# Simulating Real Time Trip Planning for Rapid Bus Transit

*by* Ahmad Hasan Hammad Malik

# Simulating Real Time Trip Planning for Rapid Bus Transit

Project Team

| | |
|---|---|
| Ahmad Hasan Syed | p18-0126 |
| Hammad Malik | p18-0001 |

Session 2018-2022

Supervised by

Dr. Nouman Azam

**Department of Computer Sciences**

**National University of Computer and Emerging Sciences**
**Peshawar, Pakistan**

**June, 2022**

# Abstract

The purpose of this project is to develop a real time trip planning system that will efficiently plan a trip for user by taking starting location and user preferences such as (seat availability and nearest stop) as inputs.The system will generate multiple optimal paths to destination. The scope of project involves finding optimal routes according to user inputs and preferences. This will be accomplished through the use of optimal path finding algorithm (contraction hierarchies) and the rapid bus transit data. This collected data is used to figure out user preferences and the finding of optimal path to destination.

# Chapter 1

# Foundation and Introduction

This chapter gives a brief explanation towards the foundation and basic introduction of the project.The basis of foundation of the project is to figure out user preferences and efficiently plan an optimal and hassle free trip with real time route planning.

4

## 1.1 Introduction

In many developing countries , transportation development is uncontrolled and unsupervised, causing various problems to the environment and human welfare. Many developing countries therefore have considered a sustainable and ecologically friendly transport mode as one of the most important issues, including low emissions, fewer traffic accidents, and less congestion.So there is need of an efficient and reliable platform to persuade private 8 mode users to switch to use public modes keeping in view the ease and comfort of user.This is an 8 approach to reduce the impact on the ecology since the pollution emission from public modes is much lower than that from private modes per passenger.As trip planning has been a huge problem on public transportation specially when it comes to user Preferences.So this project focuses on planning a hassle free trip with an optimal path to destination and allows user to plan the trip according to prefrences like stops in the trip,seat availability.The inputs given by user will decide the output of the system.This system will plan a complete trip for user with real time route planning as it will start from the initial point i.e home etc. to the available nearest terminals and then according to user prefrences it will show multiple routes to destinations depending upon the prefrences and the best efficient path available.

### 1.1.1 Problem Description

Planning a trip on public transport reduces pollution and traffic congestion on roads but traveling on public transport has Many doubts and uncertainties as i.e where is the nearest terminal of the bus leading to our destination?,what time will it take and when we will be at our desired desired point and what chance it is that there will be an empty seat or if passenger has to stop somewhere in route so these all questions were noted by us in daily life problems and so it brought us together to design about a system that does it all for the passenger and is reliable and comfortable so passenger could choose public transport over private.

### 1.1.2 Motivation

Commute is an important factor in our daily lives and we prefer a comfortable and reliable platform for commute as reliability is a major factor for trip planning.There are vast gaps between the required and actual platfroms being used for daily commute as most basic requirements are not fulfilled for user experience.The uncertainty factor to travel using public transport is much higher than expected as user wants a hassle free trip which can be achieved only by catering all the user requirements and prrefrences all together and design a system that assures passenger with all basic and required prefrences and user can hence trust public transport as a daily commute.

The goal of this project is to provide something reliable covering the basic requirements and provide user a planned trip leading to destination with best optimal path and providing all the information necessary for the trip.

### 1.1.3 Goals and Objectives

The following are the targets and key points of this project:

1.Selecting the optimal algorithm for route planning (multiple routes).

2.Obtaining results using OSM and SUMO.

3.Monitoring vehicles and Passengers using SUMO.

4.Testing algorithm on OSM (open street maps).

—

# Chapter 2

# Review of Literature

We have gone through many research papers and while reading papers we have learned various types of techniques(algorithms) used for real-time route planning. Researchers used different techniques, traditional algorithms, hybrid algorithms, and heuristic algorithms for real-time route planning. Some of them used the Dijkstra algorithm, A* algorithm, and dynamic Dijkstra algorithm,Ant colony optimization algorithm [24] [15]. Some of them have RRT algorithm and their various types. Some of them used hybrid algorithms such as BAS-APF (Beetle antennae Search (BAS), Artificial potential field (PAF)). Some of them used the parallel computing technique to find the shortest path. In a research paper, the RRT* algorithm is defined, and then the type of RRT* is briefly defined such as B-RRT, RRT*FN, RRT*-Smart, and much more [16]. In another research paper, there is a comparison of four algorithms Dynamic Dijkstra, Dynamic A*, Floyd Warshall, and Contraction Hierarchies (CH) algorithm [19]. Attributes of road segments such as static attribute and dynamic attribute are clearly defined and find the shortest path all pairs shortest path (APSP) with parallel computing [14][7][5]. Mobile robots used different algorithms such as artificial potential field (APF), rapidly-exploring random trees (RRT), and probabilistic road map method (PRM) for path planning. Also used hybrid algorithm of A* and RRT [28]. In other research papers, the RRT algorithm is used and applied to real-life problems and shows good results [23]. In another research paper, Bellman-Ford and Dijkstra used to find the shortest path in applications like GIS [13]. So extracting out our best information and details we concluded that Contraction Hierarchies

(CH) algorithm is efficient in finding the real-time route planning in dynamic and as well as in the static environment[2]. This algorithm is already applied in some path-finding applications and this works well in both dynamic and static environments. We use this algorithm in our model to find real-time route planning and provide other services.

## 2.1 Optimal Path Planning with RRT*

Some traditional algorithms have some issues in optimality, non-holonomic, and large dimensional problems. In this paper with the contribution of a lot of papers, RRT* can solve this issue and perform efficiently in complex problems [17]. A lot of variants of RRT* can arrive in recent times and all work efficiently in complex route planning and non-holonomic situation. Some new techniques are now used with the RRT* [21] algorithms which are now used in different situations [10]. Some tools are spline with RRT*, NURBS, and segment using with RRT* works in the future.[11] A lot of applications use this algorithm for real-time routing problems and some more work is needed in the future.

## 2.2 Comparative study of an efficient algorithm

Four algorithms were compared in this paper, dynamic Dijkstra, dynamic A*, Floyd Warshall, and Contraction Hierarchies algorithm. These algorithms were compared on basis of average time, an average speed of a vehicle, an average waiting time of the vehicle, and average route length covered by vehicle from source to destination. The contraction hierarchies algorithm works from all of these algorithms in terms of space complexity, average speed, and average route length. But the waiting time of the CH algorithm is a little bit more than the other algorithm. But this algorithm works well in high-dimensional complex problems, dynamic attributes, and in real-time route planning. CH use two approaches: Top-down Bottom-up But top-down approaches work better than bottom-up approaches. In this paper top-down approach is used for finding the path and the result shows that CH >Floyd Warshall> Dynamic Astar>Dynamic Dijkstra. The positive point of this algorithm is that it is already applied to a real-life problem in urban areas and it works well in vehicle transportation problems (VTP).

## 2.3 Real-time Planning with Parallel Computation

The transportation network is essential in urban areas. A good transportation network helps the users to move freely in their desired time frame and this helps a lot in reducing air pollution and traffic congestion. Traffic congestion is the main issue in the urban area and that's why a lot of people use their vehicles. To enhance the transportation network, an agent-based model is used to choose the path efficiently. This works well in real-time path planning and also in dynamic situations. The all-pairs shortest path (APSP) can be calculated in real-time, CUDA-enabled Floyd Warshall algorithm and open-MP based navigation system can enhance the efficiency of the system [9]. When work is done in parallel it can take less time to complete the task and these times of algorithms are time efficient. When this algorithm can be applied to real-time problems it reduces the average time traveling by 45

## 2.4 Hybrid Heuristic Optimization Algorithm

As traffic increase day by day, a dynamic path is now necessary to solve this issue. Due to traffic, people avoid public transport and this can cause a lot of accidents and other problems as well. In this research paper, a hybrid algorithm is used to solve this problem.[27][22] The BAS-APF method is better used for real-time route planning as well as good time estimation. Beetle antennae search (BAS) and artificial potential field (APF) algorithm are combined used for real-time shortest path planning and time estimation.[8] This algorithm finds a feasible and safe path and compared it to another algorithm. But this algorithm is not implemented in real-life cases and it requires a lot of words before implementing in real-life scenarios.[1]

## 2.5 Flex Route transit under dynamic environment

So we start the trip and when the bus is in a static environment, no route changes throughout the destination. This can cause mismanagement because when the bus starts there is no traffic on the route but after some time traffic on the route and then this static pre-calculated route is no more efficient. To solve this problem, flex-route under a dynamic operating environment is used. The real-time route changes can increase the cost of the passenger[29]. Sometimes passenger pre-books the seat and then cancels the seat, this can also cost too much for the driver. To avoid this situation, the app can check the passenger's previous record before confirmation the seats. If the passenger record is bad then the app blocks the passenger. Flex-route under a dynamic environment can enhance the efficiency of the route and sometimes this can cause an increase in waiting time. Flex-route can reduce the unnecessary travel distance and due to cancellation, slack time will be increased. In the first phase, the intelligent transportation system (ITS), automatic vehicle location (AVL), mobile data terminal (MDT) is discussed and in the second phase mix, integer programming (MIP) is applied to solve the offline routing problem. In the last stage, dynamic scheduling is applied to tackle the real-time route planning in a dynamic environment which can increase the efficiency of the routing and can take less time to reach the destination.

## 2.6 Optimal Path Planning using Contraction Hierarchies

We consider the problem of computing shortest paths subject to an additional resource constraint such as a hard limit on the (positive) height difference of the path. This is typically of interest in the context of bicycle route planning, or when energy consumption is to be limited. [20] So far, the exact computation of such constrained shortest paths was not feasible on large networks; we show that [12] state-of-the-art speed-up techniques for the shortest path problem, like contraction hierarchies, [25] can be instrumented to solve this problem efficiently in practice despite the NP-hardness in general. We consider the problem of quickly computing shortest paths in weighted graphs. Often, this is achieved in two phases: (1) derive auxiliary data[26] in an expensive preprocessing phase, and (2) use this auxiliary data to speed up the query phase. By adding a fast weight-customization phase, we extend Contraction Hierarchies to support a three-phase workflow. [**?** ] The expensive preprocessing is split into a phase exploiting solely the unweighted topology of the graph and a lightweight phase that adapts the auxiliary data to a specific weight. We achieve this by basing our Customizable Contraction Hierarchies (CCHs) on nested dissection orders [3]. [6] We provide an in-depth experimental analysis on large road and game maps showing that CCHs are a very practicable solution [18] in scenarios where edge weights often change.

# Chapter 3

# Methodology

## 3.1 Overview

Real time route planning is a problem in which you are dealing with the dynamic state of envoirnment.As the environment around user is constantly changing and we need to figure out best optimal path for user. Algorithms such as CH(contraction hierarchies) you enter your initial and final points and they provide multiple optimal routes to the destination.The Algorithm is integrated into SUMO (urban mobility simulator)to simulate the optimal route planning for passenger.

3

### 3.1.1 Open Street Maps

OpenStreetMap (OSM) is a collaborative project to create a free editable geographic database of the world. The geodata underlying the maps is considered the primary output of the project. The creation and growth of OSM has been motivated by restrictions on use or availability of map data across much of the world, and the advent of inexpensive portable satellite navigation devices.We have used OSM to import and download the map of Hayatabad(Peshawar) in order to simulate our project in Hayatabad.

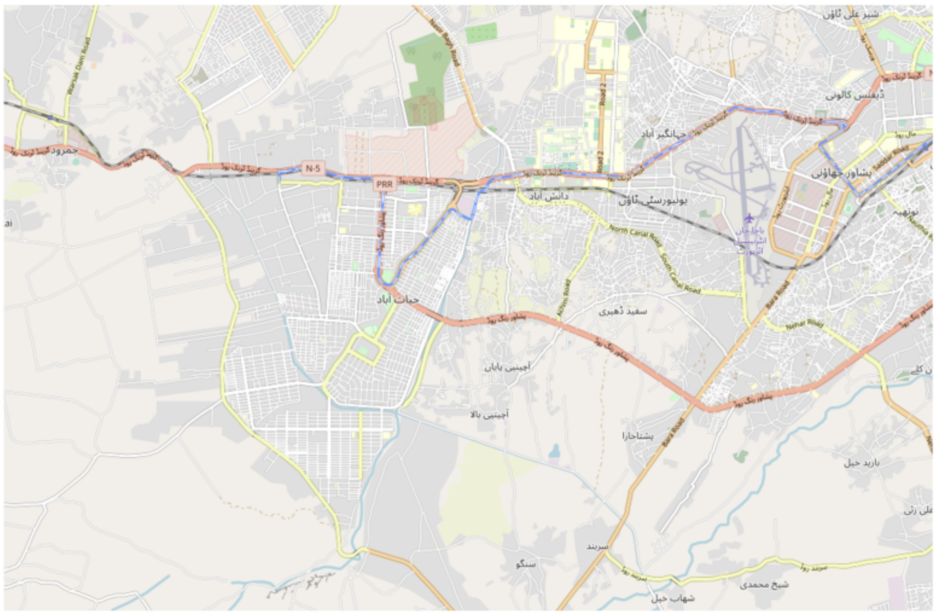


Figure 3.1: OSM(open street maps) Peshawar

### 3.1.2 NetEdit

Netedit is a visual network editor. It can be used to create networks from scratch and to modify all aspects of existing networks. With a powerful selection and highlighting interface it can also be used to debug network attributes.Using Netedit unwanted junctions and joints are removed from the graph of Peshawar.Netedit converts the osm imported map into a editable graph with nodes and edges to calibrate.In this scenario the map imported using OSM was map of Peshawar Hayatabad sector.As to remove the internal street edges going out of haytabad and constructing the edges and junctions for smooth traffic flow all this was brought to completion using net edit as it helps with obtaining the exact geo location and its edges and so it helps modifying the network file.

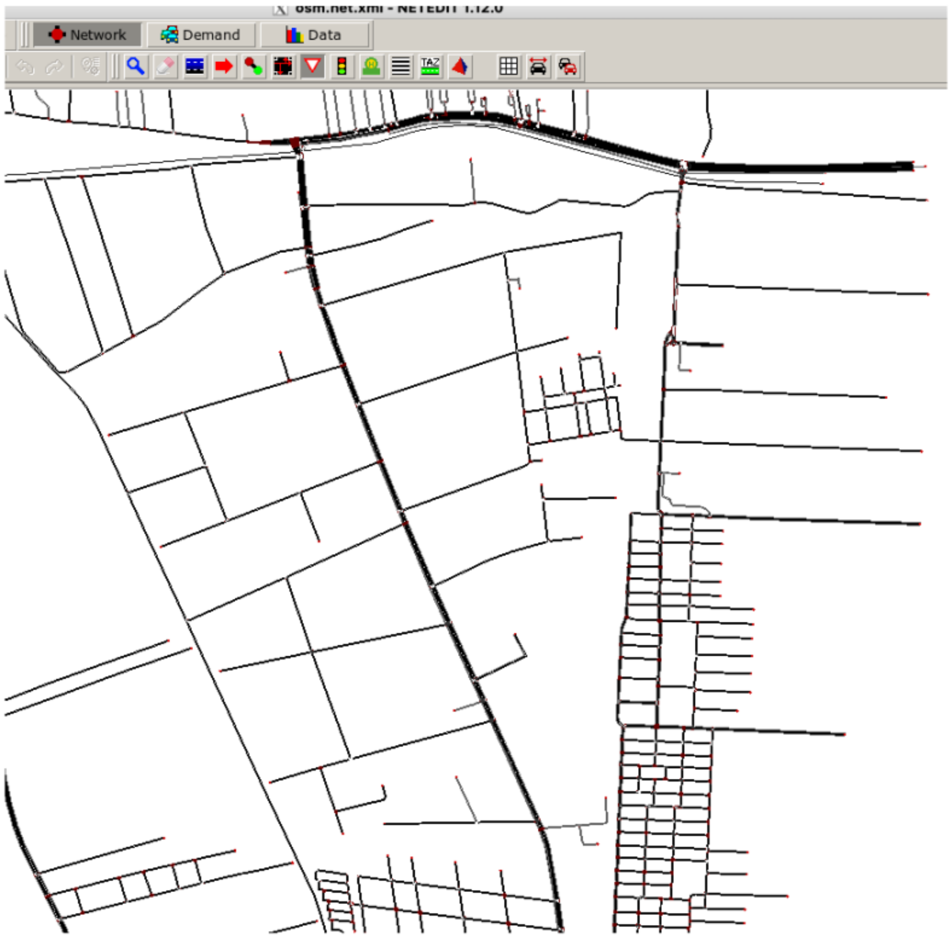


Figure 3.2: Netedit (Map to Graph)

### 3.1.3 SUMO

"Simulation of Urban MObility" (SUMO) is an open source, highly portable, microscopic and continuous traffic simulation package designed to handle large networks. It allows for intermodal simulation including pedestrians and comes with a large set of tools for scenario creation[4].SUMO GUI supports integration with NetEdit in order to edit the graph and process it side by side.SUMO has built in intelligence for traffic systems i.e(vehicle braking,traffic lights,lanes separation)etc.The buses on routes are scheduled on SUMO by calibrating the depart time of simulation and the bus.SUMO allows user to play the simulation at variable playback speeds in order to allow user to capture small events.

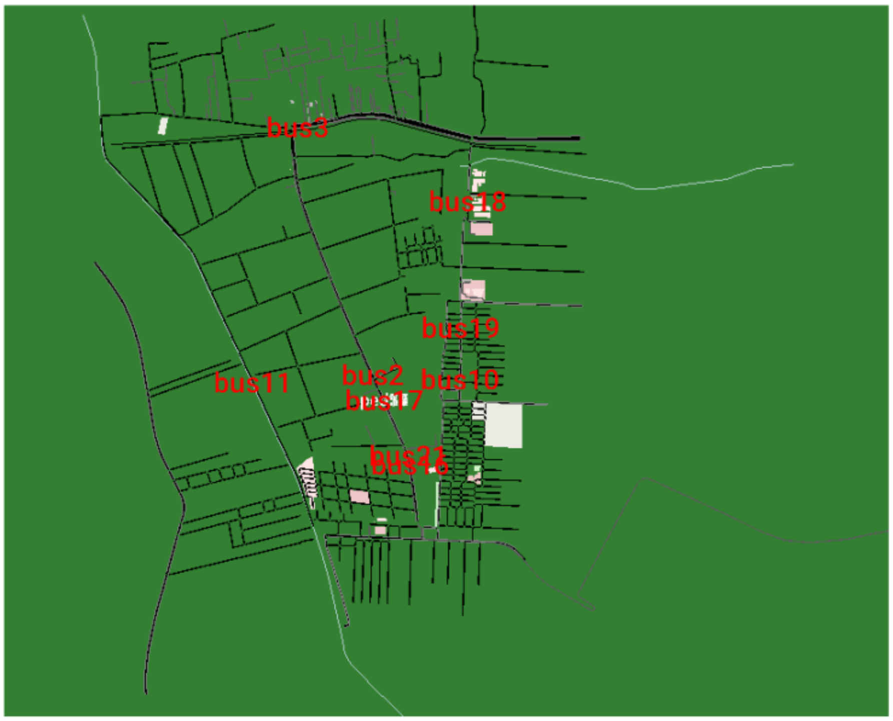


Figure 3.3: Simulations using SUMO

The figure3.4 shows the passengers loading in the bus from bus stop (id=busstop8).Multiple buses with different routes will reach at bus stop according to their schedules and stop at bus stops for a specific given amount of time (waiting for passengers to load).The code snippet to generate bus routes and stops is attached below (figure3.5).

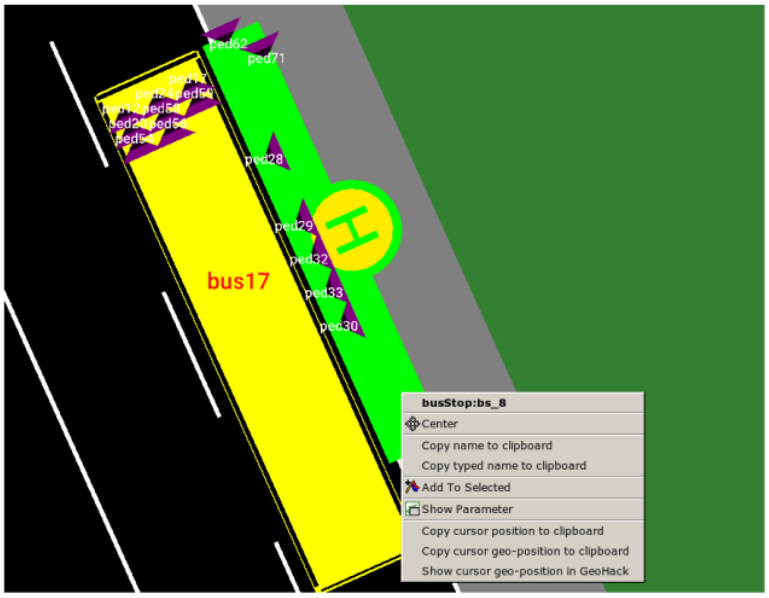
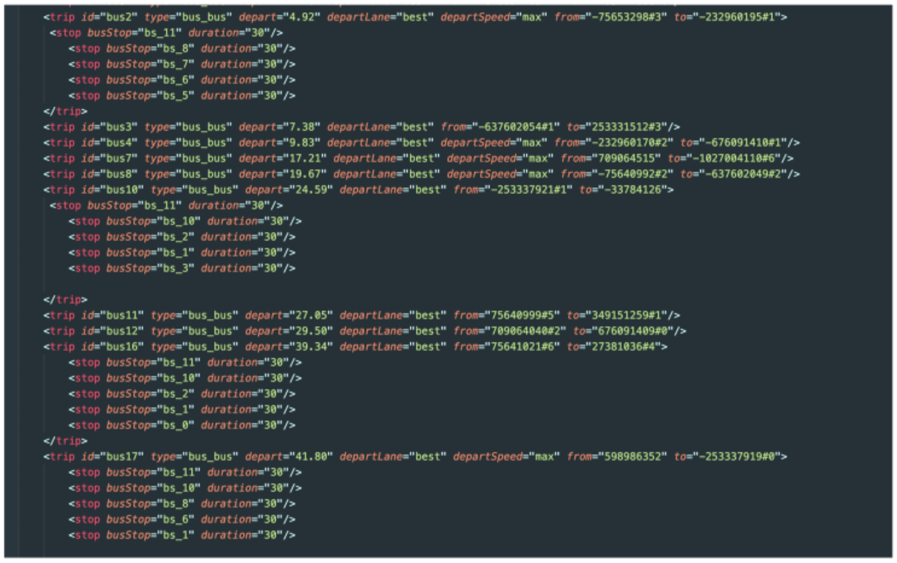

ped62
ped71
ped17
ped24
ped50
ped58
ped28
ped29
ped32
ped33
ped30
bus17
busStop:bs_8
Center
Copy name to clipboard
Copy typed name to clipboard
Add To Selected
Show Parameter
Copy cursor position to clipboard
Copy cursor geo-position to clipboard
Show cursor geo-position in GeoHack

Figure 3.4: passengers loading in bus at bus stop

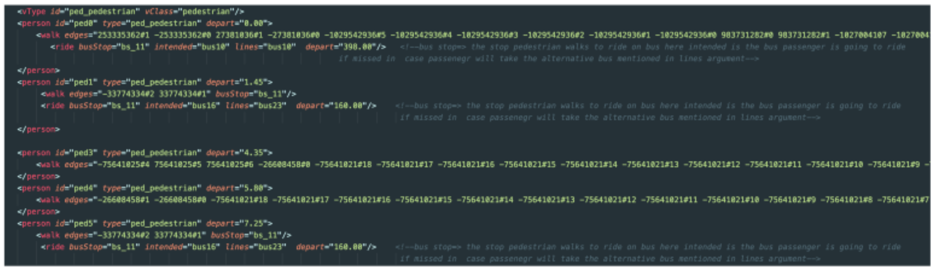


```
<trip id="bus2" type="bus_bus" depart="4.92" departLane="best" departSpeed="max" from="-75653298#3" to="-232960195#1">
 <stop busStop="bs_11" duration="30"/>
    <stop busStop="bs_8" duration="30"/>
    <stop busStop="bs_7" duration="30"/>
    <stop busStop="bs_6" duration="30"/>
    <stop busStop="bs_5" duration="30"/>
</trip>
<trip id="bus3" type="bus_bus" depart="7.38" departLane="best" from="-637602054#1" to="253331512#3"/>
<trip id="bus4" type="bus_bus" depart="9.83" departLane="best" departSpeed="max" from="-232960170#2" to="-676091410#1"/>
<trip id="bus7" type="bus_bus" depart="17.21" departLane="best" departSpeed="max" from="709064515" to="-1027004110#6"/>
<trip id="bus8" type="bus_bus" depart="19.67" departLane="best" departSpeed="max" from="-75640992#2" to="-637602049#2"/>
<trip id="bus10" type="bus_bus" depart="24.59" departLane="best" from="-253337921#1" to="-33784126">
 <stop busStop="bs_11" duration="30"/>
    <stop busStop="bs_10" duration="30"/>
    <stop busStop="bs_2" duration="30"/>
    <stop busStop="bs_1" duration="30"/>
    <stop busStop="bs_3" duration="30"/>

</trip>
<trip id="bus11" type="bus_bus" depart="27.05" departLane="best" from="75640999#5" to="349151259#1"/>
<trip id="bus12" type="bus_bus" depart="29.50" departLane="best" from="709064040#2" to="676091409#0"/>
<trip id="bus16" type="bus_bus" depart="39.34" departLane="best" from="75641021#6" to="27381036#4">
    <stop busStop="bs_11" duration="30"/>
    <stop busStop="bs_10" duration="30"/>
    <stop busStop="bs_2" duration="30"/>
    <stop busStop="bs_1" duration="30"/>
    <stop busStop="bs_0" duration="30"/>
</trip>
<trip id="bus17" type="bus_bus" depart="41.80" departLane="best" departSpeed="max" from="598986352" to="-253337919#0">
    <stop busStop="bs_11" duration="30"/>
    <stop busStop="bs_10" duration="30"/>
    <stop busStop="bs_8" duration="30"/>
    <stop busStop="bs_6" duration="30"/>
    <stop busStop="bs_1" duration="30"/>
```

Figure 3.5: Code snippet of bus trips and routes/stops

The start and ending edges of bus are given as input along with the bus stop id and the duration to stop at each bus stop all of these inputs are to be specified inside the file (bustrips) as this will now generate routes and schedule the buses at the stops.The depart speed can be scaled but if set at max as in figure3.5 then SUMO is intelligent and it scales the speed of the different buses on same route to avoid them to collide to each other.



```
<vType id="ped_pedestrian" vClass="pedestrian"/>
<person id="ped0" type="ped_pedestrian" depart="0.00">
    <walk edges="253335362#1 -253335362#0 27381036#1 -27381036#0 -1029542936#5 -1029542936#4 -1029542936#3 -1029542936#2 -1029542936#1 -1029542936#0 983731282#0 983731282#1 -1027004107 -1027004
      <ride busStop="bs_11" intended="bus10" lines="bus10"  depart="398.00"/>  <!--bus stop=> the stop pedestrian walks to ride on bus here intended is the bus passenger is going to ride
                                                                              if missed in  case passenegr will take the alternative bus mentioned in lines argument-->
</person>
<person id="ped1" type="ped_pedestrian" depart="1.45">
     <walk edges="-33774334#2 33774334#1" busStop="bs_11"/>
     <ride busStop="bs_11" intended="bus16" lines="bus23"  depart="160.00"/>    <!--bus stop=> the stop pedestrian walks to ride on bus here intended is the bus passenger is going to ride
                                                                                 if missed in  case passenegr will take the alternative bus mentioned in lines argument-->
</person>

<person id="ped3" type="ped_pedestrian" depart="4.35">
    <walk edges="-75641025#4 75641025#5 75641025#6 -26608458#0 -75641021#18 -75641021#17 -75641021#16 -75641021#15 -75641021#14 -75641021#13 -75641021#12 -75641021#11 -75641021#10 -75641021#9 -
</person>
<person id="ped4" type="ped_pedestrian" depart="5.00">
    <walk edges="-26608458#1 -26608458#0 -75641021#18 -75641021#17 -75641021#16 -75641021#15 -75641021#14 -75641021#13 -75641021#12 -75641021#11 -75641021#10 -75641021#9 -75641021#8 -75641021#7
</person>
<person id="ped5" type="ped_pedestrian" depart="7.25">
    <walk edges="-33774334#2 33774334#1" busStop="bs_11"/>
     <ride busStop="bs_11" intended="bus16" lines="bus23"  depart="160.00"/>    <!--bus stop=> the stop pedestrian walks to ride on bus here intended is the bus passenger is going to ride
                                                                                 if missed in  case passenegr will take the alternative bus mentioned in lines argument-->
```

Figure 3.6: Code Snippet of passengers trips

16

# Chapter 4

# Analysis and Design

## 4.1 Use Case Diagram

The primary usecases that system will perform to execute the proposed concept are decipted in the Use case diagram Figure 1.1.The user will start using the system by logging onto the system using credentials.The system will now ask for the initial input such as initial location(starting location) and arrival point(final destination).The system will now process the inputs received from the user and will show all the terminals near to the user on which the bus to final destination is scheduled.The user can then select the terminal (optimal according to the user).The system will now display the details of the terminal selected by the user such as distance and time to the terminal and the list of upcoming buses scheduled to the final destination.The total trip time from starting location to final destination will be displayed to user(including the time summed up at the terminal stops).User can now select the bus(suitable according to user).System will generate the bus route and the number of seats left in the bus(seat availability).In SUMO you can add the maximum loading capacity of the bus(Set to 15 maximum loading capacity in simulation)and the bus details are retrieved such as bus id(Bus number) and the highlighted route of the bus selected along with the route the route time and the stop time (delay time) on bus stops is displayed.The user can check if there is seat available in the bus selected as system will generate multiple buses through multiple routes to the final destination so user can select anyone if them.

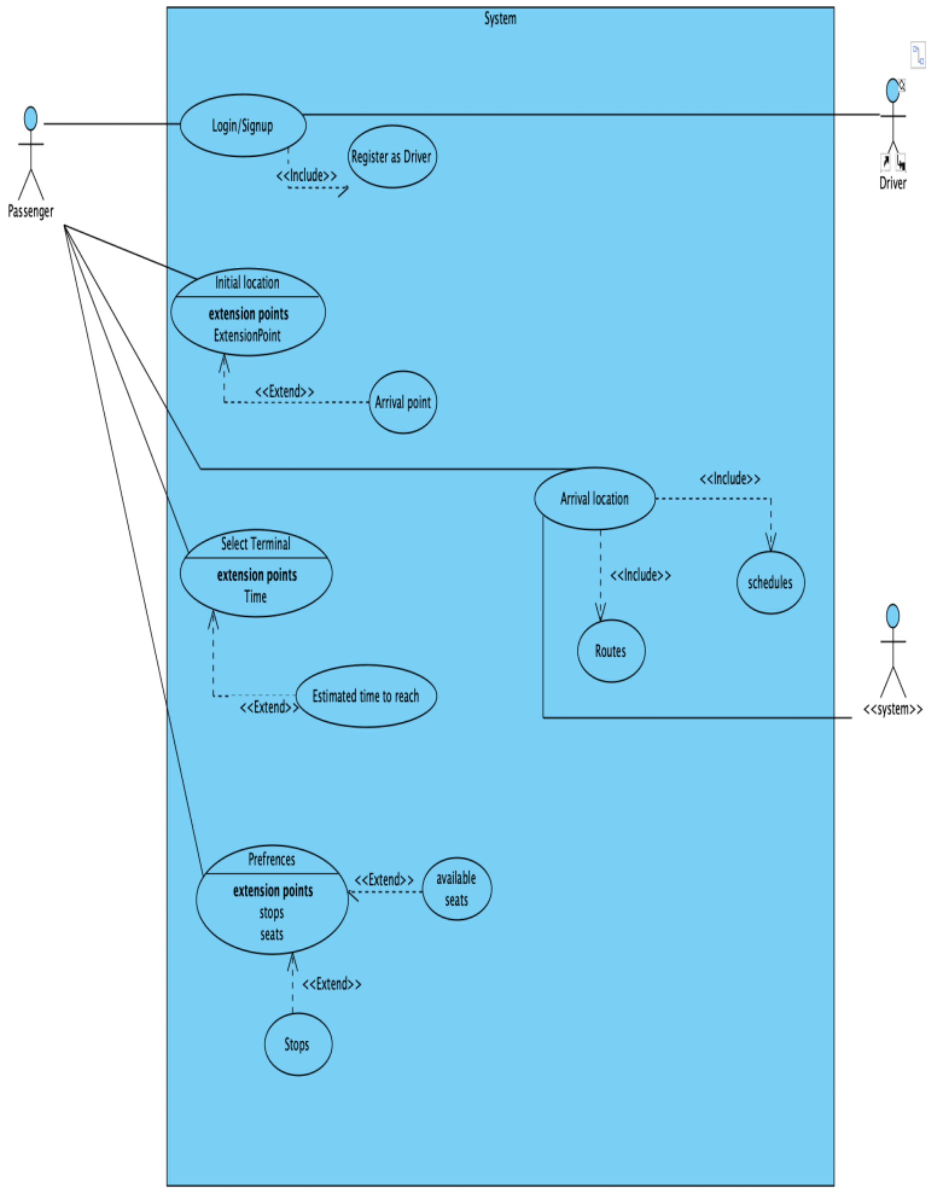


Figure 4.1: Use Case Diagram

9

## 4.2 Use Case and Use Case Description

| Use case Description | 1 |
|---|---|
| Goal | Initial Location |
| Precondition | User logged /registered onto system |
| Successful End Condition | Logged into system Successfully |
| Alternative Flow | Enter Location Manually |
| Main Flow (M) | 1. Open Application . . . <br> 2. log into system. . . <br> 3. Enter the initial location (can be obtained by GPS). . . <br> 4. Enter Arrival Point . . . |

| Use case Description | 2 |
|---|---|
| Goal | Prefrences |
| Precondition | Desired Bus to travel is selected |
| Successful End Condition | Bus Selected Successfully |
| Alternative Flow | Check for other available buses |
| Main Flow (M) | 1. Open Application . . . <br> 2. log into system. . . <br> 3. Enter the initial location (can be obtained by GPS). . . <br> 4. Enter the arrival point. . . <br> 5. Select the bus on route. . . <br> 6. Check for available prefrences i.e Seat availability . . . |

| Use case Description | 3 |
|---|---|
| Goal | Select Terminal |
| Precondition | Initial location successfully obtained |
| Successful End Condition | Stations in range listed |
| Failed End Condition | No station within range |
| Main Flow (M) | 1. Open Application . . .<br>2. log into system. . .<br>3. Enter the initial location (can be obtained by GPS). . .<br>4. Enter the arrival point. . .<br>5. Select the optimal terminal . . .<br>6. Route to selected terminal . . . |

| Use case Description | 4 |
|---|---|
| Goal | Arrival Location |
| Precondition | Seat available in desired Bus |
| Successful End Condition | Multiple routes to arrival point and one of them is selected and followed |
| Alternative Flow | Check other busses or other routes |
| Main Flow (M) | 1. Open Application . . . <br> 2. log into system. . . <br> 3. Enter the initial location (can be obtained by GPS). . . <br> 4. Enter the arrival point. . . <br> 5. Select the bus on route. . . <br> 6. Check for available preferences i.e Seat availability. . . <br> 7. Routes to arrival point . . . |

### 4.2.1 State Machine Diagram

This UML diagram highlights the inputs and outputs of the system along with each component processing the inputs of the system and producing outputs according to them.

The System is processing inputs w.r.t passenger (user) as the driver has only registration.Passenger can cancel or modify any preferences at two points while planning trip.

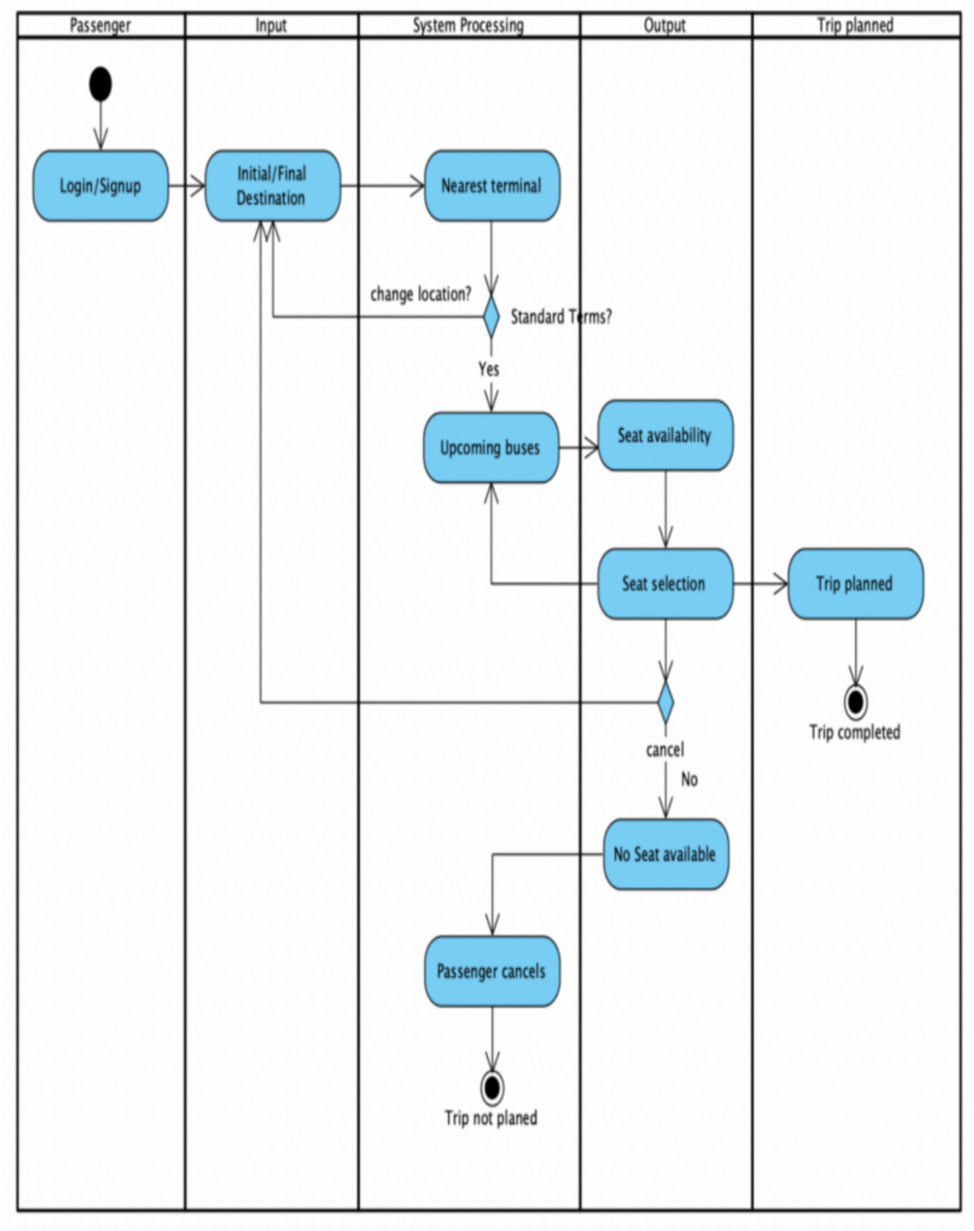


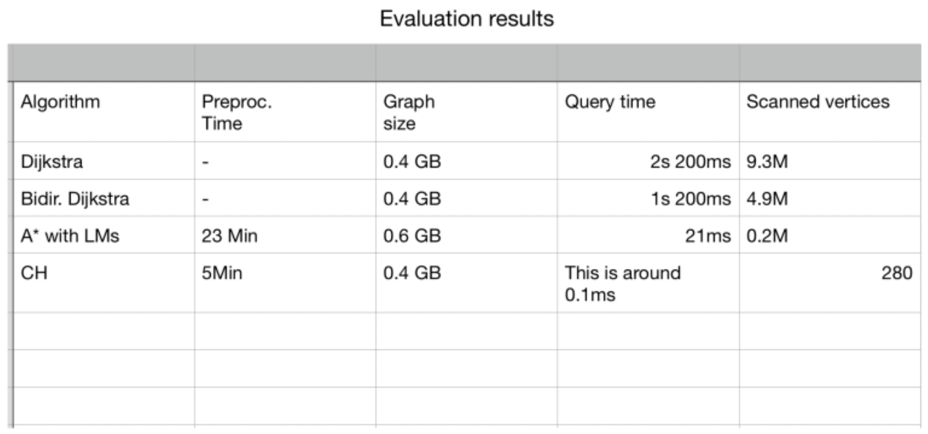
Figure 4.2: System Flow w.r.t passenger

# Chapter 5

# Results

## 5.1 Results of Algorithms

Testing different optimal route planning algorithms on map of berling using (OSM) maps in python (import OSM) and obtained results on the map and compared all of them with the selected algorithm for this project (contraction hierarchies)drove us to a conclusion to choose CH as route planning algorithm for this project as due to its pre-processing time and much lesser number of scanned vertices along with the query time for the given graph. The comparison results are attached below in figure5.1.

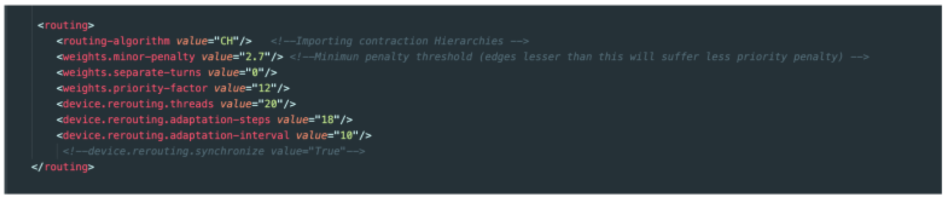


Evaluation results

| Algorithm | Preproc. Time | Graph size | Query time | Scanned vertices |
|---|---|---|---|---|
| Dijkstra | - | 0.4 GB | 2s 200ms | 9.3M |
| Bidir. Dijkstra | - | 0.4 GB | 1s 200ms | 4.9M |
| A* with LMs | 23 Min | 0.6 GB | 21ms | 0.2M |
| CH | 5Min | 0.4 GB | This is around 0.1ms | 280 |
| | | | | |
| | | | | |
| | | | | |

Figure 5.1: Evaluating route planning algorithms

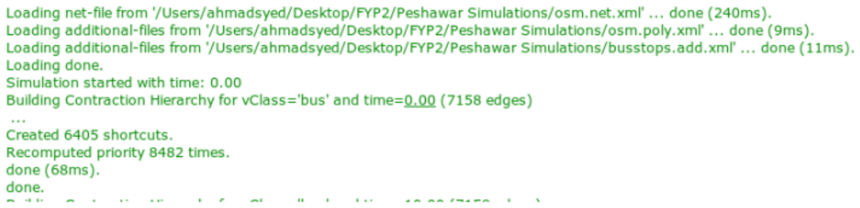
## 5.2 CH inegration and results on SUMO



```
<routing>
    <routing-algorithm value="CH"/>   <!--Importing contraction Hierarchies -->
    <weights.minor-penalty value="2.7"/> <!--Minimun penalty threshold (edges lesser than this will suffer less priority penalty) -->
    <weights.separate-turns value="0"/>
    <weights.priority-factor value="12"/>
    <device.rerouting.threads value="20"/>
    <device.rerouting.adaptation-steps value="18"/>
    <device.rerouting.adaptation-interval value="10"/>
     <!--device.rerouting.synchronize value="True"-->
</routing>
```

Figure 5.2: integrating CH into SUMO

The figure 5.2 shows the integration of Contraction Hierarchies into SUMO.The factors are calibrated according to the environment of Peshawar Hayatabad (can be later modified according to network graph) ,Here the minor penalty factor for weights is the minimum weight of the edge to be evaluated as a street all the edges having weight more than it are considered to be main highways (streets where buses can travel) all the edges having weight lesser than the minor penalty factor are to be considered as bike lanes or pedestrian lanes.(SUMO will allocate equal porportion of lanes left as per the initial settings).



```
Loading net-file from '/Users/ahmadsyed/Desktop/FYP2/Peshawar Simulations/osm.net.xml' ... done (240ms).
Loading additional-files from '/Users/ahmadsyed/Desktop/FYP2/Peshawar Simulations/osm.poly.xml' ... done (9ms).
Loading additional-files from '/Users/ahmadsyed/Desktop/FYP2/Peshawar Simulations/busstops.add.xml' ... done (11ms).
Loading done.
Simulation started with time: 0.00
Building Contraction Hierarchy for vClass='bus' and time=0.00 (7158 edges)
 ...
Created 6405 shortcuts.
Recomputed priority 8482 times.
done (68ms).
done.
```

Figure 5.3: System Results

The output results of CH(contraction Hierarchies) on the map of Peshawar (Hayatabad) are mentioned in figure as the number of shortcuts created by the system on map and re-computation of priority of edges (minor penalty weight is set for this purpose) is attached in the output of system.The system was tested and simulated on Macbook Pro M1(2020).

# Chapter 6

# Discussions

Our project has main two objectives,firstly we have to select the optimal routing algorithm for route planning and the second is to simulate the project on a intelligent simulation tool in order for things to work smoothly.

## 6.1 Selecting the optimal route planning algorithm

In order to select the optimal route planning algorithm we had to go through thoroughly through the best known route planning algorithms we had to consider many factors such as pre-processing time,query time,graph size,augmented graph size taking all these factors under consideration and providing the initial and final location (edges) to the algorithms (algorithms were tested using OSM library in python and importing graph into python) we decided to choose the contraction hierarchies due to the fastest query time and pre-processing plus the additional feature that attracted us towards it was that it removes all the unnecessary edges and vertices from the graph and directly creates shortcuts by removing them this leads to much smaller graph size handling and generation of multiple shortcuts (multiple optimal routes) as it pre processes the graph removes shortcuts (edge priority) and recomputes the priority of edges at each step so that if there is any unnecessary node to be removed or any shortcut to be added

## 6.2 Approach towards SUMO

After selecting the optimal route planning algorithm the next step was to select a framework to put this all together as we have mentioned above we tested the algorithms on (import OSM) python but to simulate this project we were searching for a transport simulator and we ended upon SUMO (urban mobility simulator) ,the next step was to integrate the algorithm into SUMO.The algorithm integration required all the parameters for algorithm such as weights of edges to prioritise and rerouting techniques as after integrating the algorithm into SUMO the next upcoming task was to generate traffic (SUMO has support for it) SUMO allows you handle traffic management along with its own intelligent algorithm to generate a scenario.

# Chapter 7

# Conclusions and Future Work

## 7.1 Conclusions

In this project we tried to solve a problem related to daily transit.The problem is addressed as the daily commute is a normal part of our lives and it has been affecting our lives due to many factors and some key factors were uncertainty and route planning.The project is divided into two part 1)Optimal route planning. 2)User preferences.In part 1 we are dealing with the problem of optimal route planning for passenger and for this purpose we have used contraction hierarchies Dijkstra(modified) form as this algorithm has much lesser pre-processing time compared to other route planning algorithms studied as well as it generates optimal and multiple routes to destination.The 2nd part of the project was to deal with the user preferences (uncertainty problem) in this part user preferences such as bus capacity and the time bus is going to arrive and if there is any seat available in the bus all of these uncertainties are covered.The final part was to simulate the project as bringing this all to completion for this purpose SUMO(urban mobility simulator) is used along with some supporting tools 1)OSM(open street maps) used to import the real world maps and work on it as it is a free open source world map to use and work on 2) Net edit it is used to edit the graph (maps imported using OSM) so that if any modifications or edges/nodes to be added or deleted from the graph.3)SUMO sumo is a urban mobility simulator in which we integrated the routing algorithm (contraction hierarchies) and simulated the bus transit on the map of Peshawar (imported using OSM and edited using netedit).

## 7.2 Future Work and Limitations

We have identified several extensions as strong opportunities for the future of this project. One path would be to export this project to smartphones, building an application that allows users to plan a trip using their initial and final location this would also include the live tracking of bus for the user to track bus and check if there is seat available to travel. Another direction would be to export this project to web based application where users can check the bus status (live tracking) along with the seat availbility and the nearest terminal to it and all the other terminals that are suitable to user can be displayed to user to give user a variety if choices.

# Simulating Real Time Trip Planning for Rapid Bus Transit

ORIGINALITY REPORT

| 13% | 12% | 8% | 5% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | explora.unex.es<br>Internet Source | 3% |
| 2 | www.aaai.org<br>Internet Source | 2% |
| 3 | webthesis.biblio.polito.it<br>Internet Source | 1% |
| 4 | Thaned SATIENNAM, Atsushi FUKUDA, Ryosuke OSHIMA. "A STUDY ON THE INTRODUCTION OF BUS RAPID TRANSIT SYSTEM IN ASIAN DEVELOPING CITIES", IATSS Research, 2006<br>Publication | 1% |
| 5 | Submitted to Liverpool John Moores University<br>Student Paper | 1% |
| 6 | hprc.tamu.edu<br>Internet Source | 1% |
| 7 | Yue Zheng, Liangpeng Gao, Wenquan Li. "Vehicle Routing and Scheduling of Flex-Route Transit under a Dynamic Operating | 1% |

|  |  |  |
|---|---|---|
|  | Environment", Discrete Dynamics in Nature and Society, 2021<br>Publication |  |
| 8 | docslide.nl<br>Internet Source | 1 % |
| 9 | etd.astu.edu.et<br>Internet Source | <1 % |
| 10 | dspace.cvut.cz<br>Internet Source | <1 % |
| 11 | doaj.org<br>Internet Source | <1 % |
| 12 | brage.bibsys.no<br>Internet Source | <1 % |
| 13 | publications.waset.org<br>Internet Source | <1 % |
| 14 | cora.ucc.ie<br>Internet Source | <1 % |
| 15 | www.mdpi.com<br>Internet Source | <1 % |
| 16 | www.slideshare.net<br>Internet Source | <1 % |
| 17 | Li, Yong, Shitai Bao, Kui Su, Qiushui Fang, and Jingfeng Yang. "", PIAGENG 2010 Photonics and Imaging for Agricultural Engineering, 2011. | <1 % |

Publication

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | Off |
| Exclude bibliography | On | | |