2022

GTFS2GMNS Users' Guide

Contributors
Xuesong (Simon) Zhou
xzhou74@asu.edu

Xin (Bruce) Wu
xinwu3@asu.edu

Fang (Alicia) Tang
fangt@asu.edu

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

Cafer Avci
mailto:cafer.avci@aalto.fi

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, and rail etc. networks for any agency and multi-agencies in the world, and output networks to csv files in General Modeling Network Specification (GMNS) format for seamless data sharing and research collaboration. gtfs2gmns mainly focus on providing researchers and practitioners with flexible, standard and ready-to-use multi-modal transit networks to facilitate various research and applications on traffic modeling.

Contents

Glossary

Term	Description
Analysis time period	A planning time interval determined by users. It can be 6AM-9AM, also it can be 6AM-12PM.
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 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 that significantly improves traffic flow without requiring additional lanes. Different travel modes will affect each other. For example, there are feedback loops between congested highway travel time and driving time of buses using the highway networks. To improve the quality of the public transit system analysis and planning program, including mobility optimization and traffic assignment, the research of multi-modal network modeling is increasingly vital.

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 utilize the existing static traffic assignment package fully, helps decision-makers explore how different modes will play out in different futures, and helps make 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 often reduces another. For example, cycling and walking were sidelined by the spread of private cars and the road infrastructure built to cater for 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 options become embedded. Users want seamless end-to-end journeys, and an integrated approach helps to plan for this.

Obviously, it is critical to define a routable multi-modal network structure. However, since the routable structures of some nodes and links are also not explicitly defined by the currently available integrated tools provided by traffic technology companies like PTV, the integrated analysis and optimization of the multi-modal l network is still a challenge. The generalized highway network is often represented by the GMNS, a great general data format. However, the GMNS data format cannot describe the basic data sharing structure for the public agencies to share the data. 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 is complicated and dynamic. The complexity has different versions online, offline. 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. Furthermore, the problems also need to consider the passengers' preferences and constraints.

In this context, a unified network structure will be the next major milestone in the evolution of mobility optimization. The unified data structure will allow us 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. Also, the open unified data hub can connect to the multi-modal and mobility service optimization, operations and control models, multiresolution modeling structure mainly for further microscopic transit simulation, model calibration using multiple data sources across different resolutions, and better calibrated and integrated models.

Transportation network modeling can offer analysis techniques that help society understand and optimize the movement of goods and people. So we try to integrate various elements into the analysis, including the movement of goods, people, and the features of infrastructure. These analyses provide us with novel ways to improve transportation safety and efficiency while reducing vehicle emissions. The growing capabilities of automation and communication technologies offer new solutions to reach this goal. Still, they also dramatically increase the dynamics and complexity of underlying transportation network structures, creating challenges for engineers/planners/officials trying to manage transportation networks.

An open unified data hub can create a window of opportunity to move towards a more integrated modeling standard. 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 path2gmns, DTALite, other Commercial tools using GMNS data specifications. Analysts can perform the assignment and feed results from one model to another while maintaining consistency between the model assumptions. Then the traffic assignment results can allow analysts to look at the physical network, the headway, the delay, and the possibilities of the transfers. Moreover, a unified data hub can 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 various 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-oriented transportation model structure.
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 on-peak service hours, connections, 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 meso-structure. Macro, 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, 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**.

Figure 1. Flowchart. Model development enhancements using open data standards

2 NETWORK DESIGN

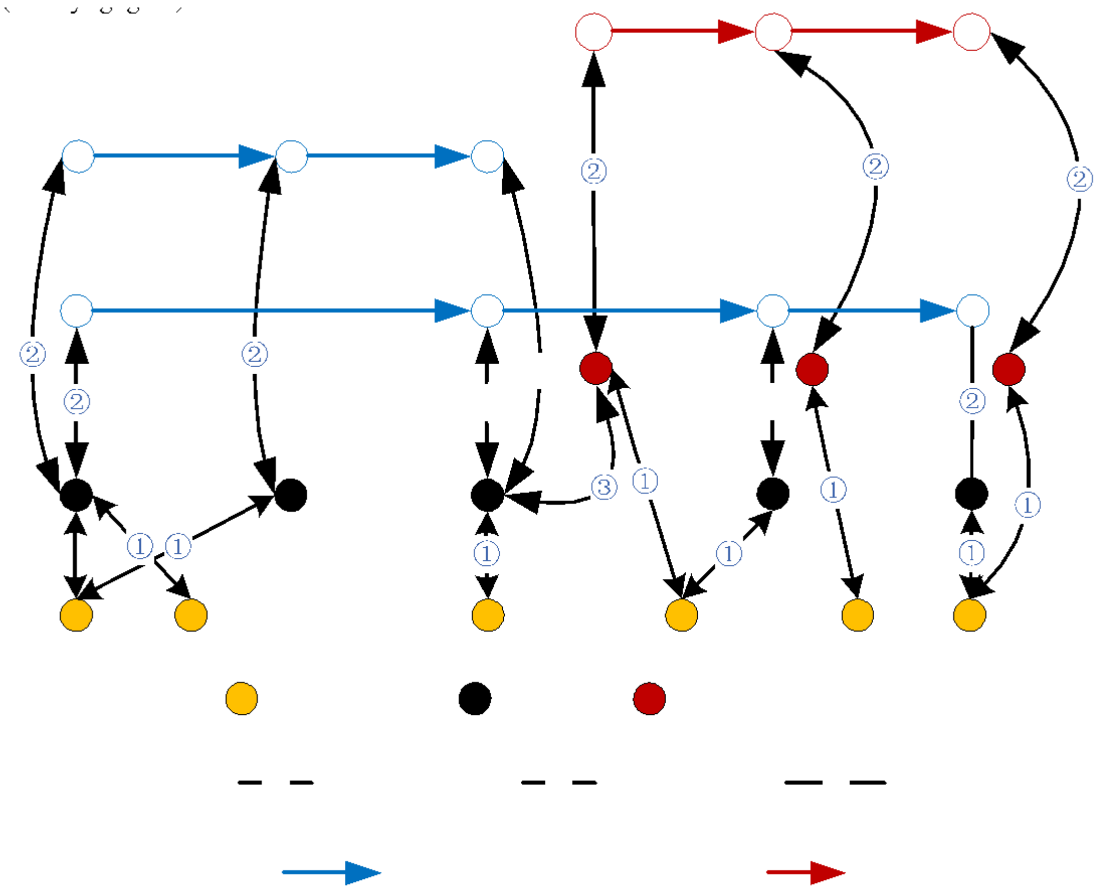


Figure 1 Physical layer and service layer of multi-modal transit network

As **Figure 1** shows, the underlying network is a physical transit network, and the upper network is a service network.

2.1 Node

As **Figure 1** shown, the transit network includes 4 types of nodes:

Node type code	Node type name	Definition
1	zone	Activity zones
2	stop	Bus stops
3	station	Metro/Rail stations
4	directed route stop	Nodes of service network

In the node structure, different node type names, such as stop and station, are convenient for us to identify different modes of transit. And to build service networks, the directed route stops are virtual stops of corresponding physical stops/stations for each route. More specifically, there could be multiple virtual stops for each physical stop/station.

2.2 Link

As **Figure 1** shown, the transit network includes 4 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	Aboard/Deboard 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 gtfs2gmms tool for now. This function can be conducted in DTALite.exe and will be added in our gtfs2gmms tool soon.

As **Figure 1** 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 deboard link, then get on the bus stop through an aboard link. That's why we set the cost of the aboard link as the passenger's waiting time and the deboard link as 0.

Figure 2 shows us the different types of links in Washington DC transit network.

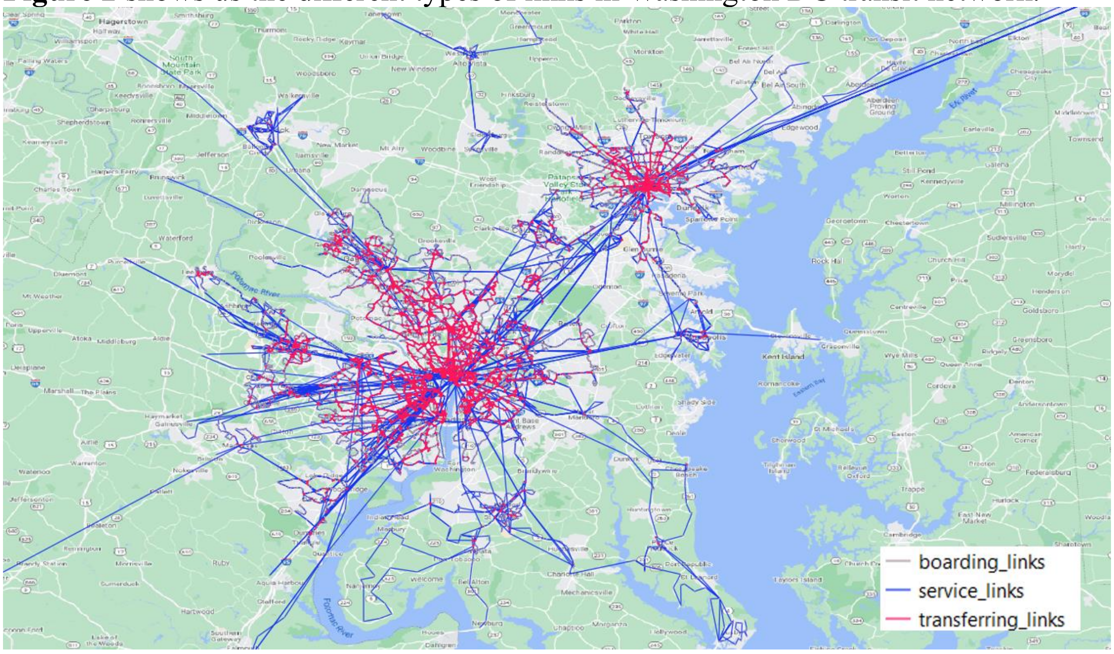


Figure 2 Different types of links of Washington DC transit network

3 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. The GMNS dataset represents the service network which is used in multi-modal static and dynamic transportation operations and planning models.

The data flow structure is shown in **Figure 3**.

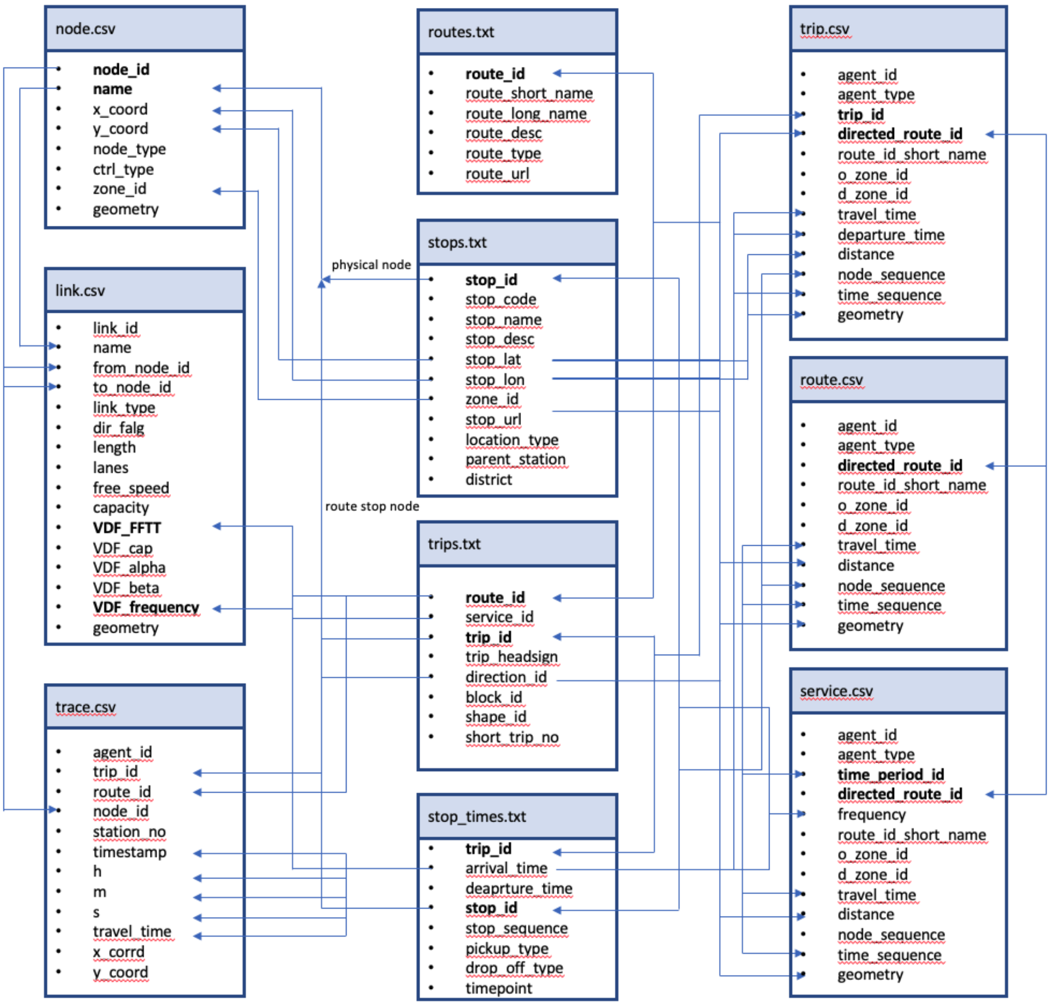


Figure 3 Data flow structure

3.1 GTFS input data

The input data can be downloaded from the GTFS download website. The original data includes the information of basic routes, trips, stops and stop times.
The input files of GTFS including:

- **trip.txt**
- **route.txt**
- **stop_times.txt**
- **stops.txt**

The definition of above files can be referenced here.

3.2 GMNS output data

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.
3. A segment_lane file that specifies additional lanes, dropped lanes, or changes to lane properties on a segment of a link.
4. A movement file that specifies how inbound and outbound lanes connect at an intersection
5. Link, segment, lane, and movement time-of-day (TOD) files, that allocates usage of network elements by time-of-day and day-of-week.
6. Signal phase and timing files, for basic implementation of traffic signals.

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

node.csv

Convert the GTFS stops into GMNS format. Node.csv includes two types of nodes, physical node (named as ‘stop’) and ‘service stop’ (names as ‘bus_service_node’ or ‘tram_service_node’). For service nodes, the specific route information is added, and the different routes (including different directions) correspond to different service nodes.

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 links are the connections between service nodes and physical nodes, classified as entrance links and exit links. Transferring links represent the walking links between different physical stops and we assume the walking speed is 1 km/h.

node_type	name	node_id	directed_route_id	directed_service_id	terminal_flag
stop	name in GTFS (e.g. 3)	from 1000001 (physical_node_id)	/	/	mark terminal flag for each stop
bus_service_node	directed_route_id.name:stop_sequence (e.g. ZOOM.1.3: 1;2;3;4;5;6;7;8;9;10)	from 1500001	route_name.direction, the direction is represented by 1 or 2 (e.g. ZOOM.1)	directed_route_id:stop_sequence (e.g. ZOOM.1:1;2;3;4;5;6;7;8;9;10)	mark terminal flag for each stop
tram_service_node	directed_route_id.name:stop_sequence (e.g. RAIL.1.5: 1;2;3;4;5;6)	from 1500001	route_name.direction, the direction is represented by 1 or 2 (e.g. RAIL1.1)	directed_route_id:stop_sequence (e.g. RAIL.1:1;2;3;4;5;6)	mark terminal flag for each stop

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_links (from stations to their passing routes)	2	bus or tram	definition same as node.csv (e.g. ZOOM.1)	entrance: $\frac{1}{2} \cdot \frac{period\ time}{number\ of\ trips}$ 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						

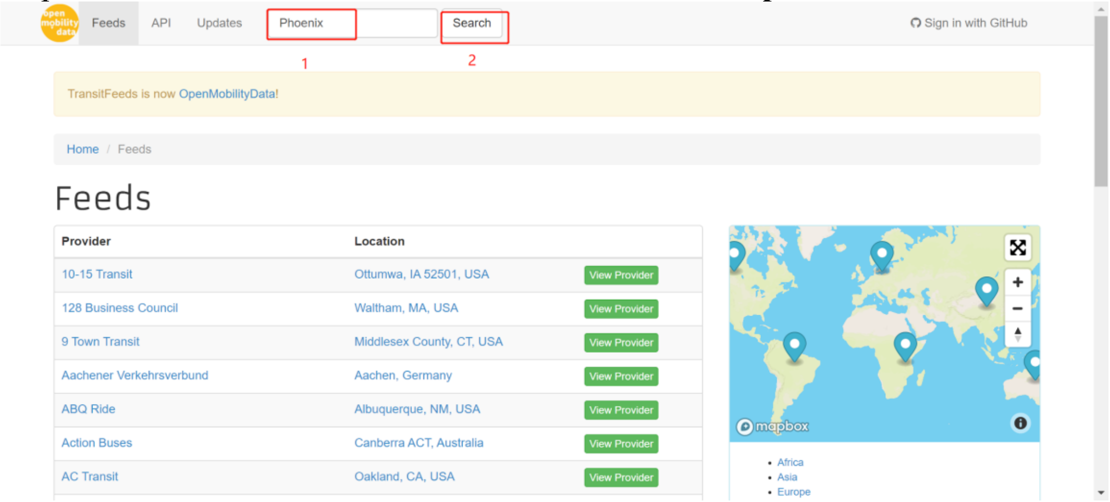
4 Quick Start

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

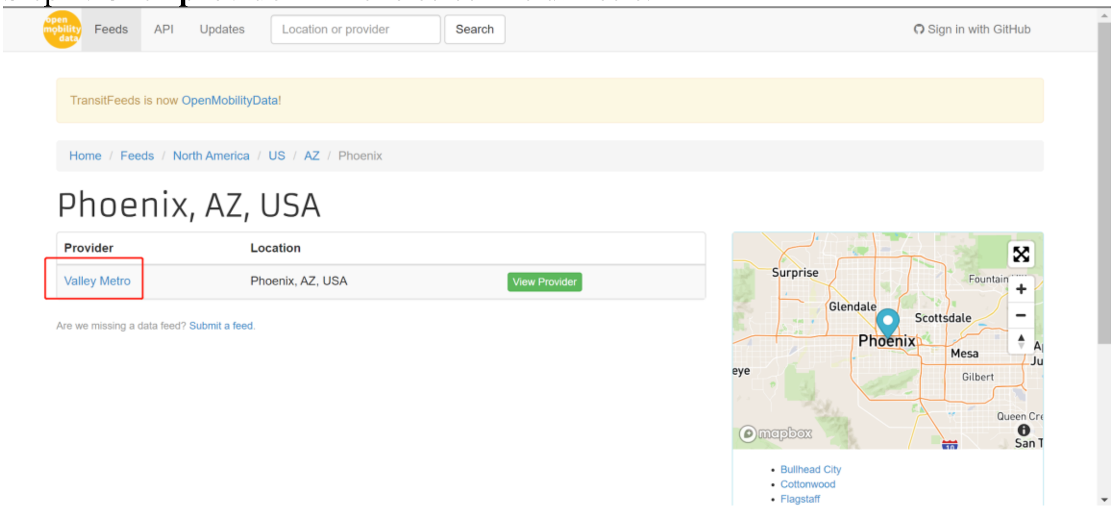
4.1 Download GTFS Data

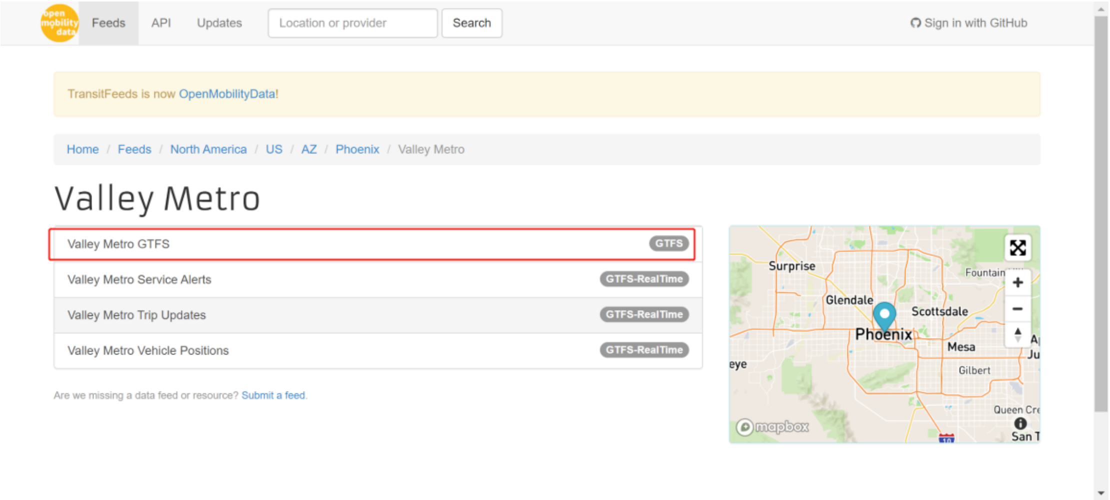
To reduce uncertainties while directly parsing network data from the GTFS, gtfs2gmns uses downloaded .txt files to extract useful network information. As a result, the first step is preparing .txt files of GTFS data.

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

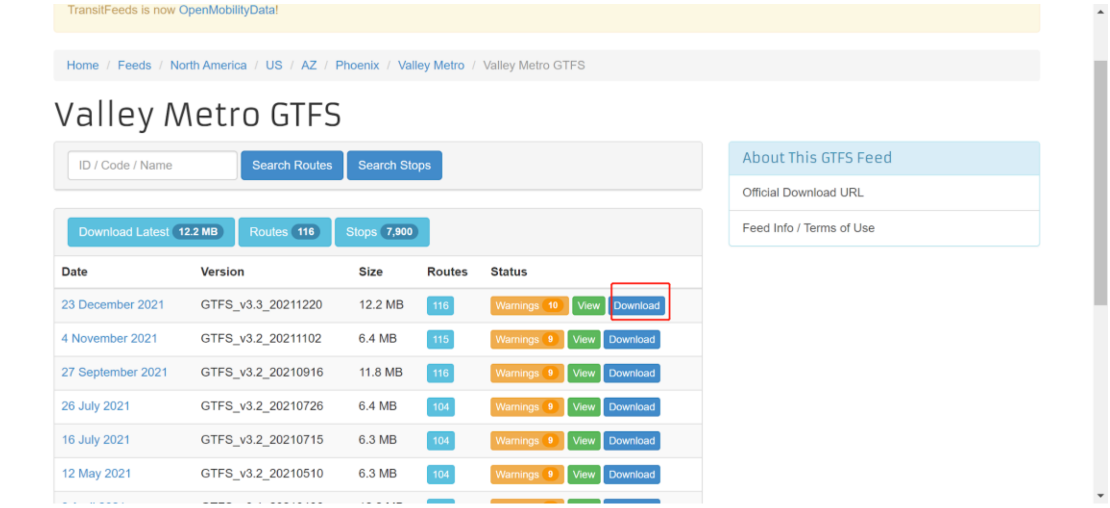


Step 2: Click **provider** name to select the datasets.

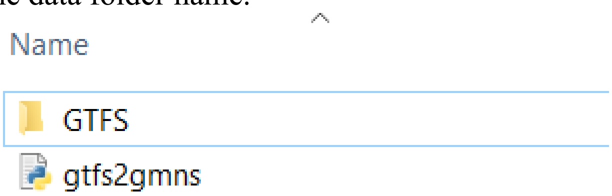




Step 3: Download GTFS datasets.



Step 4: Put the downloaded gtfs.zip into the GTFS data folder. Users can also customize the data folder name.



The GTFS datasets are ready now.

4.2 Parse GTFS Data

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

```python
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.

4.3 Output Networks to CSV

To obtain the output 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
```

Finally, users can use QGIS or Nexta to visualize the network.

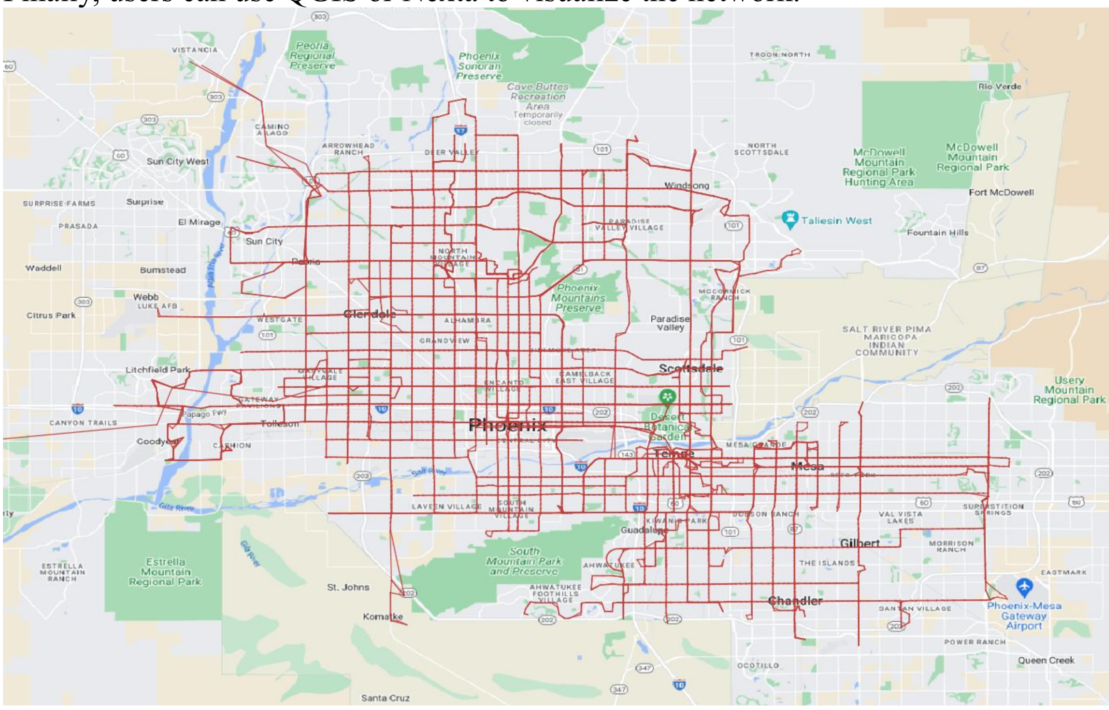


Figure 4: Transit network of Phoenix

5 EXAMPLES

5.1 Washington DC

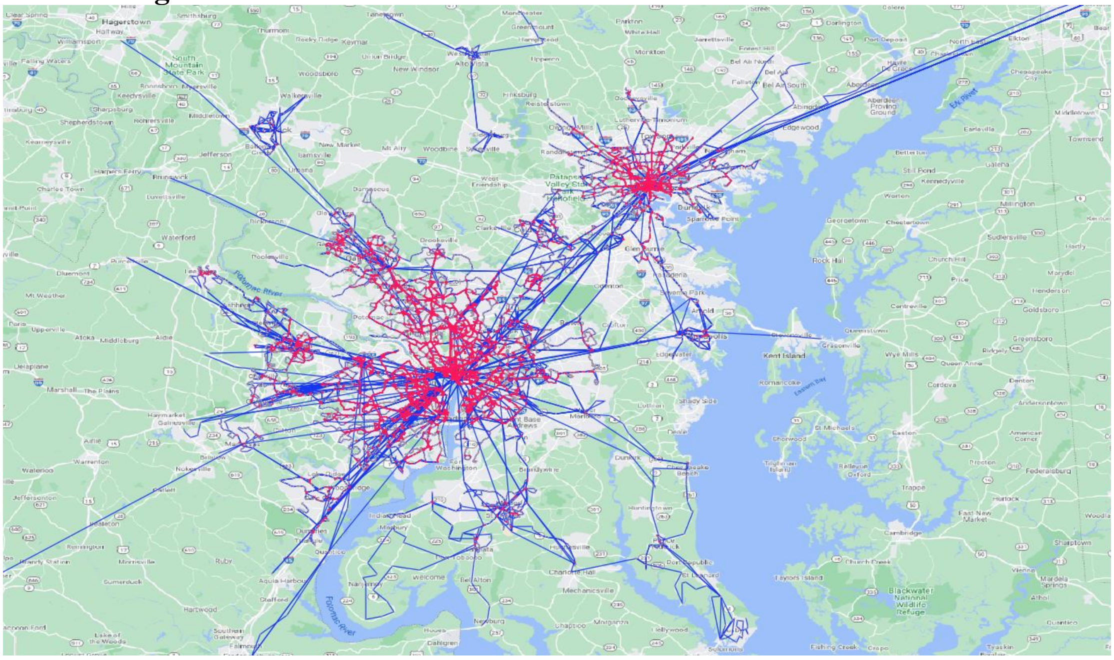


Figure 5: Transit network of Washington DC

6 TRANSIT ASSIGNMENT APPLICATION

6.1 Procedure

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

6.2 Assignment Results

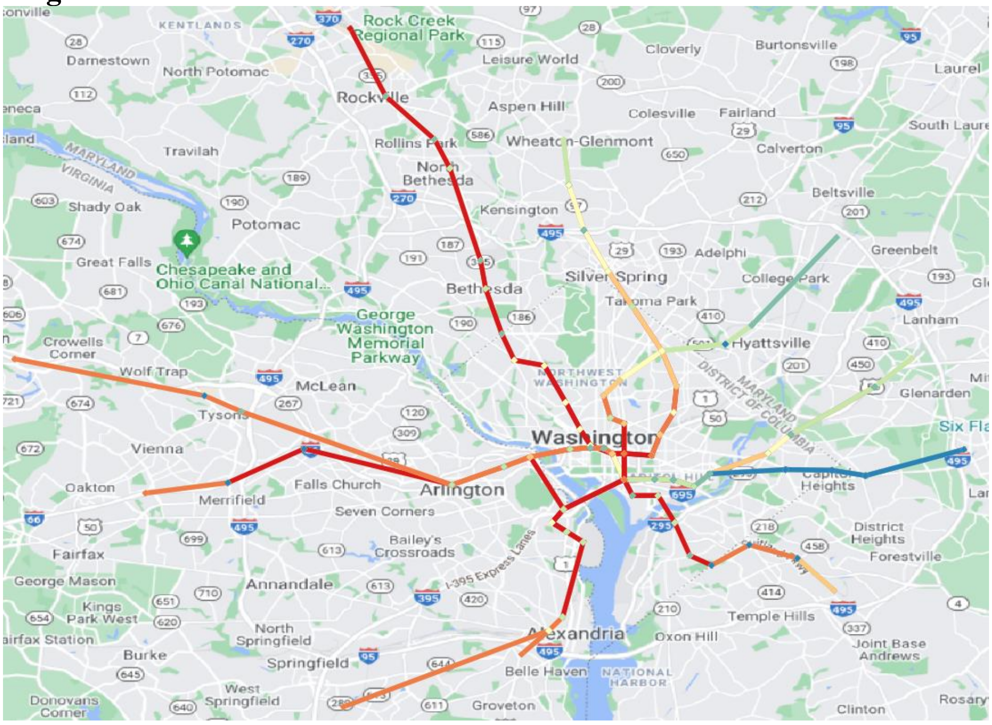


Figure 6 Transit assignment results of metro in Washington DC at AM period

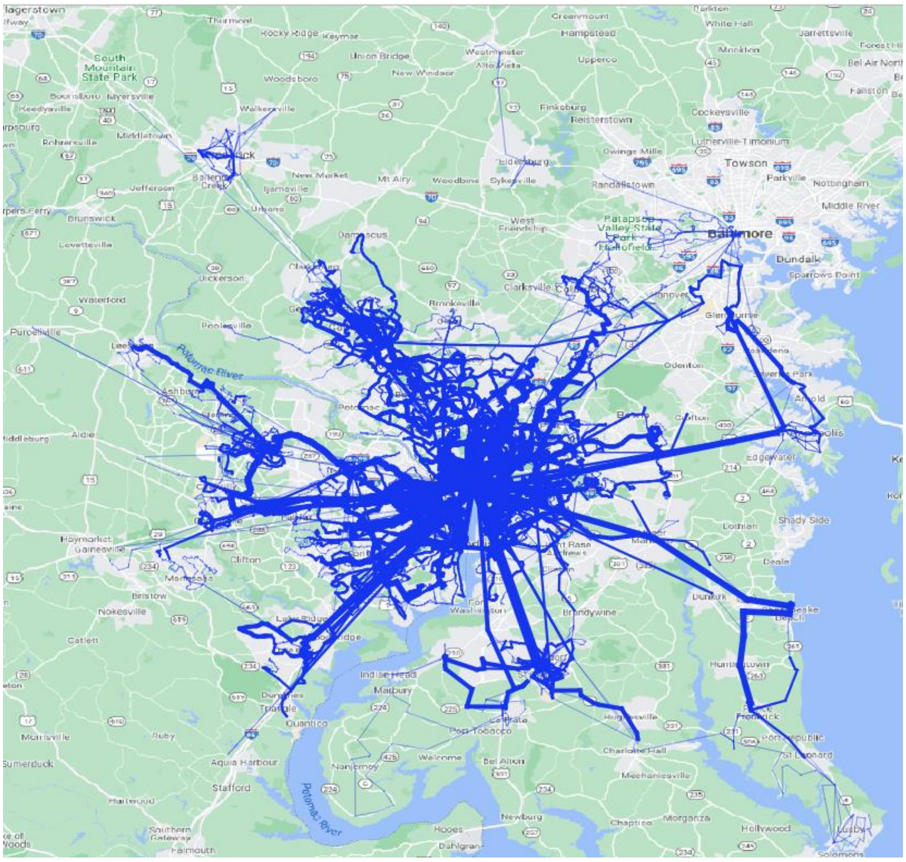


Figure 7 Transit assignment results of bus in Washington DC at AM period