SKIM GENERATION FOR BANGALORE METRO, SUBURBAN RAIl & TRANSIT

A Thesis
Submitted For the Degree of
Master of Technology
in the Faculty of Engineering

by

Sukanto Mette

Under the guidance of

Tarun Rambha

Dept. of Civil Engineering
Indian Institute of Science
BANGALORE – 560 012

June 2023

To

My Parents, Friends

Acknowledgements

I am deeply grateful for the support and guidance I received during my time as a Master's student at IISc. My advisor, Dr. Tarun Rambha, played a critical role in my success, offering valuable insight and guidance throughout my studies.

I am also thankful for the invaluable coursework and insights provided by Prof. Abdul Rawoof Pinjari and Prof. Ashish Verma. The Department of Civil Engineering and Center for Infrastructure, Sustainable Transportation and Urban Planning (CiSTUP) at IISc also played an important role, providing a comfortable working space and support from the Chair, faculty members, and staff members.

I also want to give special recognition to Prateek Agarwal for his constant guidance and support throughout the project, and acknowledge the contributions of interns like Dhanus and Kalash for their help in the project.

All of these individuals played a critical role in the successful completion of my work and I am deeply appreciative of their support.

SUKANTO METTE

Abstract

Public transport planning is essential for creating sustainable and liveable cities, as it involves designing and implementing transportation systems that are convenient, reliable, and accessible to all members of the community. Forecasting demand for public transportation is crucial for planning and can be done using mathematical models known as travel demand models. These models are used to project future traffic and inform decisions about new road capacity, transit service changes, and land use policies.

The transit assignment problem is a key aspect of public transport planning, as it is used to estimate passenger ridership and travel times for different line and frequency plans. There are two main approaches to transit assignment: schedule-based and frequency-based. The core of the assignment model is the assumption about how passengers make route choices, which can include several legs such as walking to and from stops, waiting for a line, riding a vehicle, and transferring to other lines.

Passenger route choice depends on various attributes such as traveller characteristics like socio-demographics, attitudes, lifestyle, and route-specific attributes like access time, egress time, in-vehicle travel time, out of vehicle travel time, transfer time, waiting time, fare, etc. These route-specific attributes are called skims. This project aims to generate skims for the Bangalore metro, suburban rail & transit network for different origin-destination zone pairs along an optimized route between source and destination zones. This will provide valuable insights into passenger travel behaviour and help inform future public transport planning decisions in Bangalore.

Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

BMRC	Bangalore Metro Rail Corporation
GTFS	General Transit Feed Specification
IVTT	In-Vehicle Travel Time
OVTT	Out-Vehicle Travel Time
PMR	Public Mobility Routing
PTR	Public Transit Routing
RAPTOR	Round-based Public Transit Routing
TD	Time-Dependent

Chapter 1

Introduction

1.1 Introduction

Public transport planning is crucial for the efficient functioning of cities and towns, and involves designing and implementing transportation systems that are convenient, reliable, and accessible to all members of the community. Proper public transport planning can help reduce traffic congestion, improve air quality, and promote economic development. It also plays a key role in addressing social issues, such as accessibility for people with disabilities and low-income communities. Overall, public transport planning is essential for creating sustainable and liveable cities.

Forecasting demand for public transportation is crucial for planning and can be done using mathematical models that simulate human behaviour while traveling. These models, known as travel demand models, are used to project future traffic and inform decisions about new road capacity, transit service changes, and land use policies. The travel simulation process is known as the four-step process, which includes the models for trip generation, trip distribution, modal split, and traffic assignments. These models are based on assumptions about how people make decisions and consider various factors when choosing transportation alternatives. The process begins with trip generation in a specific zone and ends with trip attraction in another zone, and simulates the movement of trips through a network of links and nodes.

The transit assignment problem is the mapping of passenger demand onto a transit network, and is a key aspect of public transport planning. It is used to estimate passenger ridership and travel times for different line and frequency plans. There are two main approaches to transit assignment: schedule-based and frequency-based. Schedule-based models simulate detailed time-dependent transit assignments and are commonly used for simulation purposes. Frequency-based models are commonly used for planning purposes, providing the average distribution of passengers over time and can handle large-scale networks. The core of the assignment model is the assumption about how passengers make route choices, which can include several legs such as walking to and from stops, waiting for a line, riding a vehicle, and transferring to other lines.

Passenger route choice depends upon various attributes such as traveller characteristics like socio-demographics, perception, lifestyle and route-specific attributes like access time, egress time, in-vehicle travel time, out of vehicle travel time, transfer time, waiting time, fare, etc. These route-specific attributes are called skims. This project will generate skims for the Bangalore metro, suburban rail & transit network and there combination for different origin-destination zone pairs along an optimized route between source and destination zones. This will provide valuable insights into passenger travel behaviour and help inform future public transport planning decisions in Bangalore.

1.2 Literature Review

The literature on public transport route choice has been limited over the past 20 years, with many studies relying on stated preference (SP) data. However, a recent study by Anderson, Nielsen, and Prato (2014) Marie Karen Anderson (2014) addresses this limitation by using revealed preference (RP) data to model actual route choices in a large and complex multimodal public transport network in the Greater Copenhagen Area. The study proposes a mixed path size correction logit model that takes into account both the similarities among routes and the heterogeneity among travelers. The study found that considering heterogeneity across travelers, explicitly accounting for the frequency of services, and examining trip purposes and trip length are important for explaining traveler behavior. This study provides valuable insight into public transport route choice

and highlights the need for more research using RP data.

The meta-analysis by Pedro A.L. Abrantes (2011) in the Transportation Research Part A journal updates and extends previous meta-analyses of UK values of travel time by including recent studies and a wider range of explanatory variables. The study found a significant correlation between the value of time and GDP per capita, and a lower ratio between walk and wait time compared to in-vehicle time. Additionally, the study found that a toll numeraire reduces the value of time by just over 20 percent, and that car time spent in congested traffic conditions is valued 34 percent more highly than time spent in free flow traffic. Another study by Wardman (2004) focuses on how the value of time varies over time and found a positive distance elasticity for in-vehicle time.

The study "Understanding the route choice behaviour of metro-bikeshare users" by Yang Liu (2022), and He uses smart card data to analyze the route choice behavior of metro-bikeshare users in order to improve personalized travel guidance services and operationalize Mobility as a Service (MaaS) in practice. The authors introduced the concept of network load status, which reflects users' perception of congestion, and found that it has a significant impact on route choice behavior. The study also found that factors such as change rate of shared bike inventory, departure time, and whether the user is a regular user also have an impact on route choice behavior. Overall, this study offers useful insights on how congestion in the metro system can impact users' travel choices and can be used to improve services and efficiency of the multimodal transportation system.

In the paper "Sensitivity to travel time variability: Travelers' learning perspective" by Erel Avineri (2005), the authors conduct route-choice laboratory experiments and computer simulations to analyze route-choice behavior under uncertainty in transportation systems. They find that as the variance in travel times increases, the sensitivity of travelers to travel time differences decreases, which conflicts with traditional views on risk-taking behavior in route-choice decision making. The authors suggest that a learning-based approach, such as reinforcement learning, may be more appropriate for modeling route-choice decisions under uncertainty. They also note that further research is needed to determine the best approach in specific situations and to fully understand the implications of these findings.

The literature review in the article "Refined choice set generation and the investigation of multi-criteria transit route choice behavior" by Benjamin J. Tomhave (2022) focuses on various methods for identifying the route choice set, including K-shortest path, link elimination, schedule-based shortest path, doubly stochastic generation, and a multivariate generalized cost labeling approach. However, it also highlights that data acquisition for applying the transit route choice model is a challenge. The article proposes a new method for estimating transit route choice models using on-board transit survey data and an iterative trip elimination methodology which yields high-quality transit path choice sets and detailed temporal information on all types of network links. The proposed choice set generation methodology is a major contribution of this study as it systematically generates choice sets of between 2 and 15 attractive paths for each passenger. The results indicate that passengers do not perceive the passage of time uniformly and that passengers perceive the relative disutility of waiting to be three times larger than local bus in-vehicle time.

This paper, "A frequency based transit assignment model that considers online information and strict capacity constraints" by Nurit Oliker (2020), proposes an efficient heuristic approach to integrate capacity constraints in a frequency-based transit assignment model that also considers online information. The model uses a path-based approach and prioritizes upstream passengers over downstream passengers in the re-assignment procedure. The model is applied to the Winnipeg network, and it was found that prior knowledge of the occupancy condition can significantly reduce travel time. The paper also suggests that the proposed model can be useful in the planning and management of transit networks and highlights the potential benefits of providing occupancy information to passengers.

In the schedule-based approach to dynamic transit modeling, a path between an origin and destination is defined by the space-time sequence including origin departure time, access to the first boarding stop, the run or sequence of runs with departure and arrival times, and egress from the last stop to the destination. The schedule-based approach considers the explicit departure and arrival times of the runs and the associated arrival times at each stop, instead of just the frequency of the service. Path choice models in this approach are specified using utility theory, where users are assumed to be rational and choose the path that maximizes their perceived utility. The probability of choosing a

specific path is based on the utility of that path compared to the utilities of other paths in the choice set. These models can be deterministic or stochastic, depending on whether or not random residuals are included in the utility function. (Reference: Nuzzolo (2004)).

Skim generation is an important process in transportation planning that aims to provide accurate and detailed information for each cell in a traffic analysis zone (TAZ) matrix. This information is used to predict travel times and mode choice for a set of synthetic individuals, enabling the creation of a daily plan. The process of generating skims is computationally heavy, as for each sampled point in each zone a complete "shortest path" tree has to be constructed from that point to all other points. To address this computational complexity, many studies have used static models of the transportation network, simplifying the network from a complex time-dependent network to a simpler network with static costs between each transit stop. These static representations do not model dwell times, boarding and alighting times, traffic congestion, or transfers. However, studies such as Thomas Koch (2020) in the paper "Door-to-door transit accessibility using Pareto optimal range queries" introduce real multi-modal trips by accounting for the first and last mile legs of journeys, mode change, and transfers. TAZ to TAZ travel times are computed by sampling locations in the origin and destination zones and by computing sets of Pareto optimal travel options. The authors show that in many cases the total travel time is not normally distributed, but rather has a bi-modal or even tri-modal distribution. They also show how this originates from accessibility properties of public transit stops. This research provides a more accurate and detailed information for each skim matrix cell in traffic analysis zones, taking travel time variability into account.

Chapter 2

Description of the Datasets

The algorithm used to find the most efficient journeys between the source stop and the destination stop for public transit and utilizes GTFS as input data. The General Transit Feed Specification (GTFS) is a format that defines a common format for public transportation schedules and associated geographic information. It is also referred to as GTFS static or static transit to distinguish it from the GTFS real-time extension. GTFS feeds allow public transit agencies to share their transit data, and developers to create applications that can use that data in a standardized way. The data is organized into a series of text files that are collected in a ZIP file. These files include information such as stops, routes, trips, schedule data and fare.

The GTFS data for the Bangalore metro and suburban rail had to be created from scratch as it was not previously available. To do this, the schedule available on the BMRC website and detailed project report by RITES for suburban rail was used to gather information about the metro lines and the suburban rail respectively. This data was then used to create the necessary files, such as stops.txt, routes.txt, trips.txt, and stoptimes.txt, using a python code. The code can also be used to generate data for additional metro lines in the future. Additionally, fare_attributes.txt and fare_rule.txt files were also created to estimate fares for the journeys.

The following files are commonly found in a GTFS dataset

Agency.txt: This file contains details about the transit agency or organization responsible for transportation services. It includes information such as agency ID, name, website

URL, and time zone.

Stops.txt: Stops.txt provides comprehensive information about stops or stations within the transit system. It encompasses data such as stop ID, name, latitude, and longitude.

Routes.txt: Routes.txt describes the different routes operated by the transit agency. It includes details such as the route ID, route short name, route long name, route description and route type (type of transportation used e.g. bus, metro, suburban rail).

Trips.txt: Trips.txt contains relevant information regarding the trips made by transit vehicles on specific routes. It includes essential fields such as trip ID and route ID.

Stoptimes.txt: Stoptimes.txt presents the timetable information for each trip, specifying the sequence of stops along with their respective arrival and departure times. Additional details may include stop sequence number and stop ID.

Transfers.txt: Transfers.txt provides information on footpath connections within the transit network, facilitating transfers between stops. It includes fields like source stop ID, destination stop ID, and minimum transfer time.

Fare Attributes.txt: Fare Attributes.txt provides insights into the fare structure of the transit system. It encompasses information such as fare ID, price, and currency.

Fare Rules.txt: Fare Rules.txt defines fare rules associated with specific fare IDs and route IDs. It includes details like fare ID, origin and destination stop IDs, and route ID.

Note: The fare rule file for bus and metro networks varies slightly due to differences in fare calculation methods. In a transit network, the fare is determined by summing the fares of all the routes involved in the journey. However, for metro and suburban rail, the fare is calculated based on the source and destination metro or rail stops, without considering specific route IDs. As a result, the fare rule file for metro and suburban rail does not include route IDs.

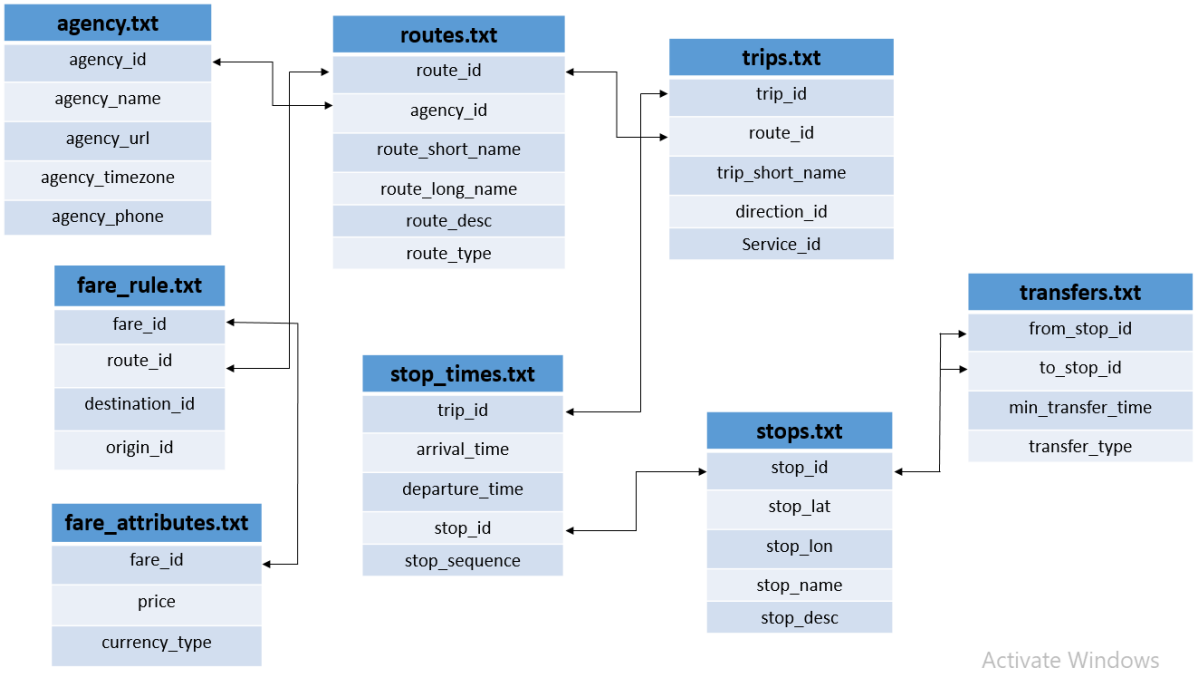


Figure 2.1: Relation among different files in GTFS

2.1 Existing And Upcoming Metro Network

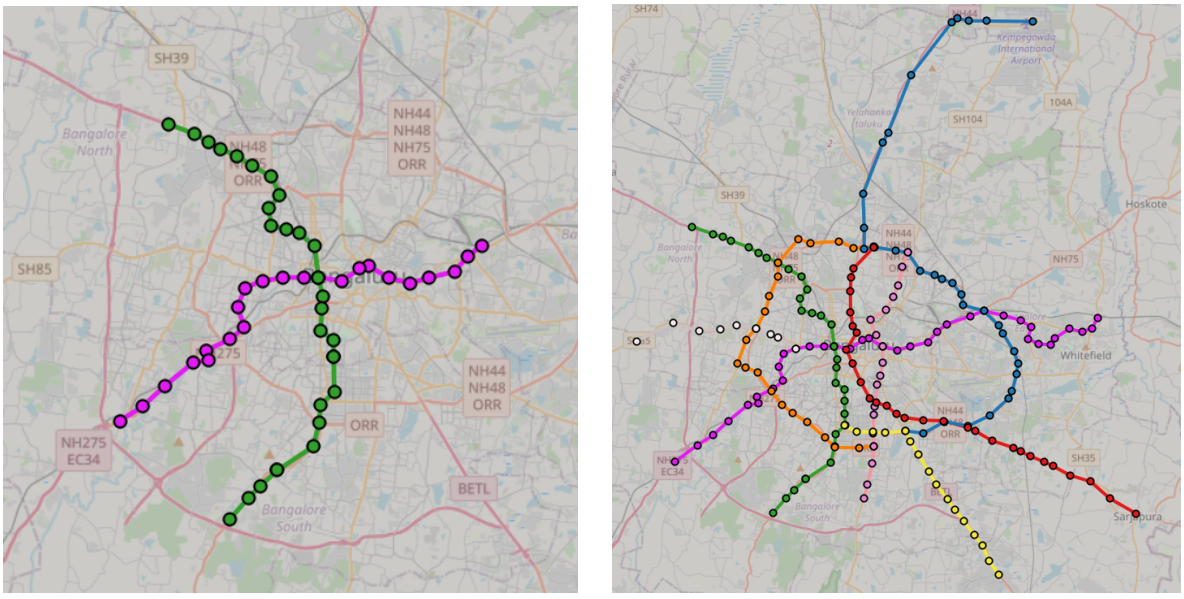
The stops.txt file, which contains information about the stations, had to be created manually by entering the name, latitude, and longitude of each station in the code. There is no automated logic for this process. The stop IDs for each station were assigned based on the network abbreviation(M for metro), line colour code and their sequence on the line. For example, "Baiyappanahalli" has the stop ID of M_P_1 and "Kengeri" has the stop ID of M_P_23 as these stations belong to the purple line of metro network and for transfer stop like "majestic" will have two stop ID one for purple and one for green line.

The assignment of route IDs in the metro system was based on a combination of the network abbreviation(M for metro), line color, and the first letter of the starting stop. For instance, the route ID for the green line from "Nagasandra" to "Silk Institute" is designated as M_GN, with "N" representing the first letter of the starting stop. In the case of routes in the opposite direction, the route ID depends on the network abbreviation, line color, and the first letter of the starting station in that direction. This information is stored in the *routes.txt* file. Currently, the metro system has four routes, while the future

metro in Bengaluru will have a total of 16 routes.

Figure 2.2: Bangalore Metro (source : https://en.wikipedia.org/wiki/Namma_Metro)

To create the *trips.txt* file, the number of trips on each route is needed. This information was obtained by using the time schedule for each route present on the BMRC website for the current metro and for the future metro network schedules has been kept same as of current purple line. The schedule includes the frequency of metro on each route, which can vary throughout the day. Using this information, the number of trips made on each route in one day can be determined. The trip IDs are assigned using the network abbreviation(M for metro), route ID and a corresponding number, which ranges over the total number of trips on that route. For example, the purple line from Whitefield to Challaghatta has 152 trips and the trip IDs range from M_PW_1 to M_PW_152. Similarly, for other routes, the trip IDs are assigned based on the number of trips on that route.



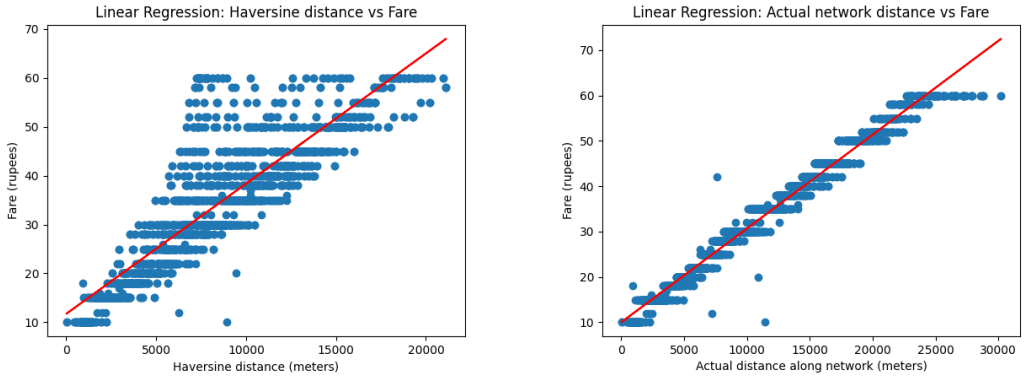
(a) Existing metro **(b)** Future metro

Figure 2.3: Bengaluru current and upcoming metro network

The arrival time at each stop on a route can be determined by using the starting time and frequency of a trip, as well as the time difference between consecutive stops. This information, along with the data for all the trips, is stored in the *stoptimes.txt* file. The time difference between consecutive stops was found using the timing information provided by the Bangalore metro website (https://bangaloremetrotimings.com/#lines).

The *transfers.txt* file was created to store information about the connections between stops for walking or transferring. The file includes the origin stop as from_stop, the destination stop as to_stop and the minimum transfer time required by walking between them. For example the current Bangalore metro lines, only one transfer station is available at Majestic, and the minimum transfer time is assumed to be 2 minutes. This means that a passenger who wants to travel from any stop on the purple line to any stop on the green line must transfer at Majestic station, and vice versa.

In addition to the other files, *fare_attributes.txt* and *fare_rule.txt* files were created to estimate the fare for a metro journey. The *fare_rule.txt* file helps to determine the fare ID for a given origin, destination and the fare corresponding to this ID can be found in the *fare_attributes.txt* file. These fare files were created using the fare charges provided by BMRC for the current metro network.



(a) (b)

Figure 2.4: Plots depicting the linear regression fit for fare prediction using either haversine or actual network distance.

For the future metro network, fare calculation was performed using linear regression with two potential independent variables: haversine distance and actual distance along the metro network. However, the figure 2.4 illustrates that the actual network distance provides a more accurate prediction. Hence, fare calculation is based on the actual network distance.

Note: A web scraping python code has been created to automatically extract all 2652 origin-destination fares for current metro network from the BMRC fare website (http://fare.bmrc.co.in/), eliminating the need for manual retrieval.

2.2 Suburban Rail Network

The assignment of stop IDs for each suburban rail station was determined by the network abbreviation (R for rail), line name code, and their sequence on the line. For instance, "KSR Bengaluru City" was assigned the stop ID of R_S_1, while "Devanahalli" was assigned the stop ID of R_S_15, as both stations are part of the Sampige line in the suburban rail network.

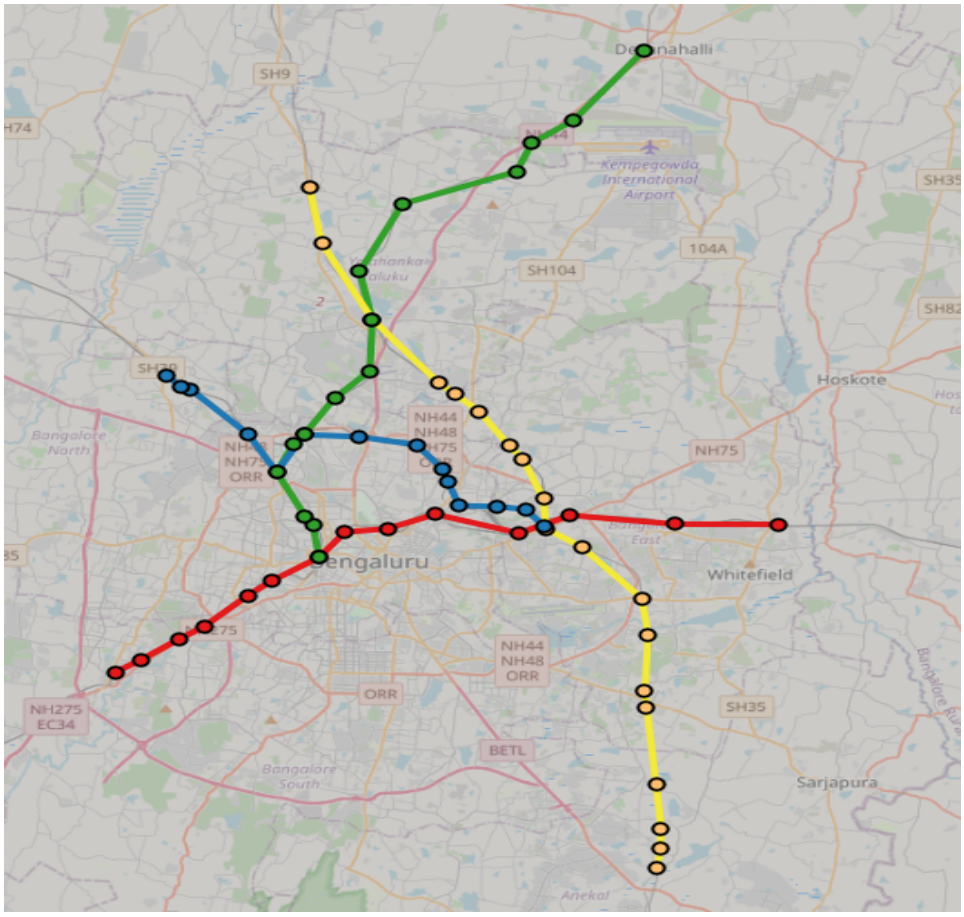


Figure 2.5: Bengaluru suburban rail network

Route IDs in the rail system were assigned based on a combination of the network abbreviation (R for rail), line name, and the first letter of the starting stop. For example, the route ID for the Mallige line from "Baiyyappanahalli" to "Chikkabanavara" is denoted as R_MB, with "B" representing the first letter of the starting stop. In the case of routes in the opposite direction, the route ID is determined by the network abbreviation, line name, and the first letter of the starting station in that specific direction. This information is stored within the routes.txt file.

To generate the trips.txt file, the count of trips for each route was obtained from the time schedule available in the RITES DPR (https://kride.in/wp-content/uploads/2021/09/Detailed-Project-Report-BSRP.pdf) for suburban rail. The schedule provided the frequency of trains on each route, which can vary throughout the day. By utilizing this information, the total number of trips made on each route in a single day was determined.

Trip IDs were assigned using the network abbreviation (R for rail), route ID, and a corresponding number that spanned the total number of trips for that specific route. For instance, the "Parijaata" line from "Kengeri" to "Whitefield" comprised 38 trips, with trip IDs ranging from R_PK_1 to R_PK_38. Similarly, trip IDs for other routes were assigned based on the respective number of trips for each route.

The arrival time at each stop along a route can be calculated by utilizing the starting time, trip frequency, and the time difference between consecutive stops. This data, along with information for all the trips, is stored in the *stoptimes.txt* file. The time difference between consecutive stops was determined using the interstation distance information provided in the RITES DPR (Detailed Project Report) and the average rail speed of 33 km/hr, as stated in the DPR. This allowed for the calculation of arrival times at each stop for the respective number of trips on each route.

Similarly to the metro network, the transfer.txt file for the suburban rail system was created to store information about the source and destination rail stops, along with a minimum transfer time of 2 minutes. This file encompasses all the transfer stations within the suburban rail network.

The fare files for the suburban rail network contain the source and destination rail stops, along with their respective fare IDs, in the *fare rule.txt* file. The fare amount can be obtained from the *fare attribute.txt* file by using the corresponding fare ID. For the suburban rail network, fares have been calculated based on the fare rules specified in the DPR (Detailed Project Report), where the fare is determined by the distance traveled in kilometers along the rail network. For instance, if the distance falls within the range of 18 to 21 km, the fare is set at 43 rupees.

2.3 Transit Network

The GTFS dataset for the Bengaluru transit network utilized in this project was developed by Dr.Tarun Rambha and Mr.Prateek Agarwal. It includes all the essential files such as stops.txt, routes.txt, trips.txt, stoptimes.txt, transfers.txt, fare rule.txt, and fare attribute.txt. The transit network comprises a total of 5638 routes, 28216 trips, 8254 bus stops, and 16723 transfers.

2.4 Combined Datasets

To create combined datasets for the different Bengaluru transportation networks, namely transit, metro, and suburban rail GTFS, special care was taken to avoid any conflicts with stop IDs, trip IDs, and route IDs across the datasets. This was resolved by adding a prefix to the IDs, representing the corresponding network abbreviation. For example, stop IDs for the metro network were prefixed with "M", suburban rail IDs with "R", and bus IDs with "B". This ensured uniqueness and avoided any ambiguity.

The process of combining the datasets involved appending the necessary files such as *stops.txt*, *routes.txt*, *trips.txt*, and *stoptimes.txt*. These files were consolidated to create a unified representation of the entire transportation network in Bengaluru.

However, the combined datasets were not complete without considering transfers between different modes of transportation. A crucial file in this regard was the *transfer.txt* file. Initially, this file consisted of the appended transfer data from different modes within the same network. It captured the transfers between stops within the transit, metro, or suburban rail networks.

To enable transfers between different networks, specific distance limits were imposed. For instance, transfers between the metro and rail networks (and vice versa) were allowed only within a distance of 4 kilometers. This distance limit ensured that the transfer between these networks was feasible and convenient for commuters. Similarly, transfers between the metro/bus and rail/bus networks (and vice versa) were restricted to a shorter distance of 0.5 kilometers.

The transfer distances between any two stops were determined using the Dijkstra's shortest path algorithm. This algorithm allowed for finding the shortest path between stops and facilitated the identification of valid transfers based on the specified distance limits and later these distances is converted to seconds using the average walking speed of 5km/hour.

As a result, the *transfer.txt* file included transfers not only within the same mode of transportation but also transfers between different modes where the road distance between the stops fell within the specified limits. This comprehensive transfer file played a vital role in ensuring transfers from one network type to other.

Network Type	Number of Routes	Number of Trips	Number of Stops	Number of Transfers
Existing Metro	4	496	52	2
Upcoming Metro	16	2096	196	40
Suburban Rail	8	414	62	10
Bus	5638	28216	8254	16723
Metro_bus	5654	30312	8450	18079
Rail_Bus	5646	28630	8316	16934
Metro_Rail	24	2510	258	751
Bus_Metro_Rail	5662	30726	8512	18992

Table 2.1: Summary of all the GTFS datasets

Chapter 3

Routing Algorithms

For a given source and destination finding the optimal journey in the transportation network is called as routing. The routing algorithms can be broadly classified in two categories personal mobility routing and public transit routing, where the main aim is find a route which can lead us to the destination as soon as possible. The major difference between the two class problems is time dependency of the transportation network. For personal mobility routing we have time independent network whereas for public mobility routing it is a time dependent (TD) network where the transit runs on the network following a predetermined scheduled. In PMR we will get only one best route that is shortest path between the source and destination node where as in PTR we may get multiple optimal journeys as in PTR it is possible to reach the destination earlier by taking more number of transfers.

3.1 Dijkstra's algorithm

It is one of the popular algorithm for PMR where it consider the transportation network as a graph and solve for the shortest path problem in that graph. It can be used to find the shortest path between two nodes in a weighted graph, where the weights represent the cost of traversing a particular edge it could be time or distance between the nodes.

Consider G is the graph, V is the set of nodes and E is the set of edges. $d(v)$ is the shortest distance from the source node to a node v and $w(u, v)$ denotes the weight

between the node u and v it could be time or distance.

Algorithm 1 Dijkstra's algorithm

1: Let $G = (V, E)$ be a weighted graph, where V is the set of nodes and E is the set of edges.
2: Let $d(v)$ be the shortest distance from the source node s to node v.
3: Let $b(v)$ be the label to backtrack the shortest path.
4: Initialize $d(s) = 0$ and $d(v) = \infty, \quad \forall\, v \in V \setminus \{s\}$
5: Initialise $b(s) = s$ and $b(v) = -1, \quad \forall\, v \in V \setminus \{s\}$.
6: Create a priority queue Q and add all nodes in V to it.
7: **while** $Q \neq \emptyset$ **do**
8: Remove a node u where $u = \arg\min_{u \in Q} d(u)$
9: **for** each neighbour v of u **do**
10: **if** $d(v) > d(u) + w(u, v)$ **then**
11: Update $d(v) = d(u) + w(u, v)$
12: Update $b(v) = u$
13: The shortest path time from s to any other node v is given by $d(v)$

The Algorithm 1 starts by initializing the distance of the source node to zero and the distance of all other nodes to infinity. Then it creates a priority queue and adds all the nodes in the graph to it. The algorithm then repeatedly removes the node with the smallest distance from the priority queue and relaxes all its edges. This process is repeated until the destination node is reached or the priority queue is empty.

The time complexity of Dijkstra's algorithm is $O(E + V \log V)$, where E is the number of edges and V is the number of nodes in the graph.

To undestand the working of the Dijkstra algoritm an example is shown in the figure 3.1.

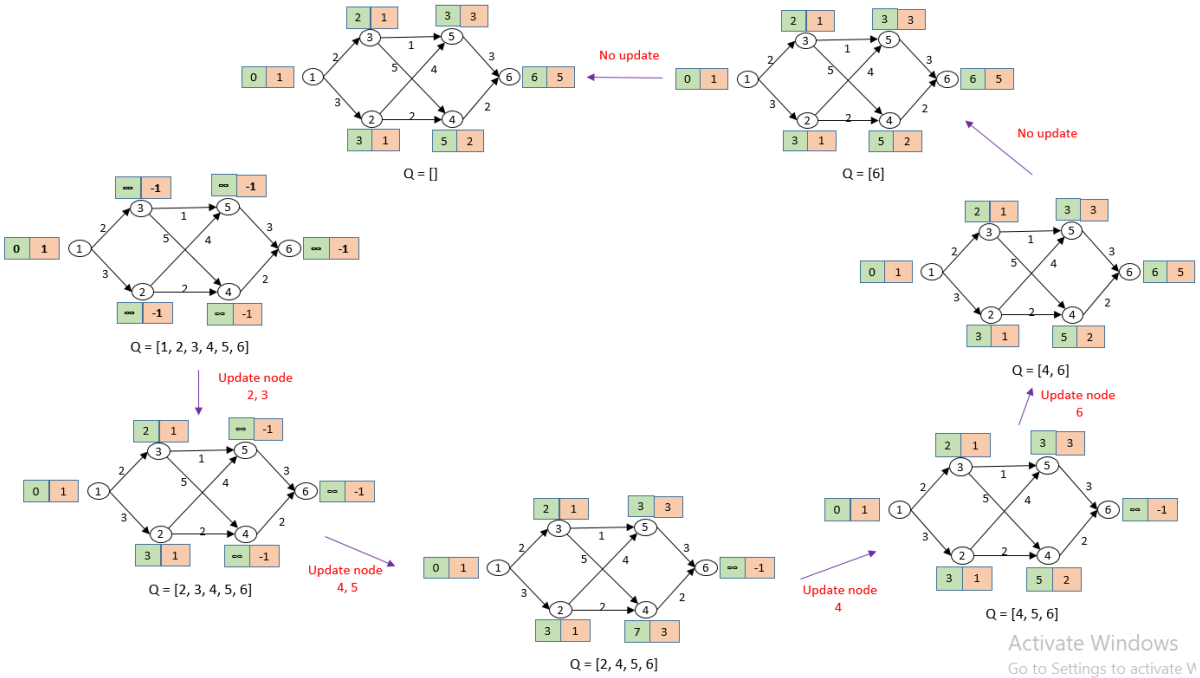


Figure 3.1: Example of Dijkstra's algorithm

Note: There are several techniques that can be used to speed up Dijkstra's algorithm, such as goal-directed search, landmark algorithm, and bi-directional Dijkstra. All these techniques can improve the time complexity of Dijkstra's algorithm, making it more efficient in certain situations. However, it's worth noting that the performance gain depends on the structure of the graph, the location of the source and destination nodes, and the specific implementation of the algorithm.

3.2 RAPTOR Algorithm

Public mobility routing algorithms works on the time dependent transportation network for finding the pareto optimal journeys between the source and destination stop with aim finding journeys through which destination can be reached as soon as possible but for transit network we can find a shorter journey with more number of transfers. RAPTOR (Round bAsed Public Transit Optimized Router) Delling et al. (2015) is a dynamic routing algorithm specifically designed for Public Transit Routing (PTR). It

minimizes the bicriteria problem involving travel time and the number of transfers and can incorporate other optimizing criteria such as the number of fare zones and cost. RAPTOR uses array and bag data structures to store the transit timetable in the form of trips and works in rounds. The k^{th} round computes arrival times at reachable stops using exactly $k-1$ transfers. Unlike other algorithms like Timed Dijkstra that treat this problem as a graph problem, RAPTOR directly works on the timetable information. It generates a set of Pareto optimal journeys for a given source, destination, departure time, and a limit on the maximum number of transfers allowed. It is more efficient and less computationally heavy compared to other algorithms because it scans a set of routes rather than the complete network in each round. RAPTOR continues until no better arrival time is found at any stop, at which point a shortest path tree is complete, with the additional bonus of journeys with a longer travel time but a lower number of transfers.

3.2.1 Terminology

In a public transit network, a timetable is a collection of information that encompasses the stops V, routes R, trips T, and footpaths F within the network. A stop v is a point in the network where passengers can board or disembark a vehicle, such as a bus or metro station. The starting and ending stops of the journey are labeled as v_o and v_d respectively. A route r is a specific sequence of stops that a vehicle follows, with the i^{th} stop on the route designated as $r(i)$, and the total number of stops on a route represented as $len(r)$. A trip t is the movement of a vehicle along a designated route. A stop event refers to a trip's arrival or departure at a stop. The i^{th} stop of a trip is represented as $t(i)$ and the total number of stops on a trip is represented as $len(t)$. The route of a trip is represented as $r(t)$ and the arrival and departure times at the stop v of a trip are represented as $arr(t, v)$ and $dep(t, v)$ respectively. Trips on a route are assumed to not overlap with each other. A footpath (v_1, v_2) is a pedestrian connection between stops v_1 and v_2. The time required to travel between stops v_1 and v_2 is represented as $f(v_1, v_2)$. Footpaths are assumed to be transitively closed, meaning that if (v_1, v_2) and (v_2, v_3) are footpaths, then (v_1, v_3) is also a footpath. Additionally, the time required to travel between stops is assumed to follow the triangle inequality, meaning that $f(v_1, v_3) \leq f(v_1, v_2) + f(v_2, v_3)$. A journey y is a sequence of trips and footpaths in the order they are traversed. Journeys are associated

with various optimization criteria, and a journey y_1 is considered to dominate another journey y_2 if all the criteria of y_1 are no worse than those of y_2. The set of journeys y that are non-dominating with respect to each other is called Pareto-optimal set of journeys and is denoted by J.

Table 3.1: Terminology

Symbol	Description
v	stop
V	set of all stops
v_o	source stop
v_d	destination stop
(v_1, v_2)	footpath connection between stop v_1 and v_2
F	set of all footpath connections
$f(v_1, v_2)$	footpath walking time between stop v_1 and v_2
t	trip
T	set of all trips
$t(i)$	i^{th} stop of trip t
$r(t)$	corresponding route of a trip t
$arr(t, v)$	arrival time of trip t at stop v
$dep(t, v)$	departure time of trip t at stop v
l_t	total stops in trip t
r	route
R	set of all routes
$r(i)$	i^{th} stop of route r
l_r	total stops in route r
τ	departure time
K	maximum allowed transfers

3.2.2 Pseudocode

Algorithm 2 RAPTOR algorithm

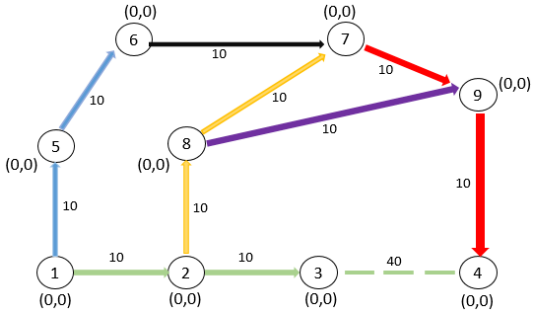
Input: GTFS, τ, v_o, v_d, K

Output: Optimal labels

1: $leb(k, v), star\_leb(v) = \infty, \infty, \quad \forall\, v \in V \setminus \{v_o\}, \forall\, k \leq K$
2: $leb(0, v_s), star\_leb(v_s) = \tau, \tau$
3: $marked\_stops = \{v_s\}$
4: **for** k = 1 ... K **do**
5: $Q = \{\}$
6: **for** $v \in marked\_stops$ **do**
7: Remove v from marked_stops
8: **for** route r serving stop v **do**
9: **if** $(r, v') \in Q$ and $arg(r(v)) < arg(r(v'))$ **then**
10: # stop v comes prior to v' in the route stops sequence
11: $Q = Q - (r, v')$
12: $Q = Q \cup \{(r, v)\}$
13: **for** $(r, v) \in Q$ **do**
14: $t = \perp$
15: **for** stop u starting from v in route r **do**
16: **if** $t \neq \perp$ and $arr(t, u) < min(star\_leb(u), star\_leb(v_d))$ **then**
17: $leb(k, u), star\_leb(u) = arr(t, u), arr(t, u)$
18: $marked\_stop = marked\_stop \cup \{u\}$
19: **if** $t == \perp$ or $leb(k-1, u) \leq arr(t, u)$ **then**
20: t = earliest trip from stop u along route r after time $leb(k-1, u)$
21: **for** $v \in marked\_stop$ **do**
22: **for** stop u where $(v, u) \in F$ **do**
23: **if** $leb(k, u) > leb(k, v) + f(v, u)$ **then**
24: $leb(k, u) = leb(k, v) + f(v, u)$
25: $star\_leb(u) = leb(k, v) + f(v, u)$
26: $marked\_stop = marked\_stop \cup \{u\}$
27: **if** $marked\_stop == \{\}$ **then**
28: **break**

RAPTOR algorithm works in round and in each round it scan set of routes which originate from the source or can be reached using transfer in $k-1$ round, hence we store best label for each stop for each round called label denoted by *leb*. Also to speed up the algorithm we do local and target pruning and mark only those stop which have label better than the best label across all the $k-1$ rounds for this we maintain a label which store the best label across all $k-1$ round called star label denoted by *star_leb* for each stop in the network. Initially *leb* and *star_leb* store infinity value for all the stops. Let the maximum number of round allowed is K that means $K-1$ transfers.

3.2.3 Example



Schedules

t1, 08:00	t2, 08:10	t3, 08:20	t4, 08:30
t5, 08:00	t6, 08:10	t7, 08:20	t8, 08:30
t9, 08:00	t10, 08:10	t11, 08:20	t12, 08:30
t13, 08:00	t14, 08:10	t15, 08:20	t16, 08:30
t17, 08:00	t18, 08:10	t19, 08:20	t20, 08:30
t21, 08:00	t22, 08:10	t23, 08:20	t24, 08:30

Figure 3.2: RAPTOR dummy network with schedule

Let understand the working of RAPTOR algorithm more clearly with the dummy network shown in the above Fig 3.2,3.3 the time duration in minute between the two nodes are shown on the edges of the network and the corresponding timetable for different trips are shown in the schedule table. In this node 1 is the source stop and node 4 is the destination stop, the footpath connection between the node 3 and 4 is shown using the dash line with corresponding walking time of 30 min. departure time from source is 08:00(τ) and labels of each node is shown in round bracket for each stop.

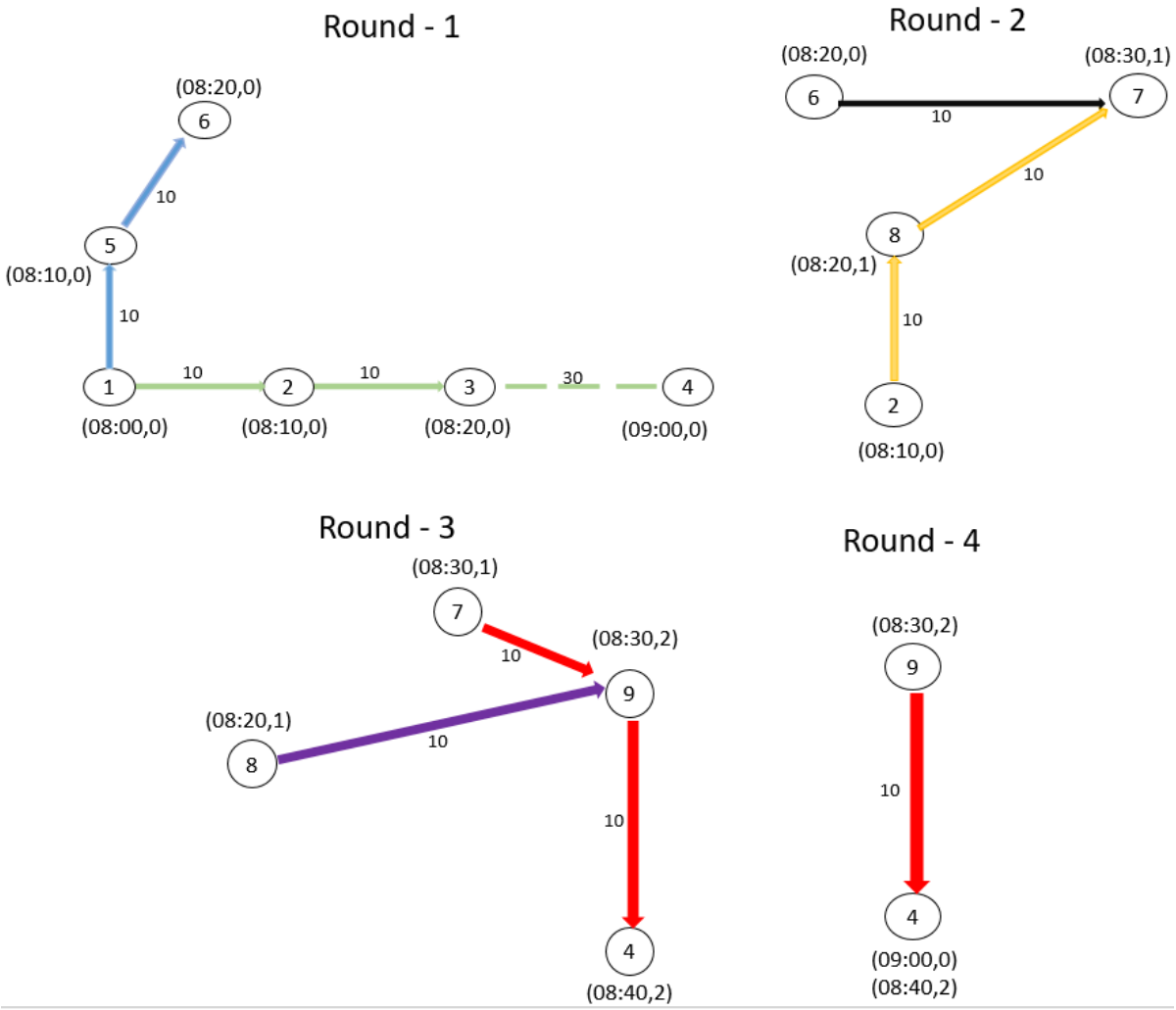


Figure 3.3: Different rounds in RAPTOR algorithm

In round 1 board trip $t1$ and $t5$ in green and blue route respectively and update the labels of the descending stops in that route and check for any footpath connection available for these stops, if present update their label as well, by doing so we can reach destination at 09:00 with no transfer. For round 2 traverse the yellow and black route by boarding the earliest trip available from stop 2 and 6 along the corresponding route after the time reached in previous round for stop 2 and 6 hence board trip $t10$ and $t15$ for yellow and black route respectively. Similarly for round 3 and 4 are shown in the figure above, note that label will be updated only if we can reach a stop earlier with more transfers, for example label of stop 9 is not been updated in round 3 along the red route as we can reach stop 9 using purple route in lesser time in round 2 itself. After

completing all the rounds we can find that optimal label for destination node, which are (09:00, 0 transfer) and (08:40, 2 transfer); hence two pareto optimal journeys are available in this example network.

Note: In the RAPTOR algorithm, which is schedule-based algorithm, the earliest trip that can be boarded at any stop takes into account the waiting time for the transit to arrive at that stop. The waiting time at any stop depend on the departure time of the passenger and the existing schedule of trips. For planning of transit network these individual level waiting time do not matter, hence for a frequency based model we need to make modification in RAPTOR algorithm, which results in tweaked raptor explained in next section.

3.3 Tweaked RAPTOR Algorithm

Tweaked RAPTOR This algorithm assume that at any stop at any point of time we will find a trip on that route. Which means no waiting time. Hence we need to change the function which gives us the earliest trip that can be board from a stop along the route after a certain time that is line number 20 in the pseudo code for **RAPTOR**. This function now give an artificial trip that can be boarded instantaneously from a stop after the time it took to reach that stop in previous round, but trip maintain the same time gaps between the two stops as it is there in actual schedule. Hence no waiting time at any stop. Else everything is same as that explained in the RAPTOR algorithm. Hence for skims generation of journeys we will be using tweaked RAPTOR for finding the pareto optimal set of journeys between the source and destination stop.

Chapter 4

Methodology

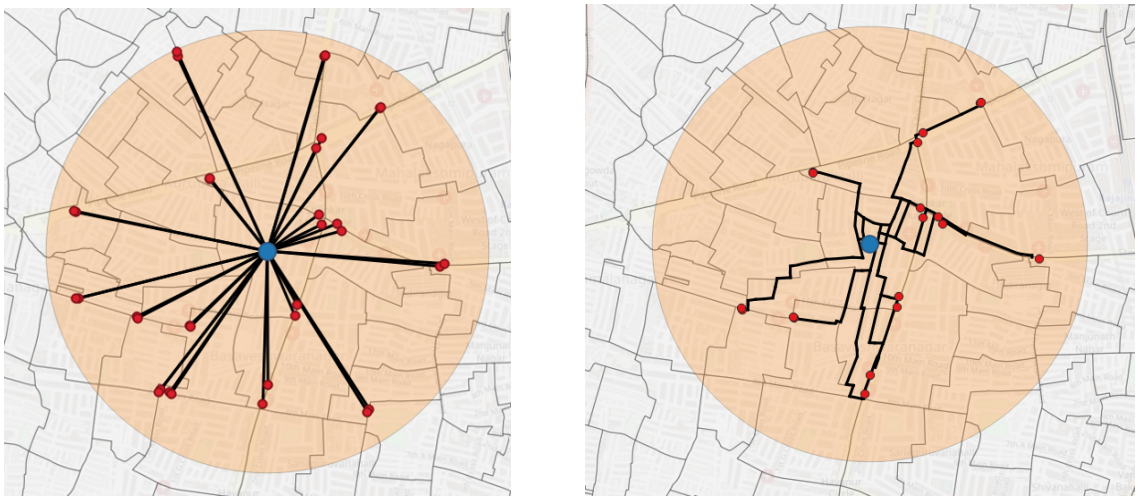
4.1 Mapping of zone centroid to nearest stations:

The mapping of zone centroids to the nearest metro, suburban rail, or transit stations is achieved by utilizing the Dijkstra algorithm. This algorithm helps identify the nearest boarding stations within a specified access radius for a given network type (e.g., Transit, Metro, and Suburban). To facilitate this process, a data structure is created to store all stations in ascending order of their Dijkstra distance for each zone centroid.

For each network type, a specific access radius is defined. For instance, a 2km access radius is set for the Transit network, while a 4km radius is considered for the Metro network. These access distances are converted into time using an assumed average walking speed of 5km/hour.

Additionally, the Python code incorporates the option for mapping based on the haversine distance, giving users the flexibility to choose their preferred approach. Similar considerations are made for the egres distance on the destination side, which is defined for each network type.

By implementing these procedures, the mapping process effectively determines the closest stations to zone centroids, facilitating convenient access to metro, suburban rail, or transit systems.



(a) Using Haversine formula.

(b) Using Dijkstra's algorithm.

Figure 4.1: Stops within a certain distance threshold from the centroid.

Network Type	Access/Egress Mode	Access/Egress Radius (Km)	Access/Egress Speed (km/hr)
Existing Metro	walk	4	5
Upcoming Metro	walk	4	5
Suburban Rail	walk	5	5
Bus	walk	2	5
Metro_bus	walk	2	5
Rail_Bus	walk	2	5
Metro_Rail	walk	4.5	5
Bus_Metro_Rail	walk	2	5

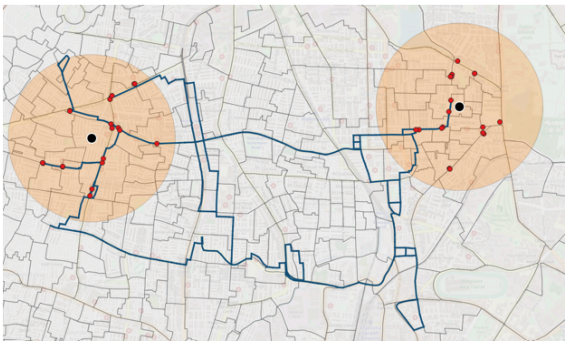
Table 4.1: Access and Egress radius for different network type.

4.2 Run the tweaked RAPTOR algorithm:

To determine the optimal journeys for a given source zone centroid, the tweaked RAPTOR algorithm is executed from all the boarding stations within the specified access radius, as determined in the previous step. This modified algorithm is an adaptation of the standard RAPTOR algorithm, but with the assumption that a trip on any route will

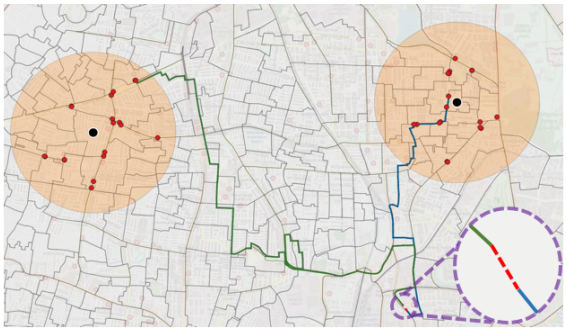
always be available at any stop and at any given time, eliminating the need for waiting time.

During this process, the tweaked RAPTOR algorithm is applied without explicitly specifying the destination alighting station. Instead, it is executed for all possible routes within the defined maximum number of transfers. As a result, only journeys that have a destination alighting station located within the egress radius for the destination zone are selected to proceed to the next step, where the relevant skim for these journeys is extracted and processed.

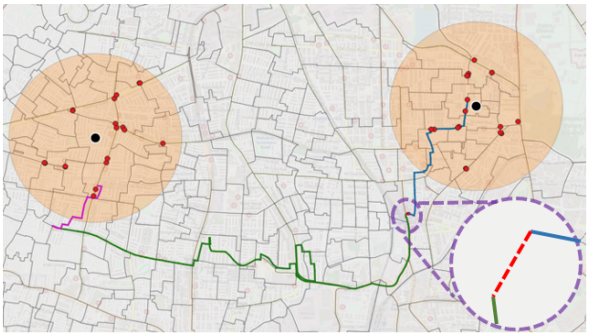


(a) Links belonging to the optimal journeys with 0, 1, and 2 transfers

(b) Links belonging to an optimal journey with 0 transfers (i.e., a direct journey)



(c) Links belonging to optimal journeys with 1 transfer



(d) Links belonging to optimal journeys with 2 transfers

Figure 4.2: Visualization of journeys involved in generating skims between an OD pair. Grey boundaries represent TAZs. Black nodes represent origin and destination zone centroids. Stops within the access/egress mode radius are indicated in red. Each leg of the journeys with multiple transfers is indicated by a different color. The dashed-red line represents walking connections.

4.3 Calculation of skims:

In the next step, the obtained journeys from step two undergo a post-processing operation to calculate relevant skims. These skims include information such as the source zone, destination zone, source stop, destination stop, number of transfers, in-vehicle travel time (IVTT), cost, walking time, access time, egress time, access distance, and egress distance.

In the case of a combined dataset, special attention is given to fare calculation. For bus journeys, the fare is determined by summing up the fares for each route included in the journey. On the other hand, for metro and suburban rail journeys, the fare is based on the specific source and destination metro or suburban rail stations, rather than the transfer stations.

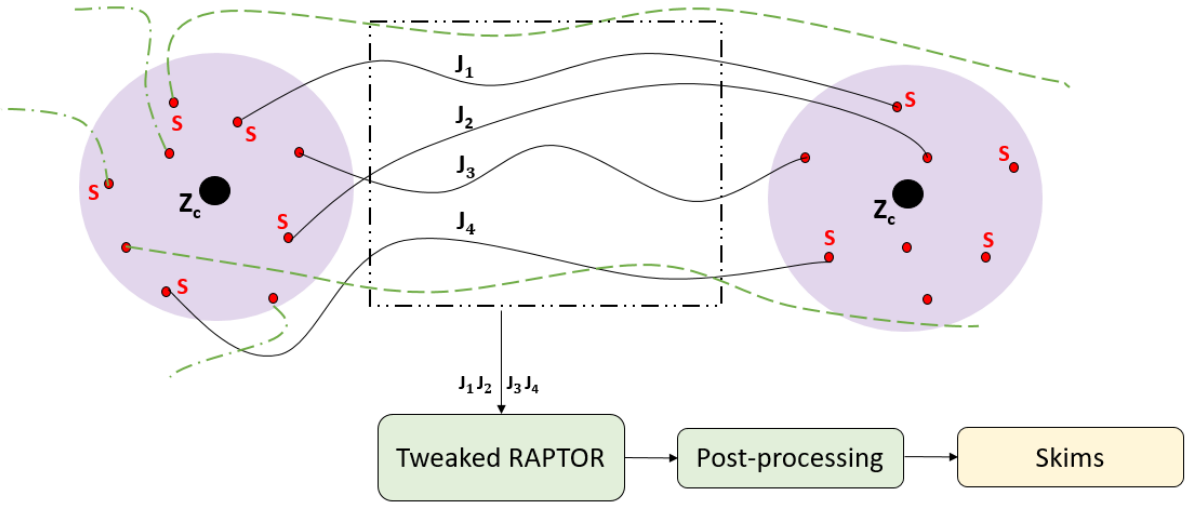


Figure 4.3: Visualization of Methodology. Zone centroid Z_c, Boarding stations within access and egress radius S, Journeys reaching destination alighting stations J, and dashed green line for journeys not passing through destination alighting stations.

4.4 Updating transit skim:

The skims initially used in the four-step model undergo updates following the traffic assignment step. The assigned traffic flows modify the transit network's schedule, which subsequently generates new skims based on the updated transit schedule. This section

focuses on determining the new skim attributes between the source and destination zones. To accomplish this, a Python code has been developed, which utilizes the link travel times obtained from the traffic assignment step. The code updates the arrival and departure times for buses along their routes. It identifies the network links between the two bus stops and adjusts the arrival time at the head stop based on the cumulative link travel time. The resulting updated schedule is then employed to find the best/optimal route between two zone centroids, consequently calculating new skims for those journeys.

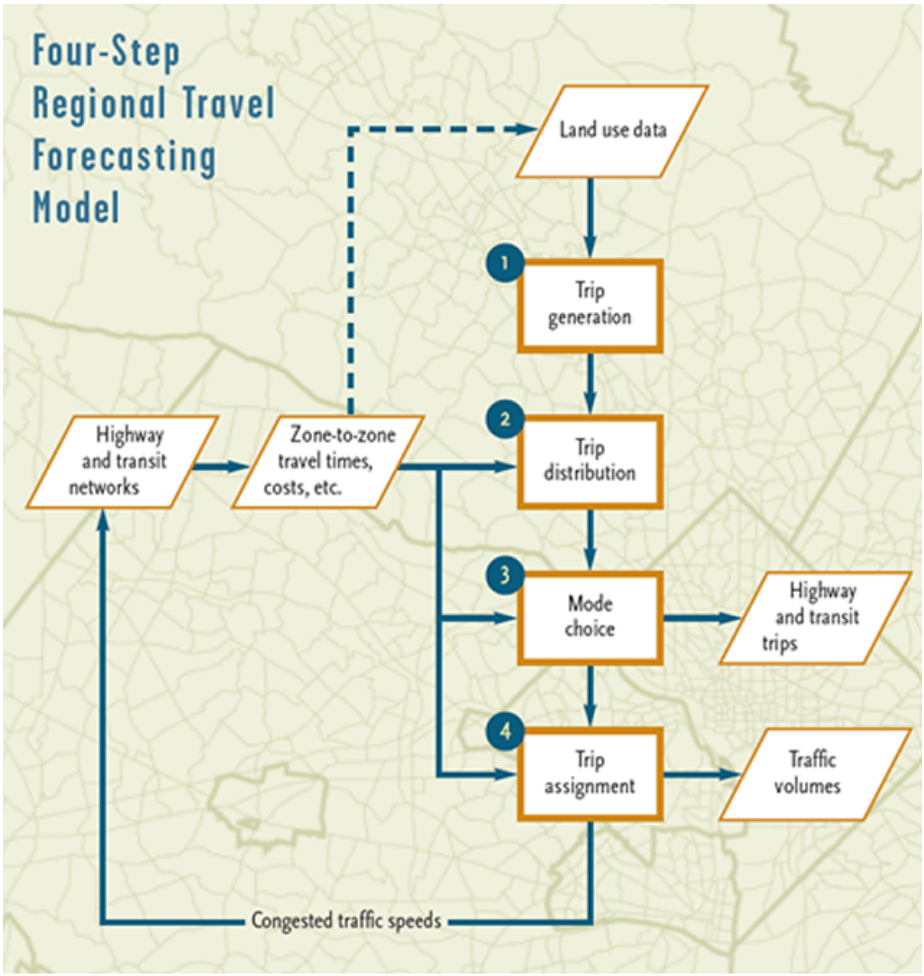


Figure 4.4: Illustration of the Four-step travel model.(source:https://www.mwcog.org/transportation/data-and-tools/modeling/four-step-model)

Chapter 5

Results and Conclusion

5.1 Skim matrices

The research aimed to generate skims for different Bengaluru network for different origin-destination zone pairs along an optimized route between source and destination zones. The process involved mapping of the nearest boarding stations to a given source and destination zone centroid using the actual distance, running the tweaked RAPTOR algorithm to find the Pareto-optimal set of journeys, and calculating skims such as access time, egress time, access distance, egress distance, number of transfers, in-vehicle travel time, walking time, and journey fare/cost as a post-processing operation and skim matrices for all eight network is given below.

Source zone	Destination Zone	Source Metro Station	Destination metro Station	Number of transfers	IVTT (minute)	Metro fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
236	1358	M_G_5	M_P_26	1	129	42	2	42.22	35.52	3.52	2.96
501	192	M_G_10	M_G_5	0	72.6	20	0	46.01	45.34	3.83	3.78
625	225	M_G_8	M_G_4	0	75	18	0	37.5	43.65	3.12	3.64
796	518	M_P_15	M_G_11	1	81	40	2	26.34	30.22	2.19	2.52
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.1: Skims for existing Bengaluru metro

Source zone	Destination Zone	Source Metro Station	Destination metro Station	Number of transfers	IVTT (minute)	Metro fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
236	1358	M_G_5	M_P_23	0	125.4	35	2	42.22	0.78	3.52	0.06
501	192	M_G_10	M_G_6	0	70.8	18	0	46.01	45.58	3.83	3.8
625	1	M_O_17	M_B_27	1	173.4	48	2	37.5	16.54	3.12	1.38
796	518	M_P_16	M_R_2	1	172.8	37	2	42.38	42.33	3.53	3.53
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.2: Skims for future Bengaluru metro

Source zone	Destination Zone	Source Rail Station	Destination Rail Station	Number of transfers	IVTT (minute)	Rail fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
236	1358	R_M_12	R_S_4	1	22.8	20	2	46.45	51.02	3.87	4.25
501	192	R_M_11	R_M_13	0	58.2	20	0	55.22	0.67	4.6	0.06
625	2	R_S_3	R_S_10	0	51.6	37	0	51.43	40.37	4.29	3.36
796	349	R_K_10	R_K_14	0	84.6	27	0	58.74	48.15	4.89	4.01
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.3: Skims for Bengaluru suburban rail network

Source zone	Destination Zone	Source Bus Stop	Destination Bus Stop	Number of transfers	IVTT (minute)	Bus fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
236	1358	876	1343	1	55.2	30	1.49	22.5	21.94	1.88	1.83
501	192	5757	4438	1	39.6	15	2.31	7.62	9.46	0.64	0.79
625	4	1366	8398	0	91.2	60	0	8.34	0.38	0.7	0.03
796	349	837	7540	0	133.2	55	0	17.93	0.29	1.49	0.02
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.4: Skims for Bengaluru transit

Source zone	Destination Zone	Source Station	Destination Station	Number of transfers	IVTT (minute)	Journey fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
236	1358	R_M_12	M_G_13	1	21.6	20	42.25	46.45	10.61	3.87	0.88
1839	1	R_P_6	M_B_27	1	67.2	43	42.64	23.46	16.54	1.96	1.38
19	7	R_S_9	M_B_27	1	147.6	68	1.3	48.43	36.18	4.04	3.02
2393	1086	M_Pi_4	R_M_5	1	35.4	36	19.88	36.04	37.54	3.0	3.13
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.5: Skims for Bengaluru metro & rail combined dataset

.

Source zone	Destination Zone	Source Station	Destination Station	Number of transfers	IVTT (minute)	Journey fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
14	1	5018	10678	1	68.4	15	0	4.09	2.81	0.34	0.23
61	1256	5801	1233	1	85.2	35	1.19	19.92	12.18	1.66	1.01
124	4	4913	8376	1	43.2	25	0.64	0.84	17.52	0.07	1.46
88	308	4994	10253	1	75.0	45	1.56	20.76	23.79	1.73	1.98
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.6: Skims for Bengaluru bus & rail combined dataset

.

Source zone	Destination Zone	Source Station	Destination Station	Number of transfers	IVTT (minute)	Journey fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
57	228	5198	M_G_5	1	71.4	40	2.27	5.15	14.88	0.43	1.24
141	1	904	1656	1	34.8	25	1.15	5.98	7.43	0.5	0.62
158	1863	2454	M_G_18	1	96.6	40	3.18	20.87	17.37	1.74	1.45
14	1009	5018	10594	0	58.2	25	9.38	4.09	20.59	0.34	1.72
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.7: Skims for Bengaluru bus & metro combined dataset

.

Source zone	Destination Zone	Source Station	Destination Station	Number of transfers	IVTT (minute)	Journey fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
57	228	5198	873	1	46.2	40	5.57	5.15	18.75	0.43	1.56
331	19	4972	14693	1	52.2	20	2.75	19.36	0.25	1.61	0.02
741	3	646	M_B_26	1	46.8	40	76.73	21.18	11.34	1.76	0.94
965	113	6608	M_R_1	1	123.6	41	2.0	2.75	23.27	0.23	1.94
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.8: Skims for Bengaluru bur, rail & metro combined dataset

.

5.2 Zonal heat maps

To visualize the percentage change in skims resulting from a change in network type, zonal heat maps can be created. These heat maps demonstrate how attributes vary between the same source and destination zone due to the network modification.

By analyzing the zonal heat map, transportation planners and analysts can easily identify areas or zones where the network type change has the most significant impact on the skims. This information is crucial for assessing the potential benefits or drawbacks of modifying the network and understanding how it affects travel times, distances, or costs between specific zones.

For instance heat maps displayed below illustrate the zonal heat map for the source zone centroid 256, showcasing the improvements in In-Vehicle Travel Time (IVTT) over the existing and upcoming metro networks. Additionally, another heat map represents the cost improvements for the bus and bus_metro_rail combined GTFS networks, depicting the journey to all other zones in fig: 5.9 and fig: 5.10.

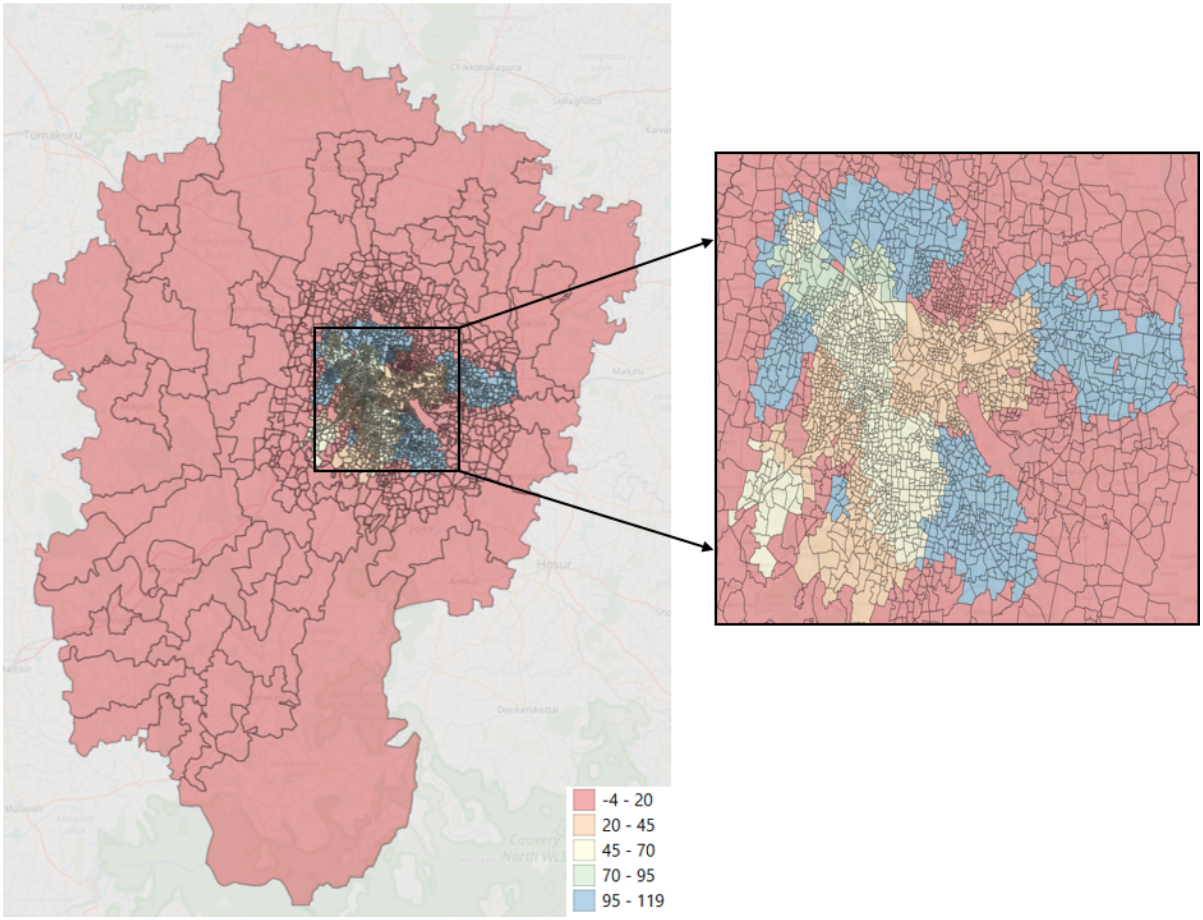


Figure 5.9: A heat map showing the percentage change in in-vehicle travel time (IVTT) from Zone 256 to all other zones between the existing and upcoming metro networks.

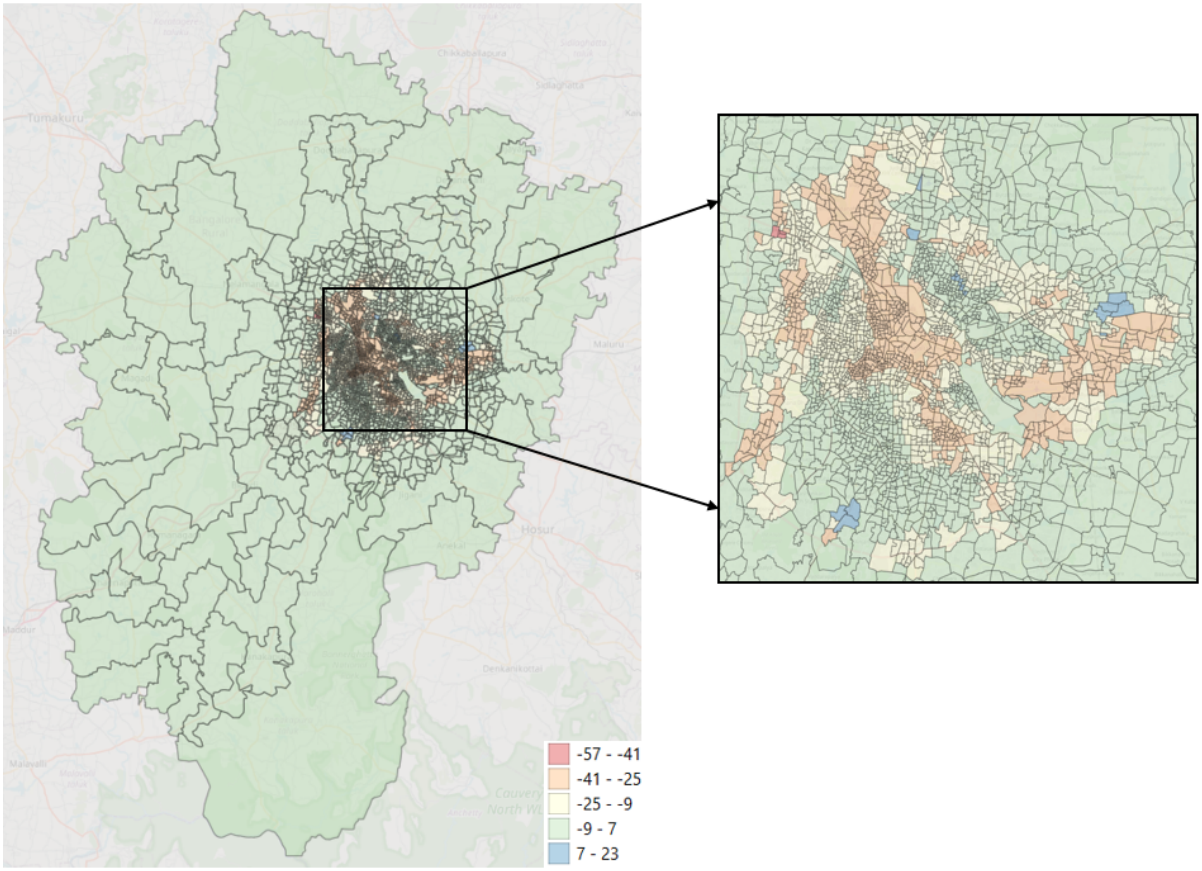


Figure 5.10: A heat map showing the percentage change in journey fare from Zone 256 to all other zones between bus and bus_metro_rail combined network.

Note: To handle the large amount of skim matrix files, which occupy approximately 250GB of data, and to enhance the visualization of the skim matrix for various network types, source zone centroids, and destination zones, a dedicated website is being developed. This website will provide an interactive platform for exploring and analyzing the skim matrix data. Additionally, multiple heat maps can be generated for different source zone centroids to visualize the relevant information in a more user-friendly manner.

5.3 Updated skim matrix for transit

Currently, the link travel times have been randomly generated since the actual data was not available. However, provision has been made to incorporate link travel times obtained from the traffic assignment step.

In the current implementation, the randomly generated link travel times serve as a temporary placeholder to test and demonstrate the functionality of the system. Once the actual data becomes available from the traffic assignment step, it can be seamlessly integrated into the system.

Source zone	Destination Zone	Source Station	Destination Station	Number of transfers	IVTT (minute)	Journey fare (rupees)	Walking time (minute)	Access time (minute)	Egress time (minute)	Access distance (Km)	Egress distance (Km)
57	228	8460	7018	1	148.3	25	0.07	15.86	20.71	1.32	1.73
216	13	10464	1878	1	180.08	35	4.77	8.17	17.29	0.68	1.44
791	1	411	10678	1	348.63	30	0	8.81	2.81	0.73	0.23
1	3	10678	1852	0	16.88	5	0.82	2.81	11.79	0.23	0.98
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.

Figure 5.11: Updated skim matrix for transit

5.4 Conclusion

The data obtained from the skims plays a vital role in shaping future public transport planning decisions in Bangalore. It serves various purposes, including evaluating the existing state of the public transport system, identifying areas that require improvements, and proposing recommendations for enhancing the transportation network. For instance, if the skims reveal longer access times to metro stations than anticipated, it suggests a need for better pedestrian infrastructure or improved bus connectivity to those stations.

Furthermore, the zonal heat maps serve as valuable tools for analysts and decision-makers. They provide a visual representation of the impact that network changes have on travel attributes between specific zones. This enables a quick identification of areas where modifications to the network have a significant influence. The heat maps facilitate the evaluation of potential benefits or disadvantages associated with altering the network

type.

Additionally, network schedules are regularly updated to reflect the current state of network congestion, enabling the generation of new skims that accurately represent the actual network conditions. These updated skims are then incorporated back into the four-step model in an iterative manner. This iterative process ensures that the model adapts to changes in the transportation system and provides reliable results for decision-making purposes.

Bibliography

Carlo Giacomo Prato Marie Karen Anderson, Otto Anker Nielsen. Multimodal route choice models of public transport passengers in the greater copenhagen area. *EURO Journal on Transportation and Logistics*, 6(1):1–24, 2014. 2

Mark R. Wardman Pedro A.L. Abrantes. Meta-analysis of uk values of travel time: An update. *Transportation Research Part A: Policy and Practice*, 45(1):1–17, 2011. 3

Zhuangbin Shi Mingwei He Yang Liu, Tao Feng. Understanding the route choice behaviour of metro-bikeshare users. *Transportation Research Part A: Policy and Practice*, 166(1): 460–475, 2022. 3

Joseph N. Prashker Erel Avineri. Sensitivity to travel time variability: Travelers' learning perspective. *Transportation Research Part C: Emerging Technologies*, 13(1):157–183, 2005. 3

Alireza Khani Benjamin J. Tomhave. Refined choice set generation and the investigation of multi-criteria transit route choice behavior. *Transportation Research Part A: Policy and Practice*, 155(1):484–500, 2022. 4

Shlomo Bekhor Nurit Oliker. A frequency based transit assignment model that considers online information and strict capacity constraints. *EURO Journal on Transportation and Logistics*, 9(1):484–500, 2020. 4

Crisalli U. Nuzzolo, A. The schedule-based approach in dynamic transit modelling: A general overview. *Springer Science+Business Media New York 2004*, 28(1):1–24, 2004. 5

Elenna Dugundji Thomas Koch, Luk Knapen. Door-to-door transit accessibility using pareto optimal range queries. *Procedia Computer Science*, 170(1):107–114, 2020. 5

Daniel Delling, Thomas Pajor, and Renato F Werneck. Round-Based Public Transit Routing. *Transportation Science*, 49(3):591–604, 2015. DOI: https://doi.org/10.1287/trsc.2014.0534. 18