# VEHICULAR MOBILITY MODELLING

MOBMOD COURSE - FALL 2023
PROF. JÉRÔME HÄRRI AND ALI NADAR

## OBJECTIVES

This lab aims at understanding the particular characteristics of vehicular mobility with the help of the traffic simulator SUMO.
We will investigate:

1. How to design a street layout for SUMO
2. How to configure a traffic demand for SUMO
3. How to analyze the traffic output: mobility as well as carbon footprint

This lab is based on Ubuntu and SUMO, and it should be available. Please check by typing **sumo** from a linux terminal.

## GETTING STARTED

We are going to use **SUMO** and you can find the binary in **/usr/bin** and the tools in **/usr/share/sumo**

SUMO is a complex simulator and its documentation can be found at http://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-_Wiki.

In this lab we are going to use:

- **`netgenerate`**: http://sumo.dlr.de/wiki/NETGENERATE
- **`duarouter`**: http://sumo.dlr.de/wiki/DUAROUTER
- **`sumo`**: http:/sumo.dlr.de/wiki/SUMO and **`sumo-gui`**: http://sumo.dlr.de/wiki/SUMO-GUI

The analysis of the simulation results can be done with a tool of your choice.

## SUMO OVERVIEW

SUMO and the SUMO tools can be used with configuration files or directly from command line. The configuration files have a common structure and they are XML files.

In order for SUMO to work, it requires in input the definition of a network and the definition of a traffic demand (and it can be customized with additional xml files containing miscellaneous information). The results of a simulation are generated (on demand) and are in XML format too.

The best way to know what kind of parameters are available in SUMO and all the tools are:

- **`program --help`**
- **`program --save-template program.cfg`**

In the lab folder you can find all the configuration files we are going to use, commented.

## HOW TO DESIGN A REGULAR STREET LAYOUT USING SUMO TOOLS

The network topology in SUMO is defined with nodes and edges in an XML file. With **`netgenerate`** is possible to automatically generate grid, radial (spider), and random networks. Most of the **`netgenerate`** parameters are explained in the abstract network generation wiki page http://sumo.dlr.de/wiki/Networks/Abstract_Network_Generation, but we suggest to read carefully and modify the **netgenerate.cfg** commented file in order to obtain the expected result.

The command can be launched directly by command line with:

```
netgenerate -c configuration_file.cfg
```

however, we recommend you to use and adapt the provided shell script to set SUMO_HOME and SUMO_TOOLS as follows:

```
#!/bin/bash
export SUMO_HOME=/usr
export SUMO_TOOLS=/usr/share/sumo/tools
$SUMO_HOME/bin/netgenerate -c netgen.cfg
```

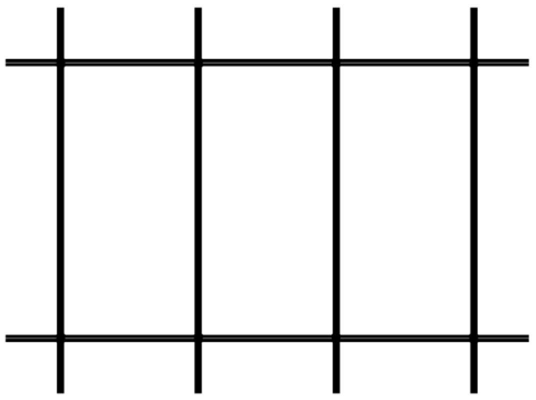
### TIP WITH NETGENERATE AND OTHER SUMO TOOLS

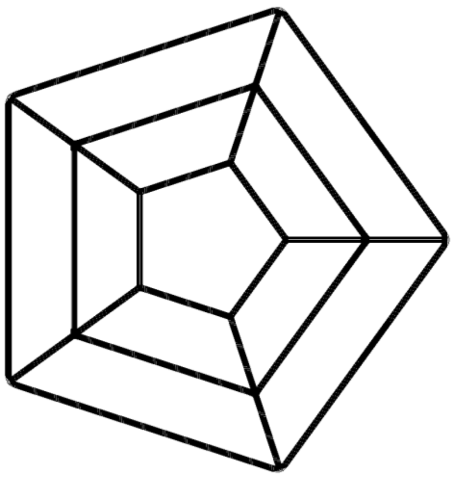
1. The script shown above, can be used for every other SUMO program and tool. We suggest you to use scripts and configuration files, in order to reduce the time you'll spend making changes and tests. SUMO_HOME and SUMO_TOOLS are standard variables that SUMO looks for, hence the export.
2. In the commented configuration file you can find the default value of the parameters. In case of doubt, leave it alone. If you need to build non-standard networks/mobility/additional, those are the parameters to test and change.
3. The commented configuration files contain every possible parameter, even the one that are mutually exclusive. We suggest to create your own configuration files, containing only the parameters you need to modify.
4. The **`sumo-gui`** provides many information simply by clicking on the different network components.
5. The log is the best place to find all the things are not working properly in the file you are generating. The **`<verbose value="true/false"/>`** option can be enabled in every tool.
6. SUMO requires only the net.xml file to work, but you may need to have other files in order to generate additional configuration files. With **`netgenerate`** is possible to ask for nodes, edges and connections in separate files using the parameter **`<plain-output-prefix value="filename"/>`**. These files are easier to read if you need to hand-check the network and the edge file will come in handy during the traffic demand generation. This same parameter is available in many other tools in order to generate the intermediary files.
7. For reproducibility purpose, we suggest the use of a fix random seed (of your choice, but specified) using the parameter **`<seed value="23423"/>`**.

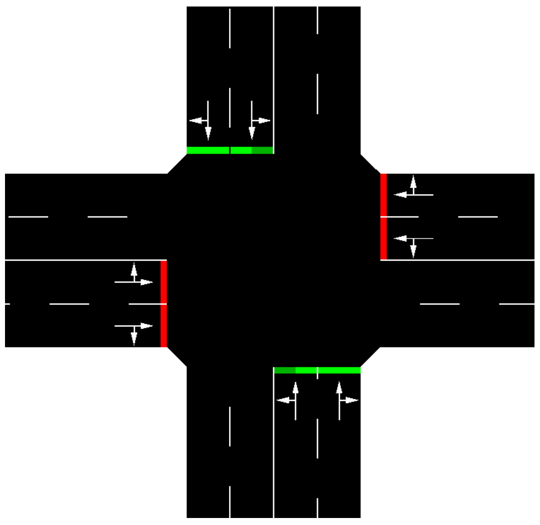
## ASSIGNMENT

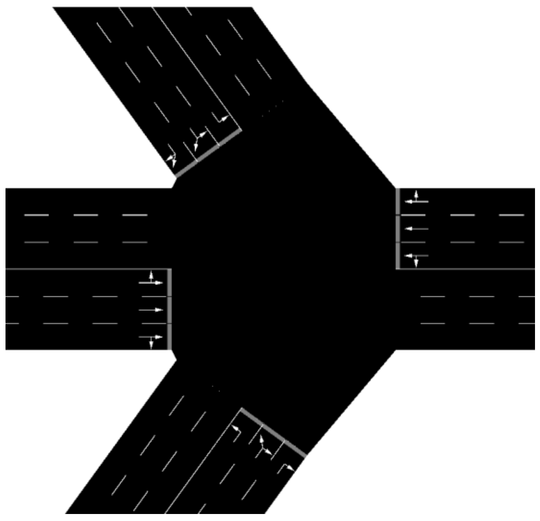
Generate the following networks:

- Grid: 2 by 4 with 2 lanes. Edges of 250m and 500m, and an attached-length of 100m. Default static traffic light at each intersection.
- Radial (spider in SUMO): 5 arms, 3 circles and spaced 250m. Without center, and default right-before-left intersections.









## HOW TO CONFIGURE A TRAFFIC DEMAND USING SUMO TOOLS

The traffic demand is mainly defined using trips and flows. A trip is associated to a vehicle and it has an origin, a destination and a departure time. Flows are defined once for many vehicles, over a period of time, with origin and destination. Trips and flows are used to generate routes for each vehicle in the simulation.

### FLOWS

Flows example:

```
<flows>
    <interval begin="0" end="3600">
       <flow id="0" from="edge0" to="edge1" number="100"/>
       <flow id="1" from="edge1" to="edge2" number="200"/>
       <flow id="2" from="edge2" to="edge3" number="300"/>
       <flow id="3" from="edge3" to="edge4" number="400"/>
    </interval>
</flows>
```

Here we have 4 different flows, over the same interval of 1 hour (3600 seconds) with a different number of vehicles. The frequency of insertion in the simulation can be computed from **number** and the **interval**.

### TRIPS

The alternative is to directly define a trip:

```
<trips>
  <trip id="t" depart="0" from="edge0" to="edge3" via="edge1 edge2"/>
</trips>
```

Where we have one trip starting at time 0 from edge0 to edge3 passing through edge1 and edge2.

An additional way to generate trips is the use of **sumo_0.31.0/tools/randomTrips.py**. The script requires in input the network file and additional parameters used to define the random mobility we need.

For example:

```
sumo/tools/randomTrips.py -n netfile.xml -b 0 -e 3600 --validate -p 20 -o
trips.xml -r routes.xml --vehicle-class=delivery --prefix=delivery -s 23432
```

generates a the random mobility for a network called netfile.xml, from 0 to 3600 seconds, where the vehicles (of class/type delivery) are generated every 20 seconds. It makes sure that the origin-destination pair are valid, and it uses a seed for the rnd generator that is 23432. It generates directly the routes in routes.xml using duarouter (discussed in the next section). Note the **--prefix** parameter, that it does not do anything more than changing the name for the vehicles, and instead of having them enumerated from 0 to X, they will be from prefix.0 to prefix.X. This option is very useful to generate multiple mobility traces with different classes of vehicle and merge them together, without having error with multiple identical ids.

The randomTrips.py documentation is available at http://sumo.dlr.de/wiki/Tools/Trip#randomTrips.py

All the possible classes of vehicles are available at
http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Abstract_Vehicle_Class

### ROUTES

The flows and trips files can be used with **duarouter** to generate the routes based on optimal path routing:

- http://sumo.dlr.de/wiki/Demand/Shortest_or_Optimal_Path_Routing
- http://sumo.dlr.de/wiki/DUAROUTER

The complete configuration file (commented) for **`duarouter`** can be generated with the **`--save-template`** option. The required inputs are the network file and the flow file, specifiable using the **`<input>`** section of the configuration file. The output of **`duarouter`** is the rou.xml file, and it contains all the vehicles with departure time, origin and destination, with a route chosen for each vehicle. In order to optimize the processing, we are going to set the parameters **`remove-loops`** and **`repair`** to **`true`**. For all intent and purposes of this lab, we are going to use the Dijkstra algorithm using the parameter **`<routing-algorithm value="dijkstra"/>.`**

An alternative usage of the trips file is directly with SUMO and the option of automatic routing http://sumo.dlr.de/wiki/Demand/Automatic_Routing.

## SUMO SIMULATION

Once we have a trips file or a route file, we can run it with SUMO and SUMO-GUI. SUMO requires in input the network and trips/routes and it provides various output files. The more important for us are:

- summary-output: saves the aggregated vehicle departure info
- tripinfo-output: saves each vehicle trip info
- emission-output: saves the emission values of each vehicle

The **`sumo-gui`** program can be launched with the following script:

```
#!/bin/bash
export SUMO_HOME=/packages/sumo
export SUMO_TOOLS=$SUMO_HOME/tools
$SUMO_HOME/bin/sumo-gui -c sumo.cfg
```

## RESULT EVALUATION

SUMO output is saved in XML format. Among the tools provided by SUMO, you can find a XML to CSV converter: **sumo/tools/xml/xml2csv.py -i file.xml -o file.xml**

Once you have the results in CSV format, you can evaluate them with your tool of choice.

SUMO provides some visualization tools too:

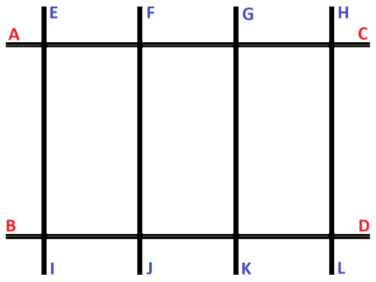
```
sumo/tools/visualization/plot_summary.py -i summary.xml --measure=tag
sumo/tools/visualization/plot_tripinfo_distributions.py -i tripinfo.xml --
measure=tag
```

In case the emission file is too big for visual tools, it can be parsed and aggregate using python and awk. An extreme possibility is to change the **device.emissions.probability** parameter in the sumo configuration file, but is going to complicate the analysis and it has to be properly explained and specified in it.

## ASSIGNMENT WITH GRID-LIKE NETWORK

Create a mobility simulation using the grid network previously generated.

1. Generate the flows file for the grid with the following flows:



- [A,D] [C,B] [E,L] [H,I]: 360 over 3600s
- [D,A] [B,C] [L,E] [I,H]: 240 over 3600s
- [F,D] [G,B] [J,C] [K,A]: 180 over 3600s
- [F,K] [G,J] [K,F] [J,G]: 120 over 3600s

2. Generate the route file using **`duarouter`** and the flows file previously created.
3. Generate the SUMO configuration file using the network file and the route files, saving **summary**, **tripinfo**, and **emissions** files. Set only the initial time, but not the end simulation time. Plot (using your tool of choice) the **`running`** values from the summary file and the **`timeLoss`** values from the tripinfo file and the **CO2** and **fuel** from the emissions file.
4. Now change in the sumo.cfg the following parameters:

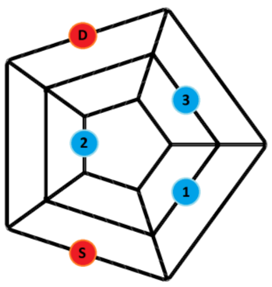
```
<routing>
    <device.rerouting.probability value="1"/>
    <device.rerouting.period value="300"/>
    <device.rerouting.pre-period value="300"/>
</routing>
```

   What does it change in the running vehicles, the time loss, the emissions and the length of the simulation? Discuss it with the support of the additional plots.
5. Regenerate the grid network using the dynamic option for the traffic lights in **`netgenerate`**. Update the route file (with **`duarouter`**) for the new network without changing the flows file.
6. Analyze the new network like in 3. What differences do you see?
7. Analyze the new network like in 4. What differences do you see?

## ASSIGNMENT WITH RADIAL NETWORK

Create a mobility simulation using the radial network (spider web) previously generated and automatic routing option.

1. Generate the trips file with the following trip:



Starting from S and finishing in D, passing by 1, 2 and 3 in order.

2. Generate a configuration file for SUMO that uses the **`device.rerouting`** to compute the missing edges in the route and run the simulation multiple times. Do you see any difference in the route taken?
3. Now change the mobility from trip to flow, over the interval 0-3600, with 3600 vehicles. Once again, run the simulation and observe the mobility generated, how does it differ from the previous one?

Create a mobility simulation using the radial network (spider web) previously generated and the randomTrip.py tool.

1. Generate 1 hour of mobility with the following vehicles:
   a. **passenger** with a frequency of 10 seconds
   b. **motorcycle** with a frequency of 15 seconds
   c. **bus** with a frequency of 120 seconds
   d. **delivery** with a frequency of 60 seconds
2. Generate the configuration file for SUMO by adding all the route files in the **route-files** parameter separated by a space (or ,). Run the simulation, is the mobility congested? Motivate your answer with the support of the data from summary and tripinfo.
3. Change the random mobility to saturate the network. Describe how you did it and analyze the new mobility compared to the previous one.