $\tau^m_{dep}(p)$부터 환승 저항 시간만큼 대기한 후 $t_{r,v}$을 탐색한다.

STEP 5. S의 J^m에서 마지막에 탑승한 노선이 R과 동일한 경우 STEP 6, 7을 생략한다.

STEP 6. R의 $t_{r,v}$을 찾고, 이를 이용하여 지나는 모든 정류장에 하차한다. 하차한 정류장에서 $J_{s,e}$의 노선 순서와 동일한 노선 순서를 가지는 경로가 J^m에 없는 경우에는 다음 과정을 진행하고, 이외의 경우는 STEP7을 진행한다. $n(J^m) < K$인 경우에는 J^m에 $J_{s,e}$을 추가한다. $n(J^m) = K$인 경우에는 $J^{m,K}$의 $\tau^m_{dep}(p)$와 $J_{s,e}$의 $\tau^m_{arr}(t,p)$을 비교하고, 갱신이 가능하면 $J^{m,K}$을 $J_{s,e}$로 교체한다. J^m에 변화가 있다면, J^m을 도착 시각이 빠른 순서대로 정렬한다.



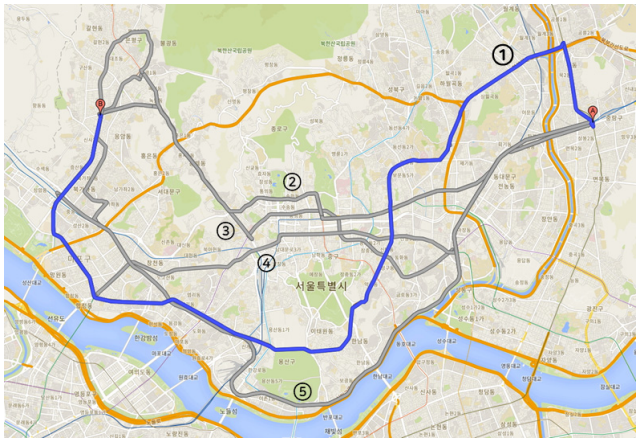
〈Fig. 5〉 Comparison of flowcharts of RAPTOR(a) and our improved RAPTOR(b)

STEP 7. 하차한 정류장의 J^m 중 $J_{s,e}$와 동일한 노선 순서를 가지는 경로를 찾고, 해당 경로의 $\tau_{dep}^m(p)$와 $J_{s,e}$의 $\tau_{arr}^m(t,p)$을 비교하여 갱신한다. 갱신이 일어나면 J^m을 도착 시각이 빠른 순서대로 정렬한다.

STEP 8. $\tau_{dep}^m(p)$이 갱신된 정류장이 있으면 STEP 1부터 라운드를 증가시켜 반복한다. $\tau_{dep}^m(p)$이 갱신된 정류장이 없으면 탐색을 종료한다.

4. 개선된 알고리즘 구현 예시

<Fig. 6>은 개선된 알고리즘을 구현한 화면이다. 개발된 알고리즘은 Intel Xeon E5-2667 2.90GHz와 DDR3-1600 RAM 64GB의 하드웨어 환경에서 구동하였다. 무작위로 선택된 O-D에 대해 알고리즘을 이용하여 10개의 경로를 산출하면, 출발 정류장→도착 정류장은 평균 3초, 출발 정류장→모든 정류장은 평균 7초가 소요되었다. K가 1인 경우에는 두 상황 모두 1초 이내로 계산이 되며, K를 크게 설정할수록 비교할 항목이 늘어나 연산 시간이 증가하게 된다. 출발 정류장은 상봉역, 도착 정류장은 응암역을 선택하였으며 출발 시각은 오전 8시 30분을 입력하였다. 도보 보정 계수는 $0.83m/s$, 환승 저항은 모든 환승 유형에 5분을 설정하였다. <Fig. 6>의 ①번은 기존 RAPTOR 알고리즘에서 찾은 최단 도착 시각 경로를 나타낸 모습이고 ②~⑤번은 개선된 알고리즘을 이용한 다중 경로의 모습이다. ②~⑤번은 실제로는 10개의 경로를 탐색한 결과인데 일부 경로의 노선이 같은 도로나 철로를 이용하는 경우가 있어서 지도에는 5개의 경로로 표시되었다. 결과를 보면, K의 개수에 따라 출발지에서 도착지까지 이동 가능한 경로가 여러 가지로 계산되는 것을 확인할 수 있다.



〈Fig. 6〉 An example of computed paths using the improved RAPTOR

Ⅳ. 실험 및 결과 분석

1. 실험 개요

환승 저항의 적용 및 다중 경로의 산출이 경로 탐색에 어떤 영향을 주는지 확인하기 위해, 기존 RAPTOR 알고리즘에 환승 저항을 적용하여 산출한 경로가 실제 승객의 이동경로와 유사한지 확인하는 실험을 진행하였다. 또한, 환승 유형별로 환승 저항을 다르게 부여할 때의 산출 경로 변화를 분석하였다.

실험에는 2017년 10월 12일부터 2017년 10월 19일까지 대중교통을 이용한 승객의 스마트카드 기록을 이

용하였으며, 10월 19일의 버스 운행정보를 이용하여 구축한 운행시간표를 사용하였다. 정류장간의 도보 환승은 정류장 간의 직선거리가 700m 이내일 경우 가능하도록 하였고, 연속적인 도보 환승이 가능하도록 하였다. 실험에 사용될 실적 경로는 일주일동안 승객 40명 이상이 이용한 경로를 선정하였다. 이들 중 환승 횟수가 0회인 경로 500개, 1회인 경로 500개를 무작위로 추출하여 1000개의 경로 데이터 셋을 결정하였다.

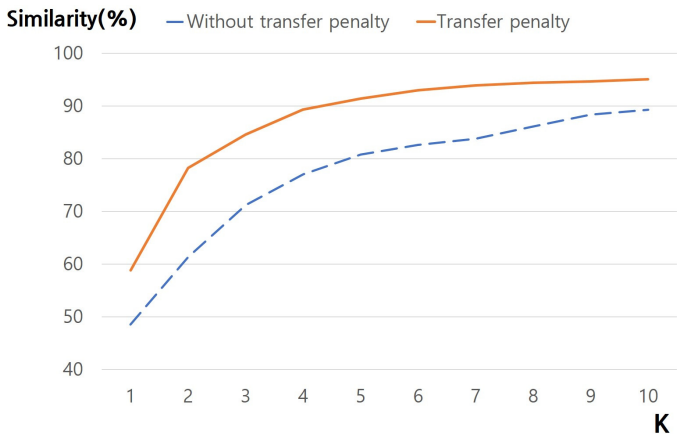
유사도는 데이터 셋의 경로 1000개 중 알고리즘 산출 경로와 일치하는 개수의 비율을 의미한다. 두 경로의 일치 여부를 확인할 때, 버스를 이용한 통행과 지하철을 이용한 통행을 구분하였다. 버스를 이용한 통행의 경우에는 알고리즘의 산출 경로와 실적 경로의 하차 정류장과 탑승 정류장이 다르더라도 동일한 노선을 이용하면 경로가 일치하는 것으로 가정하였다. 지하철을 이용한 통행은 스마트 카드에 지하철의 탑승 정류장과 도착 정류장 정보만 기록이 되므로 어떤 경로로 이동했는지, 어떤 정류장에서 환승했는지 알 수 없다. 따라서 알고리즘의 산출 경로와 실적 경로에서 승차역과 하차역이 동일하다면 두 경로가 일치하는 것으로 가정하였다. 전술한 바와 같이 가정한 상황에서 알고리즘의 산출 경로와 실적 경로 데이터 셋의 경로를 비교하여 유사도를 계산하였다.

알고리즘의 산출 경로와 실적 경로의 유사도를 비교하기 위해 실적 경로 1000개의 출발·도착 정류장 위치와 출발 정류장에서 탑승한 시각을 알고리즘의 입력 값으로 사용하였다. 탑승 시각은 실제 대중교통 승객이 버스의 경우 승·하차한 태그 시각, 지하철의 경우 개찰구에 태그한 시각을 의미한다. 알고리즘의 입력 값인 출발 시각은 실제 대중교통 승객이 탑승 시각 이전에 버스 정류장 또는 지하철역에 도착한 시각을 의미한다. 운행시간표 정보는 대중교통 노선을 운행할 계획에 대한 정보로 날짜마다 운행계획이 다르고, 실제 버스 및 지하철이 운행되면서 발생하는 변수(교통 상황, 사고 등)로 인해 운행 정보가 계획과 달라 질 수 있다. 이러한 운행정보의 변동을 고려하여 노선의 차량이 정류장에 운행계획에 맞추어 도착할 수도 있지만, 계획된 시간보다 5분 전, 또는 5분 후에 차량이 도착할 수 있다고 가정하였다. 알고리즘에서 사용한 운행시간표는 고정된 값을 이용하므로 대신 승객의 출발 시각을 조정하여 운행정보의 변동을 고려하기로 한다. 따라서 탑승 시각을 기준으로 5개의 시간 범위(탑승 시각 10분 전, 5분전, 탑승 시각, 5분후, 10분 후)를 출발 시각 후보를 선정하였다. 5개의 출발 시각 후보를 각각 알고리즘의 출발 시각으로 설정하여 1000개의 경로 정보에 대한 알고리즘 산출 경로를 탐색하였다. 이후, 모든 시각별 경로 탐색이 완료되면 산출된 전체 경로를 취합하여 소요 시간이 짧은 순서대로 상위 K개의 경로를 선택하였다.

2. 실험 결과

1) 환승 저항의 적용 여부에 따른 유사도 비교 실험

<Fig. 7>은 개선된 알고리즘의 환승 저항의 적용 여부에 따른 경로 유사도 차이를 비교한 것이다. <Fig. 6>의 X축은 K의 개수, Y축은 유사도를 나타낸다. 이 실험에서는 환승 유형에 관계없이 환승 저항에 동일한 값을 부여하였다. 점선은 도보 보정 계수는 1.2 m/s, 환승 저항은 0분을 설정한 것이고, 실선은 도보 보정 계수는 1.2m/s, 환승 저항은 3분을 설정한 것이다. <Fig. 7>에서 단일 경로(K=1)일 때, 환승 저항을 적용하지 않은 경우는 유사도가 48.5%인 반면 환승 저항을 적용한 경우는 유사도가 58.8%로 약 10%p 정도 증가하였다. 다중 경로(K=5)를 산출한 경우, 환승 저항을 적용하지 않은 경우는 유사도가 80.8%였고, 환승 저항을 적용한 경우는 유사도가 91.4%를 나타내어 두 경우 모두 유사도의 큰 차이를 보였다. 이처럼 K의 개수에 상관없이 환승 저항을 적용한 경우가 적용하지 않은 경우의 유사도보다 높은 것을 확인 할 수 있었다. 또한, 이는 기존의 RAPTOR 알고리즘으로 찾지 못한 경로 100여개를 환승 저항을 고려하였을 때 탐색이 가능해졌다는 의미이다. 이를 통해, 실제 승객이 환승 시 환승 저항을 고려한 경로를 선택하는 경우가 존재한다고 추정



〈Fig. 7〉 Comparison of similarities between without transfer penalty and transfer penalty

할 수 있다. 다만, K를 10까지 증가시켰음에도 불구하고 유사도가 100%가 되지 않는 것으로 보아 개선된 알고리즘에서도 탐색되지 않는 승객들의 이동 경로도 존재함을 알 수 있다. 즉, 환승 저항 이외에도 다른 경로 선택 조건이 추가적으로 고려되어야 한다고 판단된다.

2) 환승 유형 별 환승 저항에 따른 경로 산출 결과 비교 실험

도보 보정 계수와 환승 유형별 환승 저항의 영향을 알아보기 위해 6가지 경우로 나눠 각 파라미터 값을 미리 설정해두었고, <Table 2>에 정리되어 있다. 상황 A는 환승 저항을 고려하지 않은 경우, 상황 B는 환승 유형에 상관없이 동일한 환승 저항을 적용한 경우, 상황 C는 버스-지하철 간의 환승을 선호하지 않는 경우, 상황 D는 지하철 간의 환승을 선호하지 않는 경우, 상황 E는 도보 이동을 선호하지 않는 경우, 상황 F는 모든 환승을 선호하지 않는 경우를 의미한다. <Table 3>은 출발 시각을 오전 8시 30분으로 설정하고 $K=3$일 때, 상황별로 서울시립대에서 출발하여 강남역에 도착한 경로를 나타낸 것이다. 총 이동 시간은 차내 이동 시간, 환승 시간, 도보 이동 시간, 환승 저항이 모두 고려된 값이다.

상황 A와 상황 B를 비교해보면, 상황 A의 경우 <121 → 성동02 → 분당선 → 2호선>과 같이 총 이동 시간이 41분으로 짧지만, 환승 횟수가 3회인 경로가 산출된다. 상황 B는 환승에 대한 저항이 적용되어 상황 A와 달리 성동02 버스로의 환승을 하지 않고 바로 분당선 지하철로 환승하는 환승 2회 경로가 최소 이동 시간 경로로 산출되는 것을 확인할 수 있다. 이 과정에서 환승 저항에 따른 추가 대기 시간이 소요되면서 총 이동 시간은 54분으로 증가하였다.

〈Table 2〉 walking adjustment factor and transfer penalties for different cases

case	walking adjustment factor(m/s)	bus-bus (min)	bus-subway (min)	subway- subway (min)
A	0.83	0	0	0
B	0.83	5	5	5
C	0.83	5	15	5
D	0.83	5	5	15
E	0.2	5	5	5
F	0.83	15	15	15

〈Table 3〉 Examples of paths found of O-D(Nambuterminal Stn.→Isu Stn.)

case	calculated path	total journey time	walking time
A	121 → Sungdong 02 → Bundang line → Line 2	41m	7m 35s
	420 → 9711A → 421	42m 55s	8m 13s
	420 → Gyeongui·Jungang line → 9711A → 145	43m 29s	8m 13s
B	121 → Bundang line → Line 2	54m	4m 20s
	420 → Gyeongui·Jungang line → 421	56m 55s	8m 34s
	420	1h 52s	3m 22s
C	420	1h 52s	3m 22s
	420 → 145	1h 3m 29s	41s
	1227 → 145	1h 3m 29s	1m 38s
D	420 → Gyeongui·Jungang line → 421	56m 55s	8m 34s
	420	1h 52s	3m 22s
	121 → Bundang line → 360	1h 1m 16s	11m
E	420 → 145	1h 5m 38s	2m 50s
	1227 → 145	1h 5m 38s	6m 45s
	420	1h 11m	14m
F	420	1h 52s	3m 22s
	420 → 145	1h 13m 29s	41s
	1227 → Line2	1h 13m 29s	1m 38s

상황 B와 상황 C를 비교하면, 버스-지하철 간 환승 저항을 높게 설정한 경우인 상황 C에서 420번 버스만 이용한 경로의 이동 시간은 환승 저항의 영향을 받지 않으므로 상황 B에서 3번째 경로의 이동 시간과 동일하다. 대신 상황 B의 <420 → 경의중앙선 → 421>과 같이 버스-지하철 간 환승이 있는 경로는 환승 저항값이 크므로 상황 C에서는 탐색되지 않는다.

상황 B와 상황 D를 비교하면, 지하철 호선 간 환승 저항을 높게 설정할 경우 <121 → 분당선 → 2호선>의 경로처럼 지하철 노선을 변경하는 경로는 탐색되지 않는다. 대신 <420 → 경의중앙선 → 421>과 같이 지하철 간 환승 없이 이용하는 경우는 이동 시간이 56분 55초로 동일한 것을 확인할 수 있다.

상황 B와 상황 E를 비교하면, 420번 버스를 이용하여 3분 22초가 걸리던 도보 이동 시간이 14분으로 증가하게 되어 총 이동 시간이 1시간 11분으로 증가하였다. 이로 인해 <420 → 145>와 같이 도보 이동 시간이 2분 50초인 경로가 총 이동 시간 1시간 5분 38초로 계산되어 최소 도착 시각 경로로 검색되었다.

마지막으로 상황 B와 상황 F를 비교하면, 420번 버스를 이용한 경우 상황 B와 상황 F에서 동일하게 1시간 0분 52초가 소요되었다. 하지만 상황 F에서는 모든 환승에 대한 저항이 높아 환승 없이 이동하는 경로가 선호되므로, 상황 B와 상황 F에서 총 이동 시간의 값과 경로의 순서가 달라진 것을 알 수 있다.

실험 결과를 종합적으로 살펴보면 환승 유형에 따라 다른 저항값을 부여하면 경로를 탐색할 때 설정한 저항값에 따라 환승 시 교통수단을 다르게 선택하게 되고, 이로 인해 전체 경로에 차이가 발생하는 것을 확인할 수 있었다. 또한, 환승 저항의 적용 여부뿐만 아니라 값의 크기에 따라서도 서로 다른 경로가 산출되는 것을 확인하였다.

Ⅴ. 결 론

본 연구에서는 운행시간표 기반 RAPTOR 알고리즘에 환승 저항과 다중 경로 탐색 기법을 적용한 개선된 알고리즘을 제안하였다. 중복 없는 K개의 다중 경로를 산출하기 위해 유사 경로 판단 규칙을 설정하였고, 환승을 하는 과정에서 환승 저항을 고려할 방법을 알고리즘에 적용하였다. 환승 저항은 환승 유형에 따라 다르게 설정할 수 있도록 하였고, 도보 이동 시간을 조정하기 위한 도보 조정 계수도 추가하였다. 제안한 알고리즘에 따른 경로 탐색 결과를 확인하기 위한 프로그램을 구현하였고, 제안한 방법이 결과에 미치는 영향을 확인하기 위한 실험을 진행하였다. 실제 승객의 이동 경로와 경로 탐색 결과의 유사도를 비교하는 실험을 수행하였고, 제안한 방법을 이용하면 기존의 RAPTOR 알고리즘으로 산출한 경로보다 실제 승객의 이동 경로와 유사한 경로를 더 많이 산출할 수 있음을 확인하였다. 또한, 환승 저항값을 다르게 부여할 때마다 다양한 경로가 산출되는 것을 확인할 수 있었다.

본 연구에서 적용한 환승 저항과 도보 이동 속도는 수직적 이동과 수평적 이동에 대한 각각의 값을 지정할 수 없는 한계점이 있다. 수직적 이동에 따른 환승 저항을 일부 고려하기 위해 환승 유형별로 구분하여 각각의 환승 저항값을 부여할 수 있도록 하였고, 도보 이동 시간도 값의 조정은 가능하지만 수직, 수평 이동에 대한 세밀한 조정은 불가능하며, 후속 연구에서 이를 보완할 계획이다.

또한, 본 연구에서 대중교통 실적자료 내 승객의 이동경로와 알고리즘을 통한 계산경로의 비교를 진행하는 과정에서 지하철 노선 정보를 활용하지 못한 한계가 있다. 추후 대중교통 실적자료 내의 지하철 통행 기록을 추정하는 연구 결과를 적용하여 비교실험을 수행할 필요가 있다.

추후 연구에서는 실제 승객이 환승과 도보 이동에 대한 어떤 선호도를 가지고 이동 경로를 결정하였는지 추정할 수 있을 것으로 판단된다. 본 연구에서는 환승 저항을 고려한 경로 탐색을 제안하였는데, 이외에도 경로 선택에 영향을 주는 요금, 환승 편의성 등도 종합적으로 고려할 수 있도록 해야 할 것이다. 또한, 환승 저항은 환승 유형뿐만 아니라 환승 횟수나 O-D의 물리적 거리에도 영향을 받기 때문에 이를 고려한 알고리즘의 수정이 필요할 것으로 판단된다.

ACKNOWLEDGEMENTS

본 연구는 국토교통부 국토교통기술촉진연구사업의 연구비지원(18CTAP-C133228-02)에 의해 수행되었습니다.

REFERENCES

Arbex R. O. and da Cunha C. B.(2015), “Efficient transit network design and frequencies setting multi-objective optimization by alternating objective genetic algorithm,” *Transportation Research*, vol. 81, pp.355-376.

Cionini A., D'Angelo G., D'Emidio M., Frigioni D., Giannakopoulou K., Paraskevopoulos A. and Zaroliagis C.(2014), “Engineering graph-based models for dynamic timetable information systems,” *In OASIcs-OpenAccess Series in Informatics*, vol. 42, pp.46-61.

Delling D., Dibbelt J., Pajor T. and Werneck R. F.(2015), “Public transit labeling,” In *International Symposium on Experimental Algorithms*, Springer, pp.273-285.

Delling D., Pajor T. and Werneck R. F.(2012), “Round-based public transit routing,” *Proceedings of*

the Meeting on Algorithm Engineering & Expermiments, Kyoto, Japan, pp.130-140.

Dibbelt J., Pajor T., Strasser B. and Wagner D.(2013), *In International Symposium on Experimental Algorithms*, Springer, pp.43-54.

Garcia-Martinez A., Cascajo R., Jara-Diaz S. R., Chowdhury S. and Monzon A.(2018), "Transfer penalties in multimodal public transport networks," *Transportation Research Part A: Policy and Practice*.

Guo J. and Jia L.(2017), "A new algorithm for finding the K shortest paths in a time-schedule network with constraints on arcs," *Journal of Algorithms & Computational Technology*, vol. 11, no. 2, pp.170-177.

Hu X. and Chiu Y. C.(2015), "A Constrained Time-Dependent K Shortest Paths Algorithm Addressing Overlap and Travel Time Deviation," *International Journal of Transportation Science and Technology*, vol. 4, no. 4, pp.371-394.

Kim E. C. and Kim T. H.(2009), "K-path Algorithm for a Transfer of the Mobility Handicapped," *Seoul Studies*, vol. 10, no. 2, pp.147-159.

Kim E. J., Lee S. H., Cheon C. K. and Yu B. Y.(2017), "Development of the Algorithm of a Public Transportation Route Search Considering the Resistance Value of Traffic Safety and Environmental Index," *The Korea Institute of Intelligent Transport Systems*, vol. 16, no. 1, pp.78-89.

Madkour A., Aref W. G., Rehman F. U., Rahman M. A. and Basalamah S.(2017), *A survey of shortest-path algorithms*. arXiv preprint arXiv:1705.02044.

Park B. H. and Oh S. J.(2001), "Effects of Transfer Penalty in the Public Transit Assignment," *Social Economy & Policy Studies*, vol. 17, no. 2, pp.65-92.

Park H. C., Kim Y. S., Kang S. P. and Ko S. Y.(2012), "A Study of Diffenrence Between Intramodal and Intermodal Transfer Penalties," *Proceedings of the KOR-KST Conference*, vol. 66, pp.555-560.

Pyrga E., Schulz F., Wagner D. and Zaroliagis C.(2008), "Efficient models for timetable information in public transportation systems," *Journal of Experimental Algorithmics*, vol. 12, pp.1-39.

Strasser B. and Wagner D.(2014), "Connection scan accelerated," In *2014 Proceedings of the Sixteenth Workshop on Algorithm Engineering and Experiments*, pp.125-137.

Wang S., Yang Y., Hu X., Li J. and Xu B.(2016), "Solving the K-shortest paths problem in timetable-based public transportation systems," *Journal of Intelligent Transportation Systems*, vol. 20, no. 5, pp.413-427.

Witt S.(2015), "Trip-Based Public Transit Routing," In Algorithms-ESA 2015, Berlin, Heidelberg, Springer, pp.1025-1036.

Yang S. J.(2017), *A study on Route Choice Modeling in Rapid Transit Network Considering Transfer Penalty and Angular Cost*, Master thesis, Korea National University of Transportation, Seoul, Korea.

Yoo G. S.(2015), "Transfer penalty estimation with transit trips from smartcard data in Seoul, Korea," *KSCE Journal of Civil Engineering*, vol. 19, no. 4, pp.1108-1116.

Yoo H.(2017), *A Study for Improving the Convenience of Transfer in Urban Railway : Cases of the Airport Railroad*, Master thesis, Korea National University of Transportation, Seoul, Korea.