2022

GTFS2GMNS Users' Guide

Contributors

Fang (Alicia) Tang
fangt@asu.edu

Xin (Bruce) Wu
xinwu3@asu.edu

Xianting (Alice) Huang
xiantinh@andrew.cmu.edu

Cafer Avci
cafer.avci@aalto.fi

Taehooie Kim
taehooie@urbansim.com

Xuesong (Simon) Zhou
xzhou74@asu.edu

Abstract

gtfs2gmns is an open-source Python code that enables users to conveniently construct any multi-modal transit network from General Transit Feed Specification (GTFS). With the clean Python code, users can obtain and model multi-modal, including bus, metro, rail, etc. networks for any agency and multi-agencies in the world, and output networks to csv files in the General Modeling Network Specification (GMNS) format for seamless data sharing and research collaboration. gtfs2gmns mainly focuses on providing researchers and practitioners with flexible, standard, and ready-to-use multi-modal transit networks to facilitate diverse research challenges and applications in traffic modeling.

Contents

Glossary

Term	Description
Analysis time period	A planning time interval determined by users. It can be 6 AM - 9 AM, also it can be 6 AM - 12 PM.
Stop	The bus stop
Station	The metro/ rail station
Trip	The space-time trajectory of a transit line
Route	All the time-dependent trips connect between an origin station and a destination station over a whole analysis time period.
Directed route	A route with a certain direction
Directed route stop	The virtual stop along with a directed route
Service	All the routes over a whole analysis time period
Frequency	The number of the trips for each route over a whole time period. All this frequency is going to be used in the traffic assignment.
FFTT	Free-flow travel time of a link, whose unit is minutes
Traffic assignment	Assign traffic volume on the trips (Principle: Long-distance passengers are loaded first; Fast trains are selected first)

1 INTRODUCTION

In the United States and many parts of the world, the increasing rate of surface traffic congestion is outpacing the available roadway infrastructure in urban areas. Agencies and governments are pursuing the public transit system analysis and planning for the purpose of traffic flow improvement without constructing additional lanes. In transportation networks, different travel modes affect each other. In particular, there is an interactive dependency between congested highway travel time and the driving time of buses on the highway networks. That is, it is imperative to systematically address the complexity of the interrelationship affected by multi-modal systems, proposing a proper multi-modal network modeling to improve the quality of the public transit system analysis and planning program, including mobility optimization and traffic assignment.

This project recognizes the imperative to research multi-modal network modeling and to do so in the context of a collaborative decision-making process that allows analysts to fully utilize the existing static traffic assignment package, helping decision-makers explore how different modes will play out in different traffic environments and makes policy decisions more resilient. The following sections provide an overview meant to put the discussion of individual task activities into proper context.

1.1 Some Historical Perspective

Transport – how we are mobile – comes in various modes, including walking, cycling, cars, railways, ships, and airplanes. These modes are all deeply interrelated: the increasing use of one potentially reduces another. For example, cycling and walking were sidelined by the spread of private cars and the road infrastructure built to cater to them. In other cases, the relationship has been complementary. Rail freight stimulated road transport, for example, to and from railheads. Therefore, for users of our transport system, multi-modal travel is likely to grow in importance as new alternatives become embedded. In addition, integrated modeling approaches have been explored to understand the realistic behavioral patterns observed in users (i.e., seamless end-to-end journeys).

In the transportation network modeling field, scholars have pointed out the importance of developing a routable multi-modal network structure. However, because of the difficulty of defining the routable structures using some nodes and links through currently available integrated tools (e.g., PTV), the integrated analysis and optimization of the multi-modal network is still a challenge.

The generalized highway network is often represented by GMNS. However, the GMNS data format has technical difficulty in describing the basic data-sharing structure for public agencies. And the public transit network represented by the GTFS data format cannot be integrated with the highway network.

Another challenge is that the multi-modal problem here shows a highly complicated and dynamic environment. The complexity has different online and offline versions. And the dynamics have the day-to-day time of a day. Take Phoenix as an example. It contains 103 lines, 16,691 transit time-dependent trajectories, and 8,633,373 arrival or departure times from 6 AM to 9 PM on Jun 4, 2007. In the multi-dimensional system, it is also necessary to consider the rationality of the passengers (e.g., preferences and constraints).

In this context, a unified network structure will be the next major milestone in the evolution of mobility optimization. Specifically, the unified data structure will allow planners/modelers to connect transit data to a data hub through standard human and machine-readable formats with routable network structures.

1.2 Motivation

This paper aims to build an open unified data hub to clearly define a routable model structure for the standard traffic assignment utilities to analyze the different measures of transport supply, operating cost, and fare revenue, based on the assignment results. In addition, through the open unified data hub we demonstrate the connectivity to (1) the multi-modal and mobility service optimization, (2) operations and control models, (3) multiresolution modeling structure mainly for further microscopic transit simulation, (4) model calibration using multiple data sources across different resolutions, and (5) better calibrated and integrated models.

In general, transportation network modeling has offered analysis techniques that help society understand and optimize the movement of goods and people. In light of this, our team integrates various elements into the analysis, including the movement of goods, people, and the features of infrastructure such that this integration provides us with novel ways to improve transportation safety and efficiency while reducing vehicle emissions. It is also worth noting that the growth of capabilities of automation and communication technologies proposes new solutions to help us achieve the optimal goal mentioned, but we carefully handle the complexity of the dramatically increased dynamics of underlying transportation network structures to avoid creating additional challenges for engineers/planners/officials trying to manage transportation networks.

By doing so, an open unified data hub can create a window of opportunity to move toward a more integrated modeling platform. This has standardized nodes, links, and agents to represent the multi-modal transportation network model, which helps data format maintain consistency with standard traffic assignment utilities, such as path4gmns, DTALite, and other commercial tools using GMNS data specifications. Analysts can further perform the assignment and feed results from one model to another while establishing consistency between the model assumptions. Then, the traffic assignment results enable analysts to look at the physical network, the headway, the delay, and the possibilities of the transfers. Moreover, a unified data hub is able to map the static physical network representation and time-dependent public transit service network representation, which will ensure a fully integrated transport system.

1.3 Guiding Principles

This section discusses a unified data importing and conversion workflow and a prototype of organizing step-by-step data integration tools to allow effective modeling and analyses across different transportation modeling domains and scales. It starts with the guiding principles of the workflow design:

1. The base network generated by GTFS data should **integrate** with other networks, such as **highway networks** (freeway and arterial). Then the integrated network can generate OD demand which involves mode choice. In addition, analysts can connect **external points of interest (POI)** and land use data to the node, link, and zone layers in a GMNS-structured transportation network.
2. The **open modeling standard** will allow us to connect transit data to a **data hub** through standard **human and machine-readable formats** with routable network structures. We will use the GMNS as the building block and enhance the base GMNS specification to consider dynamic service frequency measures (e.g., 15-min headways during the morning peak hour). Open data specifications can precisely represent a multiresolution physical traffic system and support secure data sharing.
3. Simple **node-link** structure can **maintain consistency with standard traffic assignment** utilities. To be compatible with the central node-link structure in GMNS, we need to integrate both physical nodes and service nodes into a unified node structure. To be consistent with the overarching GMNS requirement and maintain the mapping with transit and highway layers, simple **node-link** structure, to reduce **coupling complexity** for many issues. In the transfer links and the boarding links, we need to consider how to convert the time-of-day frequency to the period-dependent waiting time and determine the period-based capacity considering possible congestion due to oversaturation.
4. Multi-modal and mobility service **optimization** can be built. The network representation with generalized routes enables us to conduct integrated optimization between different modes, each with different features like frequency during the peak and non-peak service hours, connections, and fleet capacity per vehicle.
5. Multiresolution modeling structure is mainly for further microscopic **transit simulation**. It is practically difficult to develop a one-size-fits-all network representation, so we mainly focus on the mesostructure. Macro, a static node-link structure, goes from zone centroids to stops. Links are representing aggregated route structures. Capacity is not considered separately for different routes/modes. The primary focus is on general accessibility rather than bottlenecks. Meso focuses on different routes and different service periods, time-dependent accessibility at different periods, involving a different number of transfers. Different route stops, like consider separate lanes on a link in the highway network. Microscopic is schedule-based and has detailed timetables. Its modeling elements are individual agents.
6. In the stage of **model calibration**, the workflow should calibrate **demand-side models**, individually or simultaneously, where the former includes O-D flow and path flow estimation. The column generation algorithms will be adapted as a post-processing process to enumerate candidate paths and add the toll for different paths using different service segments. The feedback loops between congested highway travel time and driving time of buses using the highway networks will be considered.
7. Data from multiple sources across different resolutions will be merged for model calibration. The uniform node and link structure make it possible to apply multiple source data with different dimensions like mobile phone data, link count data, and boarding ridership data.

1.4 Research Approaches

Analysts should convert the GTFS data files into the standard node and link network files using the GMNS as the building block. Then they can integrate the base network generated by GTFS data with other networks, such as highway networks. GMNS-oriented open modeling standard data covers a range of public transit modes, such as bus, metro rail, or railway, which facilitates a multi-modal modeling integration effort. Then the integrated network can generate OD demand considering mode choice. In addition, analysts can connect external points of interest (POI) and land use data to the node, link, and zone layers in a GMNS-oriented transportation model structure. Finally, we can utilize this open-source network structure to complete various complicated tasks, including the standard traffic assignment, the multi-modal and mobility service optimization, operations and control models, microscopic transit simulation, and model calibration, as shown in **Figure 1.1**.

Figure 1.1 Flowchart. Model development enhancements using open data standards

2 DATA STRUCTURE

In this section, we try to help users learn the input and output data structure, the GTFS format, and the GMNS format, respectively. The GTFS defines a common format for public transportation schedules and associated geographic information. It is used by thousands of public transport providers. GMNS, proposed by the Zephyr Foundation, defines a common human and machine-readable format for sharing routable road network files. It is designed to be used in multi-modal static and dynamic transportation planning and operations models. It will facilitate the sharing of tools and data sources by modelers. For additional information on GMNS goals, history, and requirements, please see the wiki.

GMNS (version 0.92) includes the following features for use in static models:

1. Configuration information and use definitions.
2. Node and link files, to establish a routable network.

For dynamic models, this version includes the following optional additional features:
1. A segment file, with information that overrides the characteristics of a portion of a link.
2. A lane file that allocates portions of the right-of-way. Lanes include travel lanes used by motor vehicles. They may also optionally include bike lanes, parking lanes, and shoulders.

The data flow structure of converting GTFS to GMNS is shown in **Figure 2.1**.

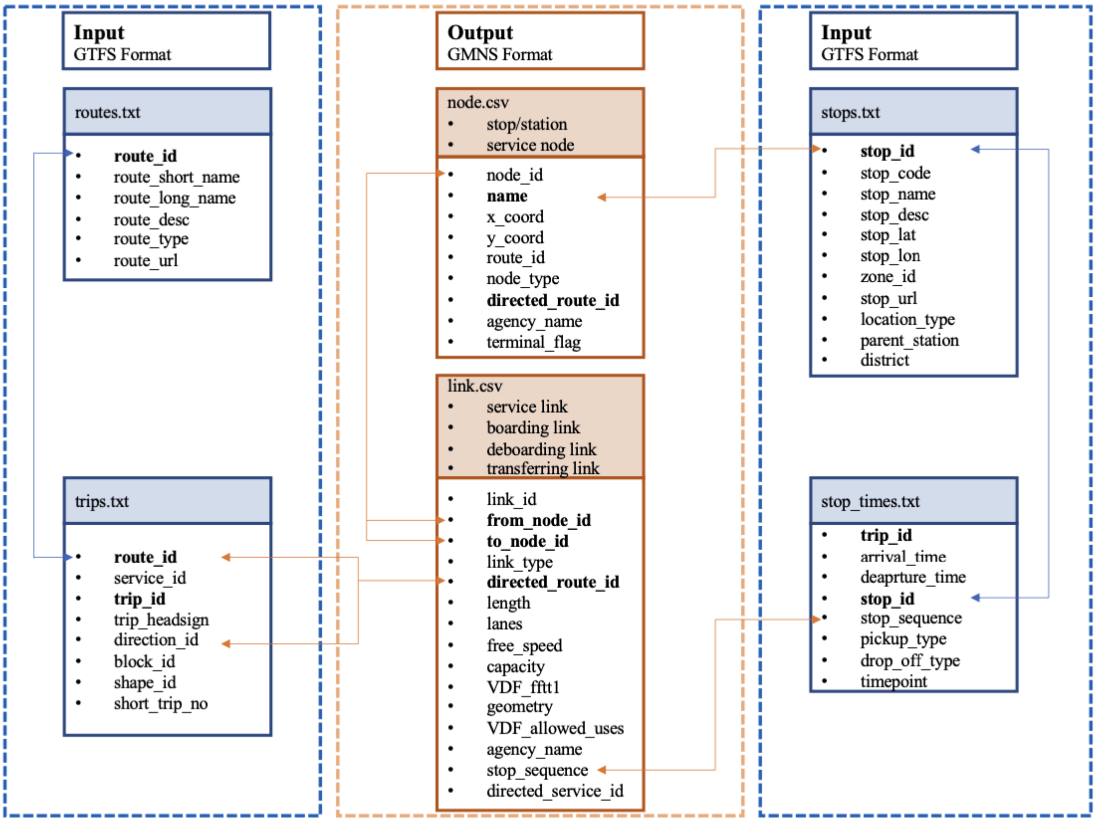


Figure 2.1 Data flow structure

2.1 Input
The input data can be downloaded from the GTFS download website. The original data includes the information on basic routes, trips, stops, and stop times.
The input files of GTFS include
- **routes.txt**
- **trips.txt**
- **stops.txt**
- **stop_times.txt**

The definition of the above files can be referenced here.

2.2 Output
gtfs2gmns uses GMNS as the standard when processing and manipulating networks, thus any network in GMNS format is fully compatible with gtfs2gmns.

As **Figure 2.2** shows, the underlying network is a physical transit network, and the upper network is a service network.

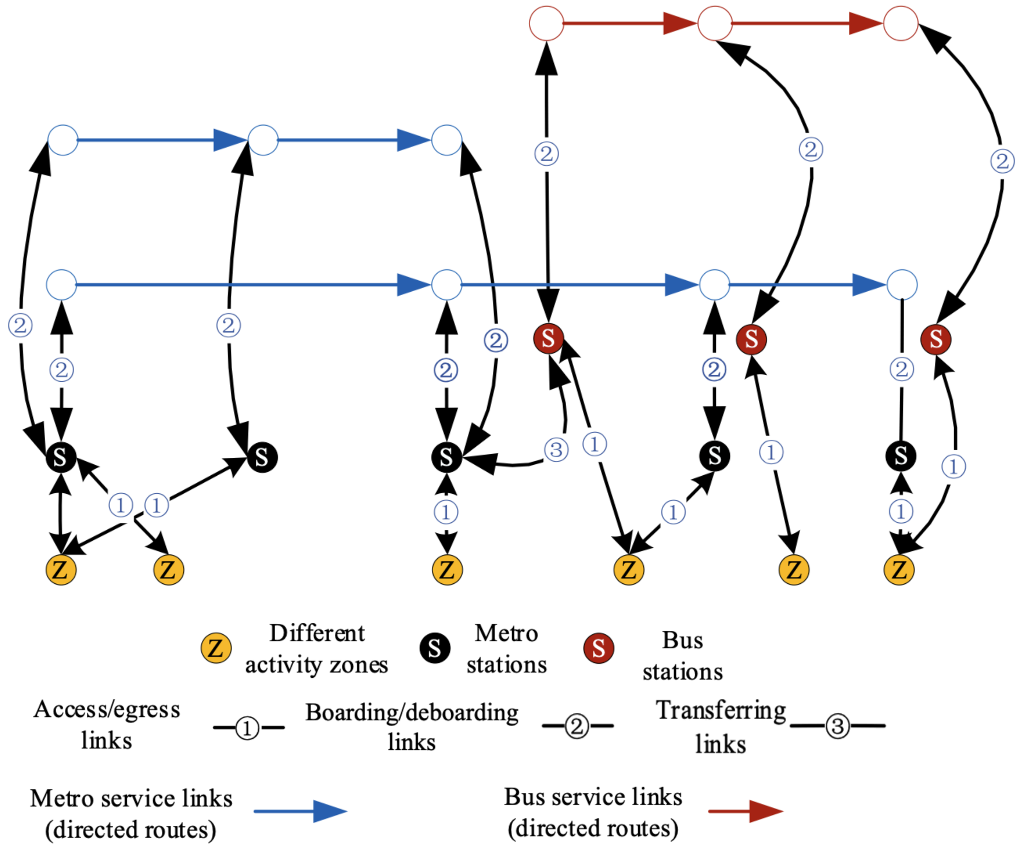


Figure 2.2 Physical layer and service layer of multi-modal transit network

This section will use a bus route and a rail route in Phoenix to explain the structure of the services network.

2.2.1 node.csv

Convert the GTFS stops into GMNS format. As **Figure 2.2** shows, node.csv includes three types of nodes, stop, station and service node (named as 'bus_service_node', 'tram_service_node' or 'metro_service_node'). In the node structure, different node type names, such as stop and station, are convenient for us to identify different modes of transit.

Table 2.1 shows the representation of different types of nodes. The stop and station essentially are physical nodes. For the representation of service nodes, the route's information is added. Thus, every route has a specific service node representation at a fixed physical node.

Physical nodes

The bus/tram stops and metro stations in the real world are called physical stops/stations. A route means a line that goes through exactly the same physical stop/station, e.g. EARTH.1 and EARTH.2. There are multiple routes on the same line because of different directions of the same line, and stopping at different physical nodes of the same direction due to one-way restrictions or construction. For example, the black dots in **Figure 2.3** and **Figure 2.4** show physical nodes of route RAIL.1 and EARTH.1, respectively.

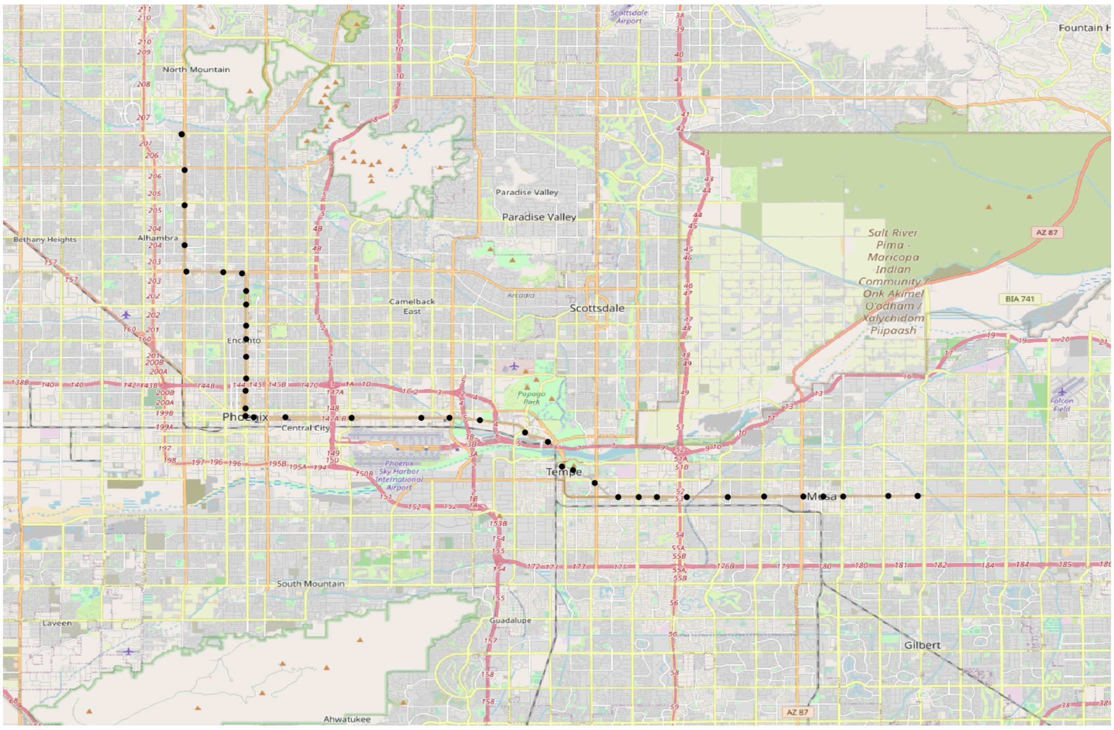


Figure 2.3 The physical nodes of Rail.1 in Phoenix

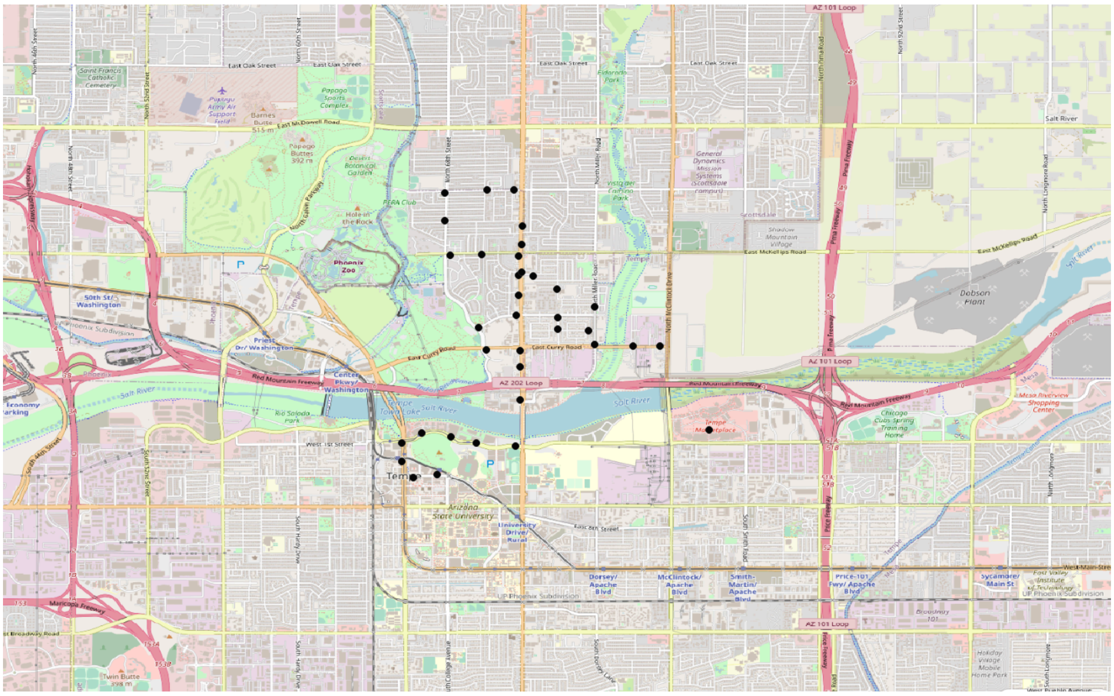


Figure 2.4 The physical nodes of Earth.1 in Phoenix

Table 2.1 the transit network includes three types of nodes:

A42 RAIL.1.6655:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36;37;38

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	name	node_id	physical_nod	x_coord	y_coord	route_type	route_id	node_type	directed_rou	directed_service_id	zone_id	agency_name	geometry	terminal_flag
2	6655	1000001	1000001	-112.09966	33.537448	0	RAIL	stop				Valley Metro	POINT (-112.09976400000001 33.537347999999994)	0
42	RAIL.1.6655:	1500001	1000001	-112.09976	33.537348	0	RAIL	tram_service_node	RAIL.1	RAIL.1:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36;37;38		Valley Metro	POINT (-112.099664 33.537448)	0
82	11	2000001	2000001	-111.93882	33.42537	3	EART	stop				Valley Metro	POINT (-111.938917 33.42527)	0
118	EART.1.11:1;	2500001	2000001	-111.93892	33.42527	3	EART	bus_service_node	EART.1	EART.1:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36		Valley Metro	POINT (-111.938817 33.42537)	0
123	EART.1.3037	2500006	2000009	-111.93616	33.42561	3	EART	bus_service_node	EART.1	EART.1:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36		Valley Metro	POINT (-111.936063 33.42571)	1

Node type code	Node type name	Definition	node_id	name	directed_route_id	directed_service_id	terminal_flag
1	stop	Bus/tram stops	f rom 1000001 (physical_node_id)	name in GTFS (e.g. 6655)	/	/	mark terminal flag for each stop
2	station	Metro stations	from 3000001 (physical_node_id)	name in GTFS (e.g. 11)	/	/	mark terminal flag for each stop
3	service node	Nodes of service network	from 1500001 (service_node_id)	directed_route_id.name:stop_sequence (e.g. RAIL.1.6655: 1;2;3;4;5;6)	route_name.direction, the direction is represented by 1 or 2 (e.g. RAIL.1)	directed_route_id:stop_sequence (e.g. RAIL.1:1;2;3;4;5;6)	mark terminal flag for each stop

Service nodes

To build service networks, we construct different virtual nodes nearby each physical node for different routes. That means each physical node could correspond to multiple service nodes, as **Figure 2.5** and **Figure 2.6** shows.

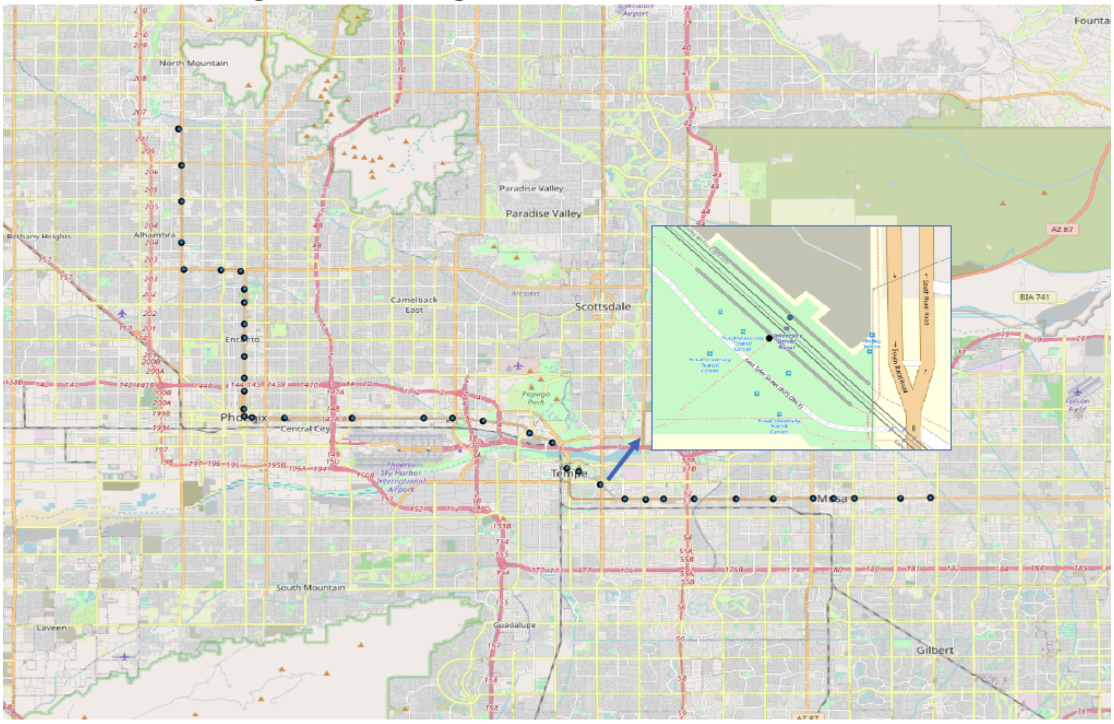


Figure 2.5 The service nodes of Rail.1 in Phoenix

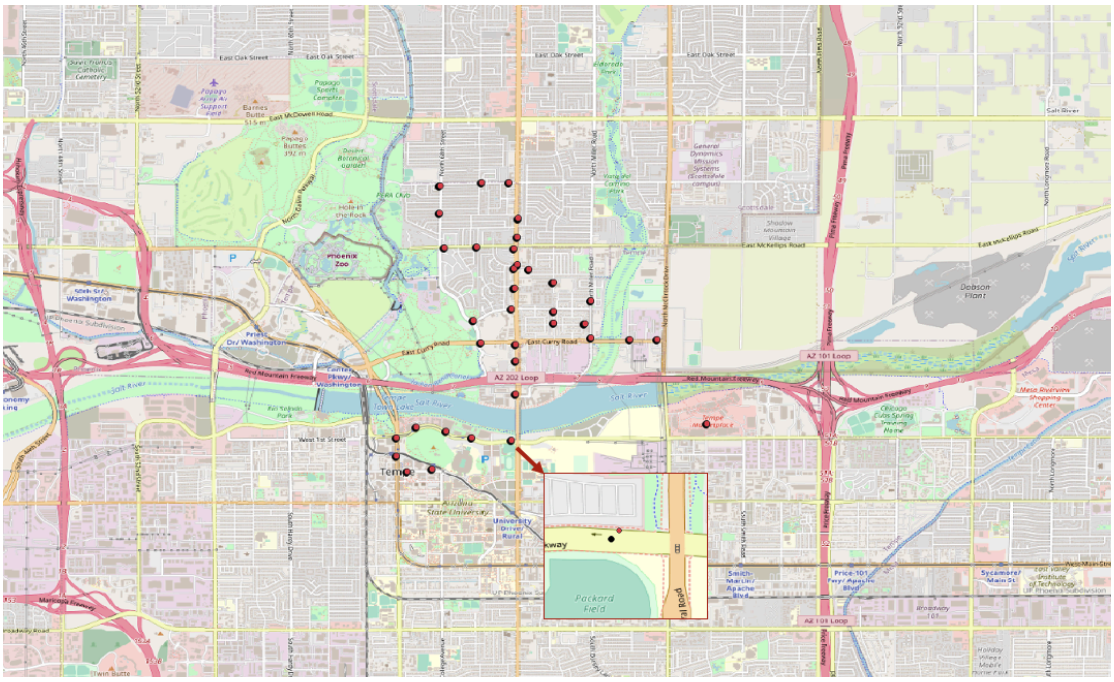


Figure 2.6 The service nodes of Earth.1 in Phoenix

2.2.2 link.csv

Convert the GTFS routes/trips into GMNS format. Link.csv includes three types of links. Service links are defined by route link and the starting/end node are both service nodes. Boarding/debearding links are the connections between service nodes and physical nodes. Transferring links represent the walking links between different physical nodes and we assume the walking speed is 1 km/h.

As **Figure 2.2** shows, the transit network includes three types of links:

Link type code	Link type name	Definition
0	Access/Egress link	Links from activity zones to bus/metro or commuter rail stations
1	Boarding/Deboarding links	Links connecting stops/stations to corresponding directed route stops
2	Service links	The links representing the planning of each route
3	Transferring links	Transferring links between different stations/stops

Note:
The access/egress links cannot be created in our gtfs2gmns tool for now. This function can be conducted in DTALite.exe and will be added to our gtfs2gmns tool soon.

Services links
In the service network, the links between two neighboring service nodes are called service links, as the blue links in **Figure 2.7** and red links in **Figure 2.8** shows.

Boarding and deboarding links
As the black links of **Figure 2.7** and **Figure 2.8** shows, the links connecting physical nodes and service nodes are called boarding/deboarding links.

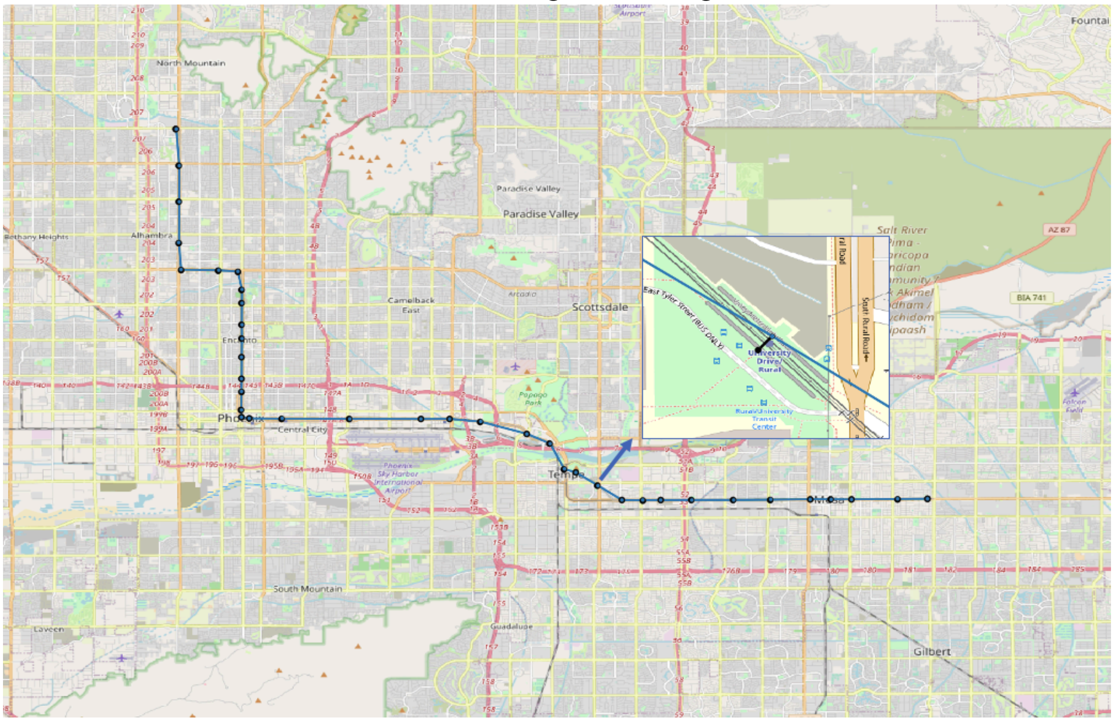


Figure 2.7 The service links and boarding/deboarding links Rail.1 in Phoenix

Table 2.2 the transit network includes three types of links

1	link_id	from_node_i	to_node_id	facility_type	dir_flag	directed_rou	link_type	link_type_name	length	lanes	capacity	free_speed	cost	VDF_fftt1	VDF_cap1	VDF_alpha1	VDF_beta1	VDF_penalty	geometry	VDF_allowed_uses1	agency_name	stop_sequence	directed_service_id
2	1000001	1500038	1500005	tram	1	RAIL.1	1	service_links	1147.06242	20	999999	34.3946752	0	2	19999980	0.15	4	0	LINESTRING	w_bus_only;w_bus_metro;d_bus_only;d_bus_metro	Valley Metro	1	RAIL.1:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36;37;38
39	1000074	1000001	1500001	tram	1	RAIL.1	2	boarding_links	14.4757029	1	999999	2	0	1.5	999999	0.15	4	0	LINESTRING	w_bus_only;w_bus_metro;d_bus_only;d_bus_metro	Valley Metro	-1	RAIL.1.6655:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36;37;38
119	2000001	2500032	2500007	bus	1	EART.1	1	service_links	1194.18935	20	999999	17.9083631	0	4	19999980	0.15	4	0	LINESTRING	w_bus_only;w_bus_metro;d_bus_only;d_bus_metro	Valley Metro	1	EART.1:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36
154	2000070	2000001	2500001	bus	1	EART.1	2	boarding_links	14.4833887	1	999999	2	0	1.5	999999	0.15	4	0	LINESTRING	w_bus_only;w_bus_metro;d_bus_only;d_bus_metro	Valley Metro	-1	EART.1.11:1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36

link_type_name	link_type	facility_type	directed_route_id	VDF_fftt1	VDF_allowed_uses1	stop_sequence
service_links	1	bus or tram	definition same as node.csv (e.g. ZOOM.1)	headway for each route	whether agent type is allowed in this link (e.g. w_bus_only;w_bus_metro; d_bus_only;d_bus_metro)	stop sequence for each route
boarding_link/ deboarding_link (from stations to their passing routes)	2	bus or tram	definition same as node.csv (e.g. ZOOM.1)	entrance: $\frac{1}{2} \bullet \frac{period\ time}{number\ of\ trips}$ exit: 0	whether agent type is allowed in this link (e.g. w_bus_only;d_bus_only)	-1
transferring_links (from stations to stations)	3	sta2sta	-1	$\frac{length}{1000}$	whether agent type is allowed in this link (e.g. w_bus_only;d_bus_only)	/
* length unit: meter. * VDF_penalty1: for service links and boarding links, penalty=0; for transferring links, if the link is from 'stop' to 'stop', the penalty=99. * VDF_cap1: = lanes*default_capacity (999999) * VDF_alpha1: = 0.15 * VDF_beta1: = 4						

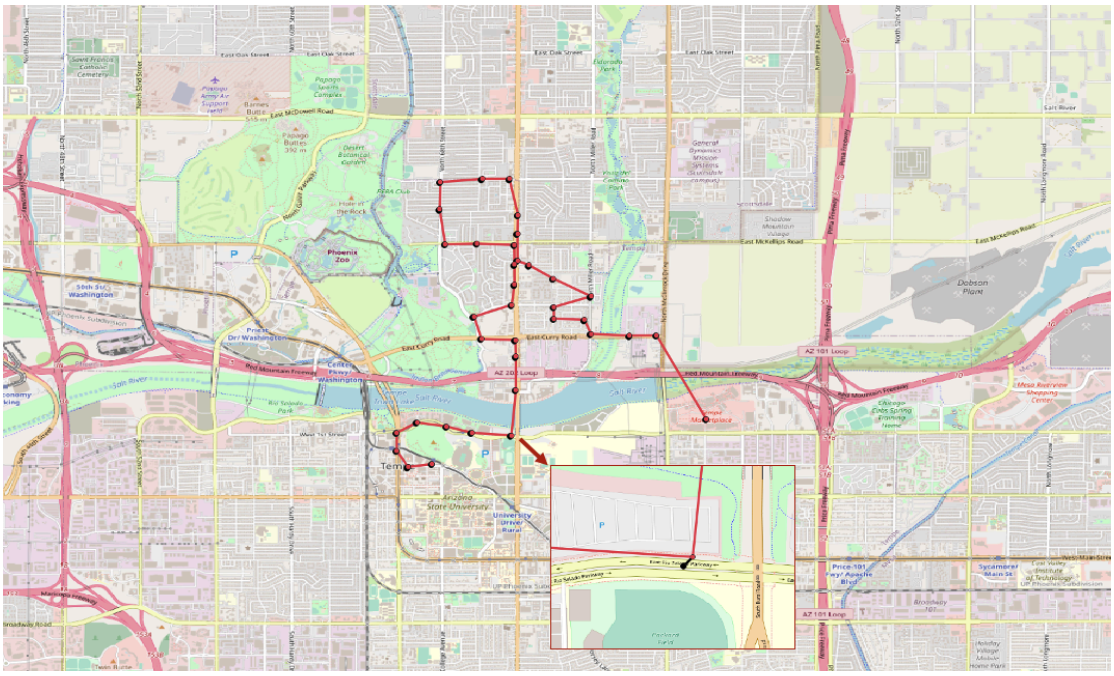


Figure 2.8 The service links and boarding/deboarding links of Earth.1in Phoenix

Transferring links

The transferring links only connect physical nodes. Specifically, as **Figure 2.2** shows, when a passenger wants to transfer from metro to bus, he can first deboard from the metro station to the physical stop through a deboarding link, then transfer from metro station to bus stop, finally, get on the bus stop through a boarding link. That's why we set the cost of the boarding link as the passenger's waiting time and the deboarding link as 0. The transferring links between RAIL.1 and EARTH.1 are shown as the green links in **Figure 2.9**.

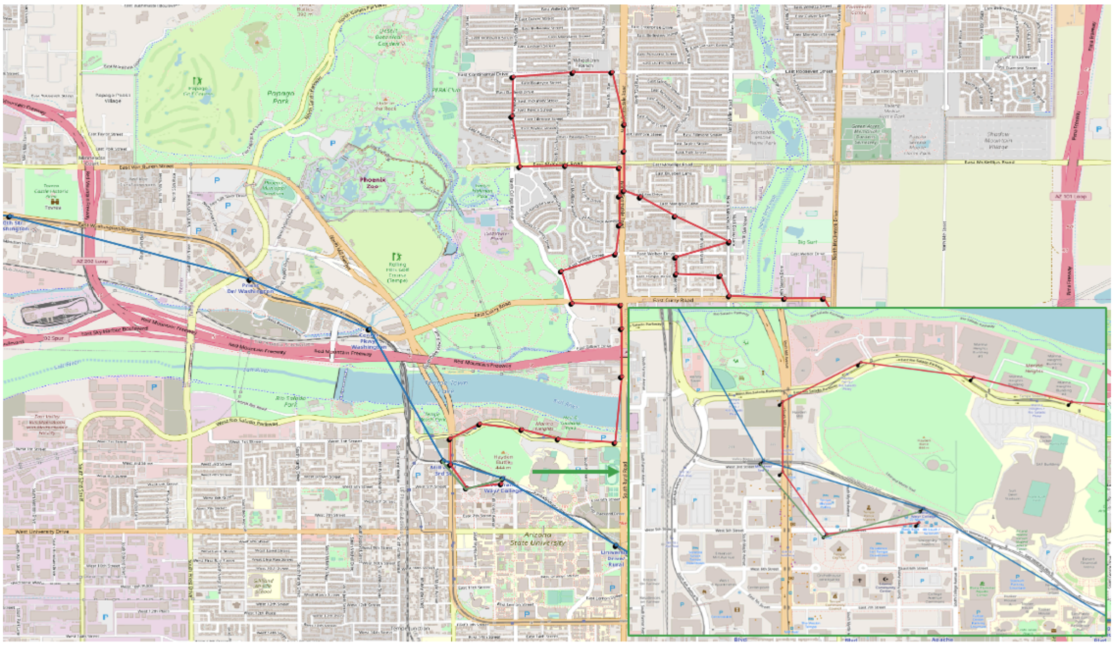


Figure 2.9 The transferring links between RAIL.1 and EARTH.1 in Phoenix

Figure 2.10 shows the different types of links in the Washington DC transit network.

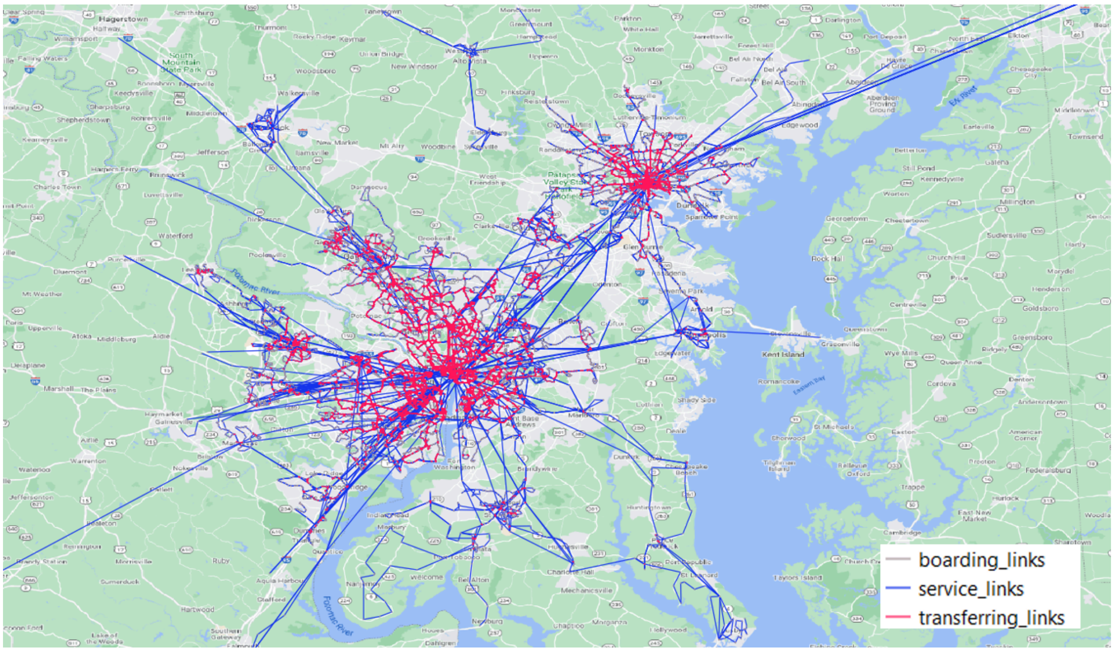


Figure 2.10 Different types of links of Washington DC transit network

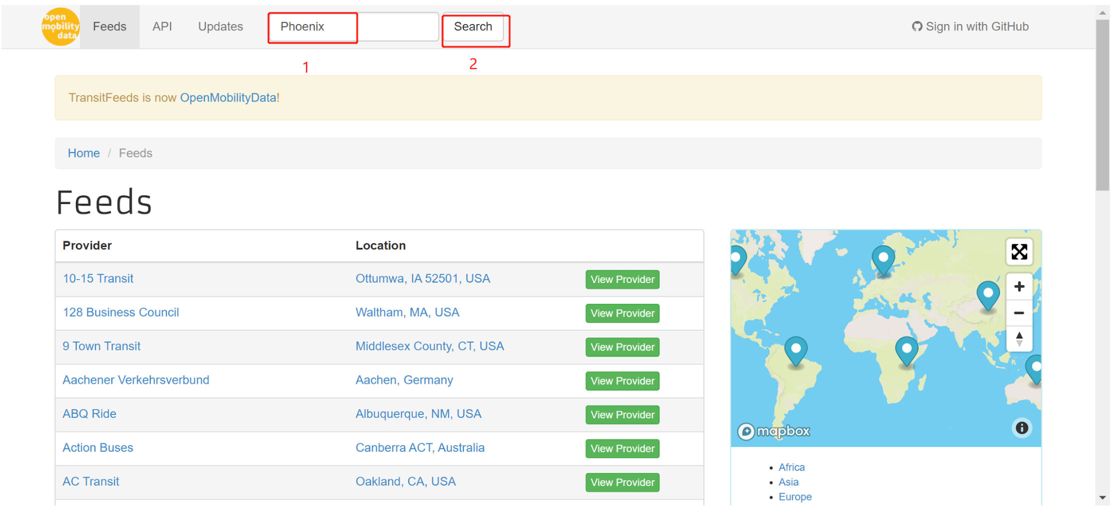
3 QUICK START

In this section, some examples are provided to quickly show how to use gtfs2gmns to generate multimodal networks.

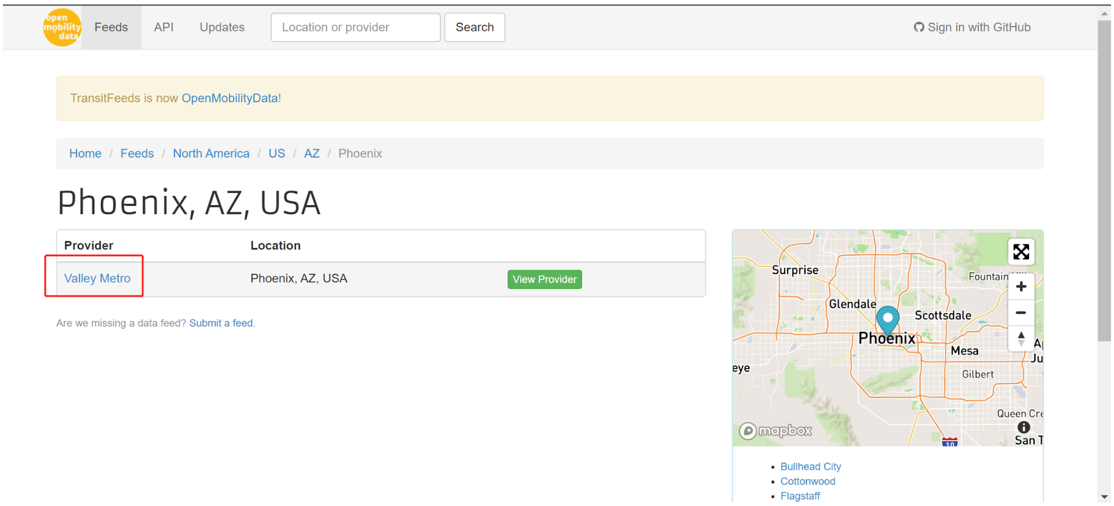
3.1 Download GTFS Data

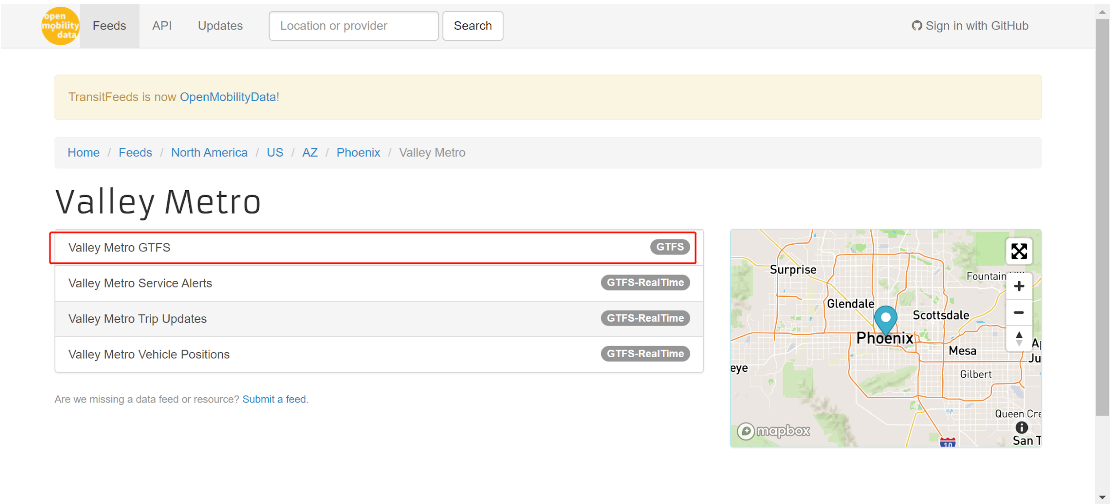
To reduce uncertainties while directly parsing network data from the GTFS, gtfs2gmns uses downloaded .txt files to extract useful network information. As a result, the first step is preparing .txt files of GTFS data.

Step 1: On the [GTFS download website](), search the **location or provider**, such as Phoenix.

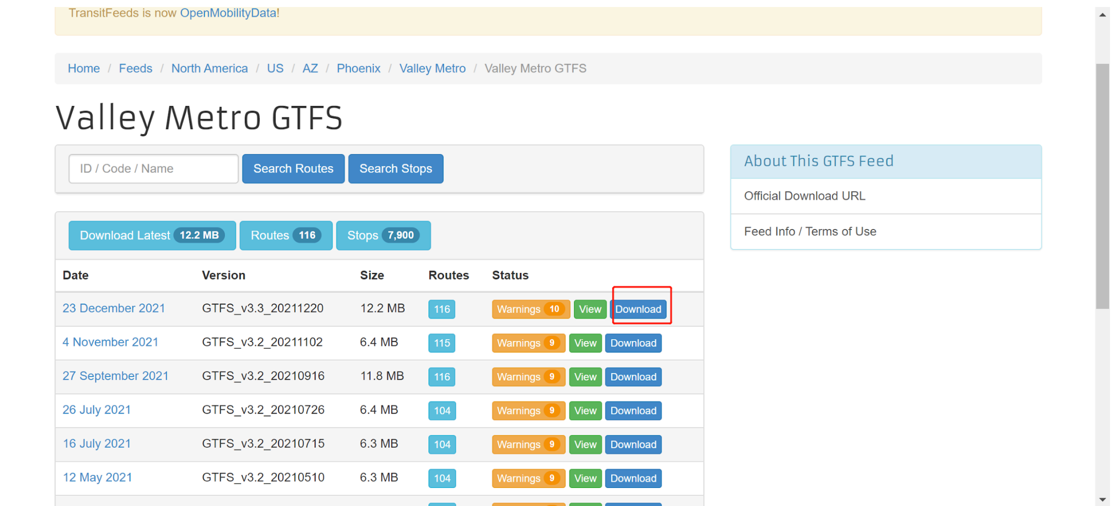


Step 2: Click **provider** name to select the datasets.

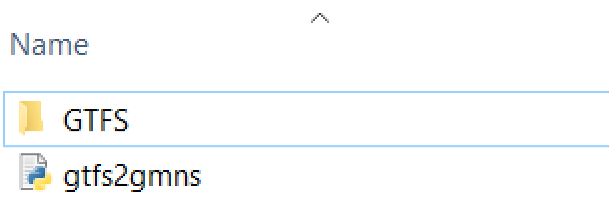




Step 3: Download GTFS datasets.



Step 4: Put the downloaded gtfs.zip into the GTFS data folder, and unzip the gtfs folder. Users also can customize the gtfs folder name, such as Phoenix.



The GTFS datasets are ready now.

3.2 Parse GTFS Data

Step 1: Ensure to read the right data folder in the main function.

```
if __name__ == '__main__':
    global period_start_time
    global period_end_time
    input_gtfs_path = 'GTFS'
    output_gmns_path = '.'
    time_period_id = 1
    time_period = '1200_1300'
    period_start_time, period_end_time = _hhmm_to_minutes(time_period)

    gtfs2gmns(input_gtfs_path, output_gmns_path)

```

Step 2: Users can customize the time period, such as 12:00 to 13:00.

3.3 Output Networks to CSV

To obtain the output service network, users only need to run the Python code.

```
convert  6950 transferring links successfully... using time 30.145007848739624 s
convert  7000 transferring links successfully... using time 30.34590172767639 s
convert  7050 transferring links successfully... using time 30.54547691345215 s
convert  7100 transferring links successfully... using time 30.67658805847168 s
convert  7150 transferring links successfully... using time 30.92137384414673 s
convert  7200 transferring links successfully... using time 31.06439185142517 s
run time --> 109.36868786811829

Process finished with exit code 0
```

3.4 Use QGIS or Nexta to visualize the network

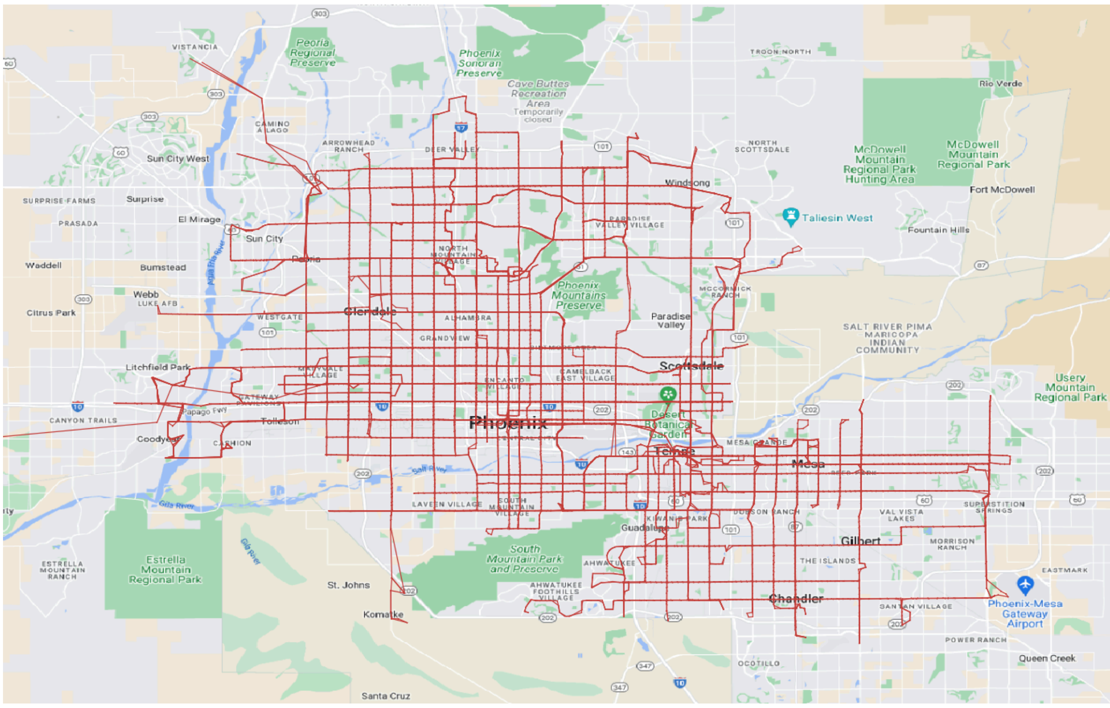


Figure 3.1 QGIS visualization for Transit network of Phoenix

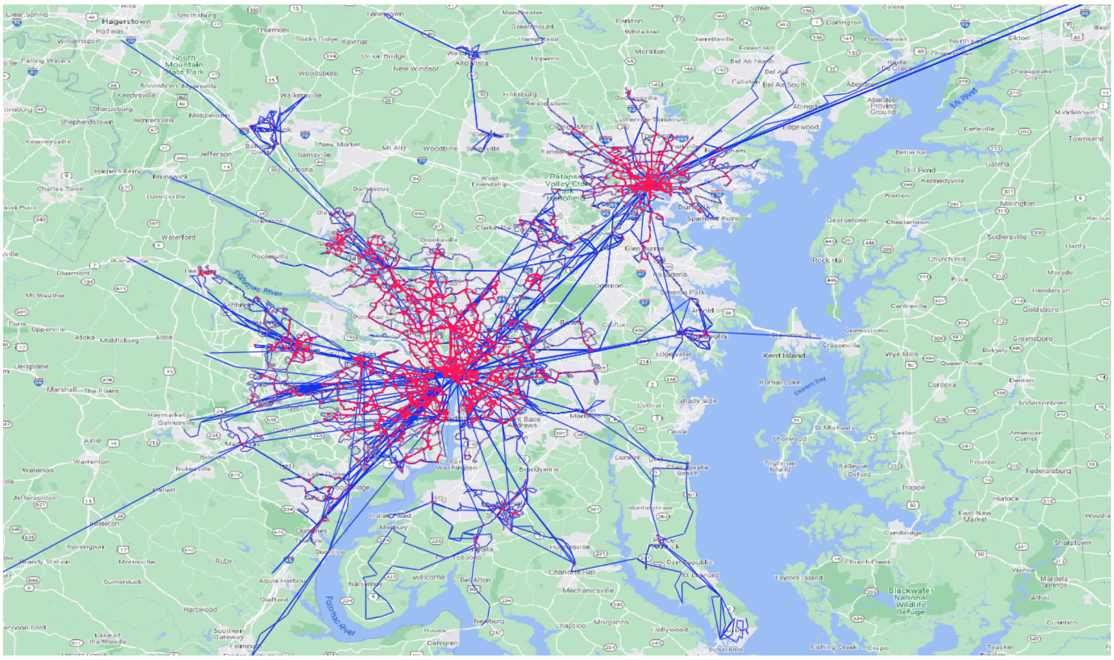


Figure 3.2 QGIS visualization for Transit network of Washington DC

4 TRANSIT ASSIGNMENT APPLICATION

Multi-modal transit traffic assignment is a significant topic. In this section, we will use Phoenix as an example to show how to handle the preparation work and results analysis before and after transit assignment, respectively.

4.1 Procedure

Step 1: Build transit service network

As Chapter 3 stated, we already know how to use the gtfs2gmns tool to build a transit service network.

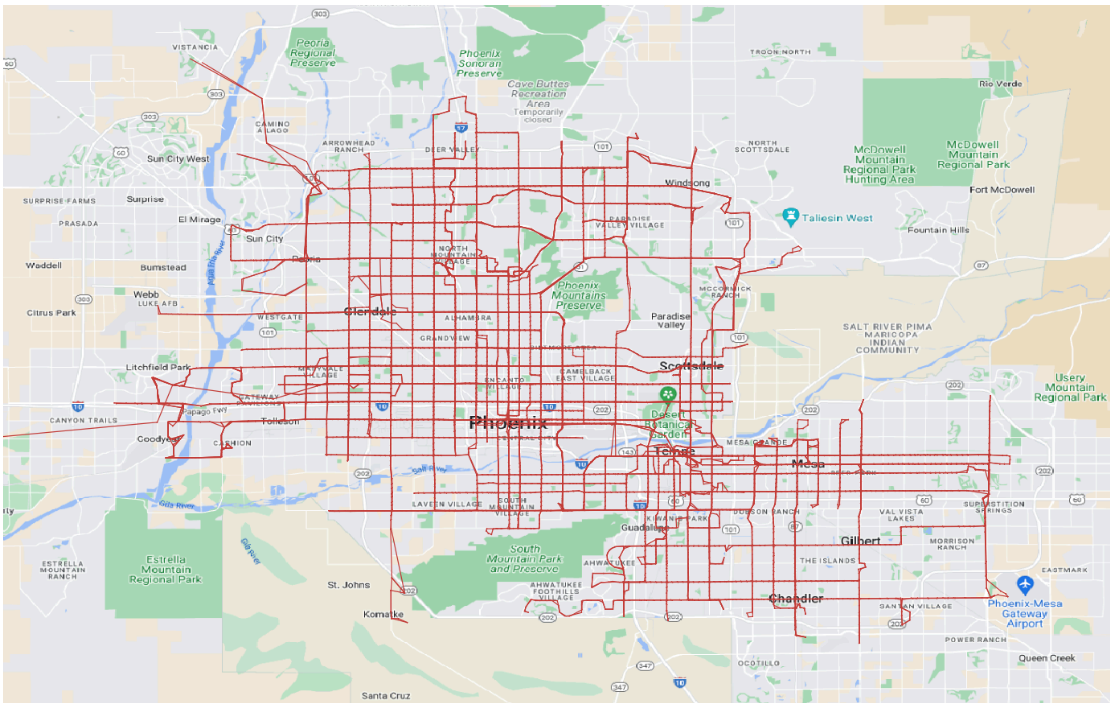


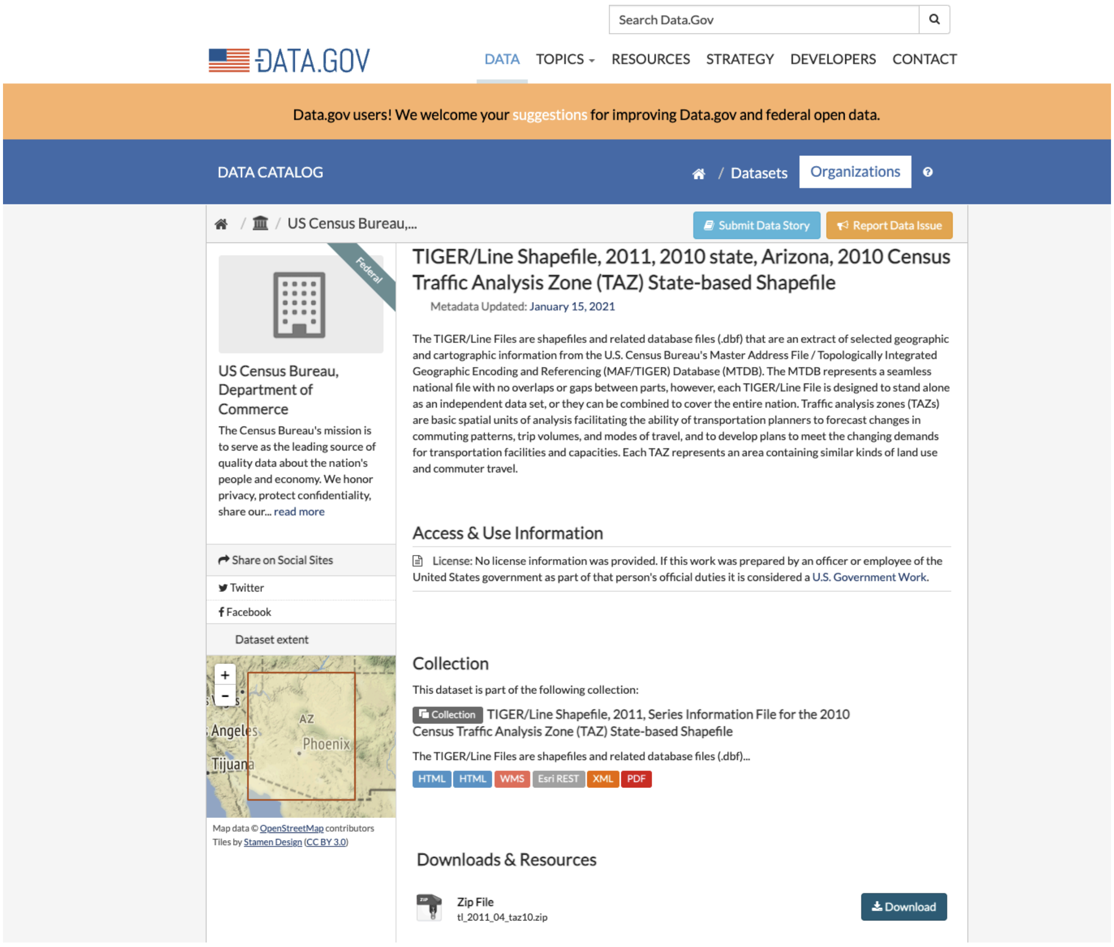
Figure 3.1 QGIS visualization for Transit network of Phoenix

Step 2: Prepare zone.csv

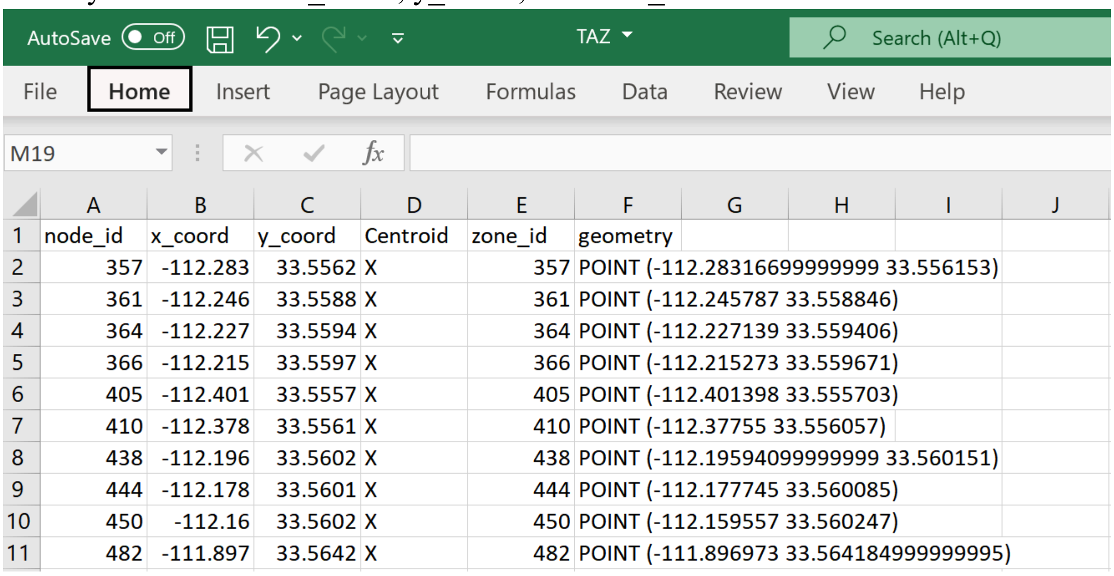
Please feel free to use the zone.csv of your project. Note that you should keep the same format and the key fields as Step 2.3.

If you do not have the zone information, we provide a method stated in the following steps to prepare zone.csv.

Step 2.1 Download census.shp file of Phoenix on the DATA.GOV

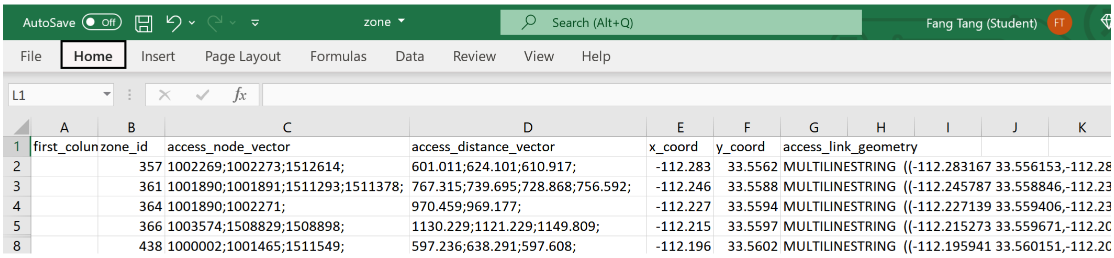


Step 2.2 Use QGIS to export TAZ as TAZ.csv
The key fields include x_coord, y_coord, and zone_id



	A	B	C	D	E	F
1	node_id	x_coord	y_coord	Centroid	zone_id	geometry
2	357	-112.283	33.5562	X	357	POINT (-112.28316699999999 33.556153)
3	361	-112.246	33.5588	X	361	POINT (-112.245787 33.558846)
4	364	-112.227	33.5594	X	364	POINT (-112.227139 33.559406)
5	366	-112.215	33.5597	X	366	POINT (-112.215273 33.559671)
6	405	-112.401	33.5557	X	405	POINT (-112.401398 33.555703)
7	410	-112.378	33.5561	X	410	POINT (-112.37755 33.556057)
8	438	-112.196	33.5602	X	438	POINT (-112.19594099999999 33.560151)
9	444	-112.178	33.5601	X	444	POINT (-112.177745 33.560085)
10	450	-112.16	33.5602	X	450	POINT (-112.159557 33.560247)
11	482	-111.897	33.5642	X	482	POINT (-111.896973 33.564184999999995)

Step 2.3 Run DATLite to generate zone.csv
Users can customize the access node vector to determine



first_colun	zone_id	access_node_vector	access_distance_vector	x_coord	y_coord	access_link_geometry
	357	1002269;1002273;1512614;	601.011;624.101;610.917;	-112.283	33.5562	MULTILINESTRING ((-112.283167 33.556153,-112.28
	361	1001890;1001891;1511293;1511378;	767.315;739.695;728.868;756.592;	-112.246	33.5588	MULTILINESTRING ((-112.245787 33.558846,-112.23
	364	1001890;1002271;	970.459;969.177;	-112.227	33.5594	MULTILINESTRING ((-112.227139 33.559406,-112.23
	366	1003574;1508829;1508898;	1130.229;1121.229;1149.809;	-112.215	33.5597	MULTILINESTRING ((-112.215273 33.559671,-112.2C
	438	1000002;1001465;1511549;	597.236;638.291;597.608;	-112.196	33.5602	MULTILINESTRING ((-112.195941 33.560151,-112.2C

Step 3: Prepare OD demand.csv

o_zone_id	d_zone_id	volume
101	101	157.33334
101	132	1
101	133	0.666666
101	135	0.333333
101	199	1.333334
101	206	0.333333
101	1011	2

Step 4: Transit assignment

Run DTALite executable to obtain assignment results, link_performance.csv especially.

Step 5: Results analysis

Step 5.1 Check OD accessibility

1. Check access node vector from each TAZ

Step 5.2 Check link_performance.csv

1. Check assignment volume.
2. Check free speed within a bound of driving and walking speed.
3. Sort the volume field from largest to smallest, to check the highest loading volume link
4. Use a pivot table to check the maximum count for each transferring link, to check highly used transferring links
5. Use a pivot table to check the average length, # of lanes, and capacity for different types of links.

Step 5.3 Check how changed zone-to-node accessibility and service network design can change the system performance.

Table 4.1 Procedure of transit assignment

Step	Function	Input	Output
1 Build transit service network	<gtfs2gmns.py> Read GTFS data and convert it into GTFS format.	• GTFS files: agency.txt, stops.txt, routes.txt, trips.txt, stop_times.txt • zone.csv (optional)	• node.csv • link.csv • source_node.csv (same as node.csv if there is no zone.csv as input)
2 Zone creation	<DTALite.exe> Create zone.csv (building the mapping from zone to activity nodes) and corresponding distance file.	• node.csv • link.csv	• zone.csv (access_node_vector) • gc_distance.csv (unit: mile)
3 Transit OD accessibility	<DTALite.exe> Create OD accessibility file with different agent types.	• node.csv • link.csv • zone.csv • gc_distance.csv	• od_accessibility.csv with OD zone id, travel time and distance for connected agent type
4 Simulation	<DTALite.exe> Create route assignment.	• all above	• route_assignment.csv

4.2 Results Visualization

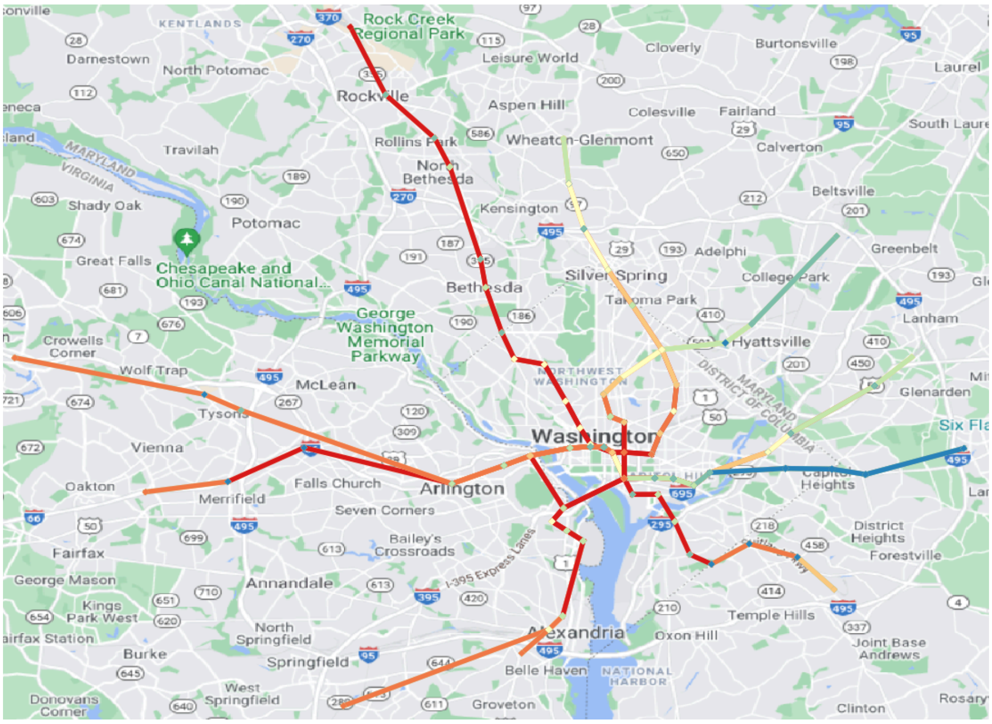


Figure 6 Transit assignment results of metro in Washington DC at AM period

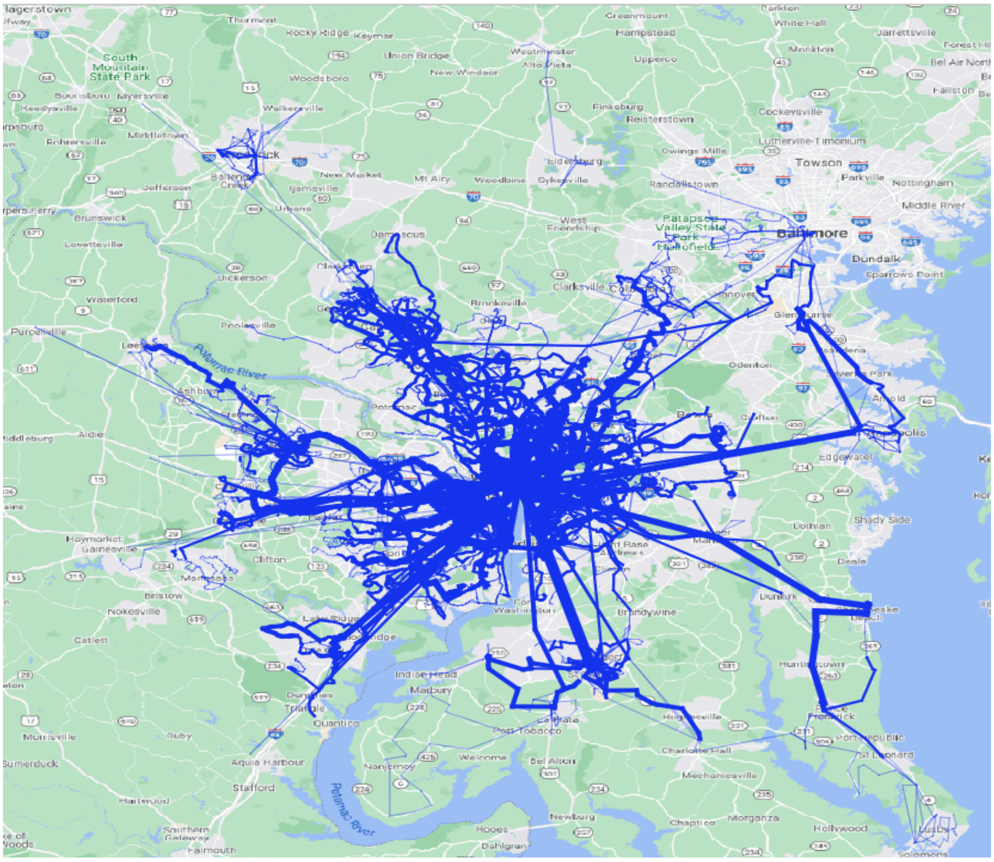


Figure 7 Transit assignment results of bus in Washington DC at AM period