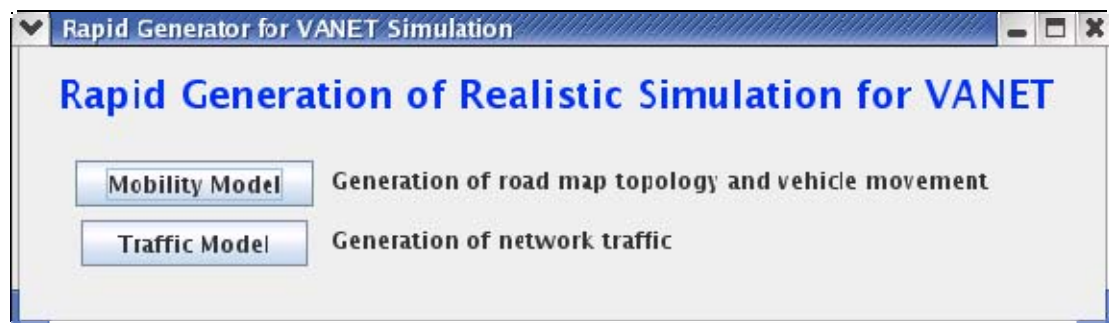


# Rapid Generation of Realistic Simulation for VANET

## Manual

Updated 30 July 2011

As developing VANET in the practical application is too costly, the development will have to rely on simulation.



### Install environment

To use this software, you will need the following software installed on your computer:

- Linux or Windows XP
- Java SDK 1.6 - <http://java.sun.com>
- SUMO version: 0.12.3 - <http://sumo.sourceforge.net>

If SUMO installs in Linux:

- Xerces (XML-parser) - <http://xerces.apache.org/xerces-c/index.html>
- FOX-Toolkit (GUI Toolkit) - <http://www.fox-toolkit.org/>
- PROJ (Cartographic Projections Library):  
<http://www.remotesensing.org/proj/>
- GDAL (Geospatial Data Abstraction Library):  
<http://www.remotesensing.org/gdal/>
- NS2 version: 2.34 (all-in-one) or later - <http://www.isi.edu/nsnam/ns/>

*Note: Refer to each software's website for details on installation. This document only shows how to install the program for specific version. However, according to the versions of Xerces, FOX-Toolkit, PROJ, and GDAL, please check SUMO's requirement.*

In Ubuntu, you need to install extra packages for SUMO.

**A. SUMO part**

```
# sudo apt-get update
```

For more detail, please refer SUMO wiki

<http://sourceforge.net/apps/mediawiki/sumo/index.php?title=LinuxBuild>

**B. NS2 part**

```
# sudo apt-get install build-essential
```

```
# sudo apt-get install tcl8.4 tcl8.4-dev tk8.4 tk8.4-dev
```

```
# sudo apt-get install libxmu-dev libxmu-headers
```

In Window platform, please refer the detail in SUMO wiki.

<http://sourceforge.net/apps/mediawiki/sumo/index.php?title=WindowsBuild>

### ■ Installation Xerces part:

Reference link: <http://xerces.apache.org/xerces-c/build-winunix.html>

For instance at your install path = /home/temp/xerces-c-src\_2\_7\_0

Next set your Xerces-C++ root path as follows:

```
export XERCESCROOT=/home/temp/xerces-c-src_2_7_0
```

One of the common ways to build Xerces-C++ is as follows:

```
cd src/xercesc
./runConfigure -plinux -cgcc -xg++ -minmem -nsocket -tnative -rpthread
```

Build xercesc

```
gmake
make install
```

Finish.

### ■ Installation FOX-Toolkit part:

You can configure simply as:

```
./configure --with-opengl=yes --prefix=$HOME
make
make install
```

Finish.

### ■ Installation PROJ and GDAL part:

You can configure simply as:

```
./configure --prefix=$HOME
make
make install
```

Finish.

## ■ Installation SUMO part:

Reference link: <http://sumo.sourceforge.net/wiki/index.php/LinuxBuild>

You can configure simply as:

```
./configure --with-fox=$HOME --with-proj-gdal=$HOME --with-xerces=$HOME --prefix=$HOME  
make  
make install
```

Enjoin!

Note: If you have gcc version problem, we suggest you input the command:

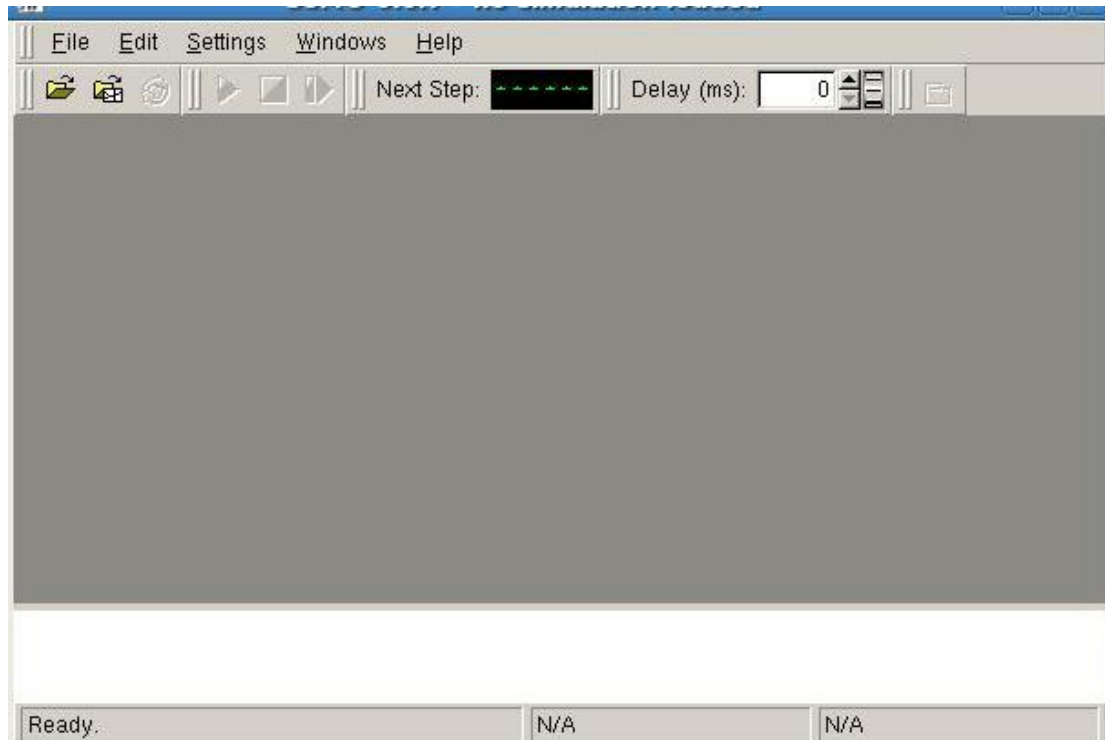
```
CC=gcc34  
CXXFLAGS=-fpermissive ./configure --with-fox=$HOME --with-proj-gdal=$HOME --with-xerces=$H  
OME --prefix=$HOME
```

You can use **GUI SUMO** to check install successes or not.

For instance, We are installation path = /home/temp/sumo-0.9.8

```
cd /home/temp/sumo-0.9.8/src  
./sumo-guisim
```

If you installation successes, you will see under picture.



## ■ Installation NS2 part:

Reference: <http://www.isi.edu/nsnam/ns/ns-build.html#allinone>

When you have download **ns-allinone-2.31.tar.gz**.

```
[root@localhost root]# tar -zxvf ns-allinone-2.31.tar.gz
```

### Install NS2

```
[root@localhost root]# cd ns-allinone-2.31
[root@localhost ns-allinone-2.31]# ./install

=====
* Testing for Cygwin environment
=====

Cygwin not detected, proceeding with regular install.

=====
* Build XGraph-12.1
=====

creating cache ./config.cache
checking for a BSD compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking whether make sets ${MAKE}... yes
checking for working aclocal
.....
```

Install finish.

Now, we must setting realation environment variable, and edit user home directory for .bashrc file. We will additional environment variable including PATH, LD\_LIBRARY\_PATH, and TCL\_LIBRARY.

```
export NS_HOME='pwd'/ns-allinone-2.31
export PATH= $NS_HOME /tcl8.4.14/unix: $NS_HOME /bin: $NS_HOME /tk8.4.14/unix:$PATH
export LD_LIBRARY_PATH=$NS_HOME /otcl-1.13: $NS_HOME /lib:$LD_LIBRARY_PATH
export TCL_LIBRARY=$NS_HOME /tcl8.4.14/library
```

We also suggest edit to ~/.bash\_profile file, and you can avoid every time login and resetting environment variable.

## Building the sources

To build the sources simply compile with "javac \*.java"

To run it simply do "java vanetsim"

## Or you can execute MOVE by jar file without building the sources

To run it simply do "java -jar MOVE.jar"

Note: If your trace is very large, you can input: **Java -Xmx512m -jar MOVE.jar**

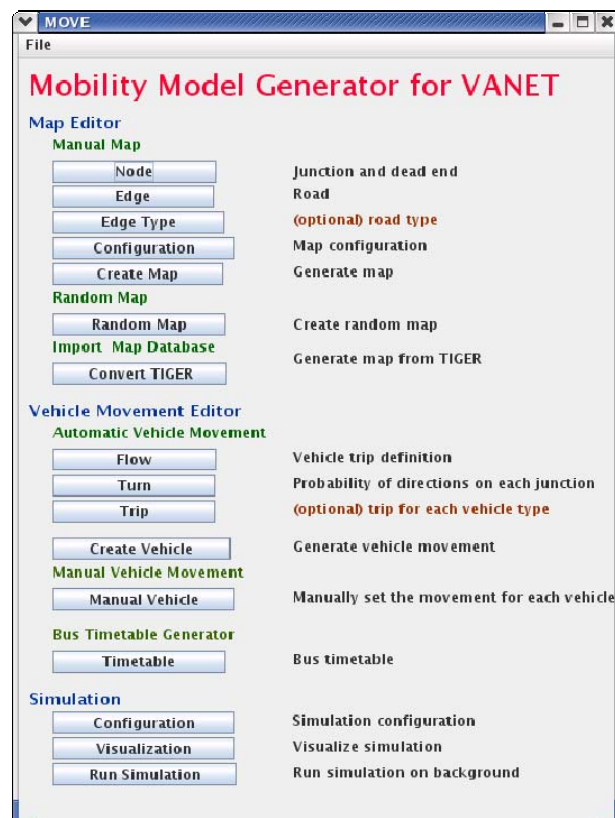
## Software binaries

Ensure that all the paths of the software binaries are exported to the shell. Therefore you are able to call the software from any current working directories. This is important as this software will not be able to set the binary paths.

## How to use this software

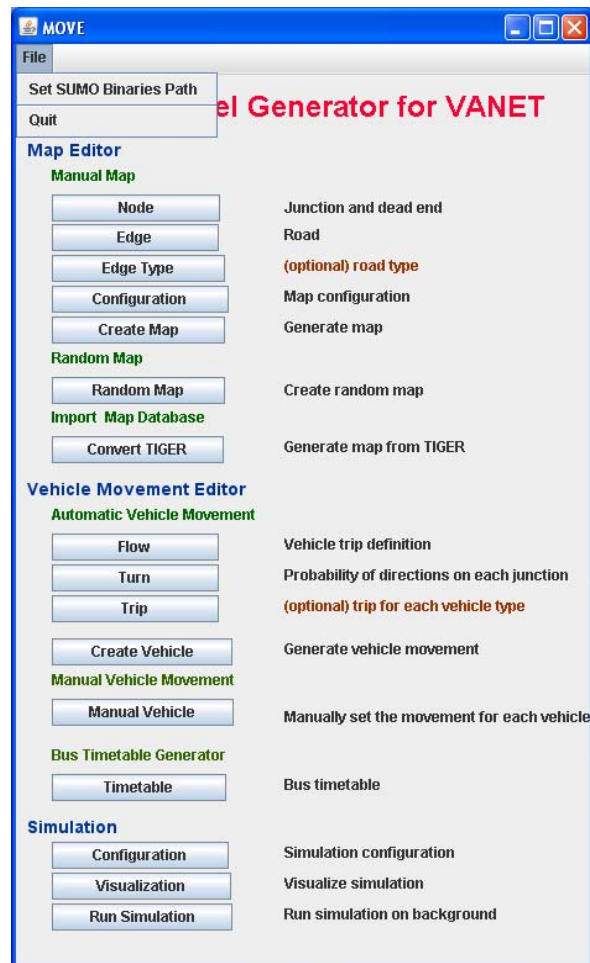
### I. Mobility Generation

This part of the software (called MOVE - MObility model generator for VEhicular networks) will generate the mobility model created by SUMO. Firstly select "Mobility Model" on the main top level menu.



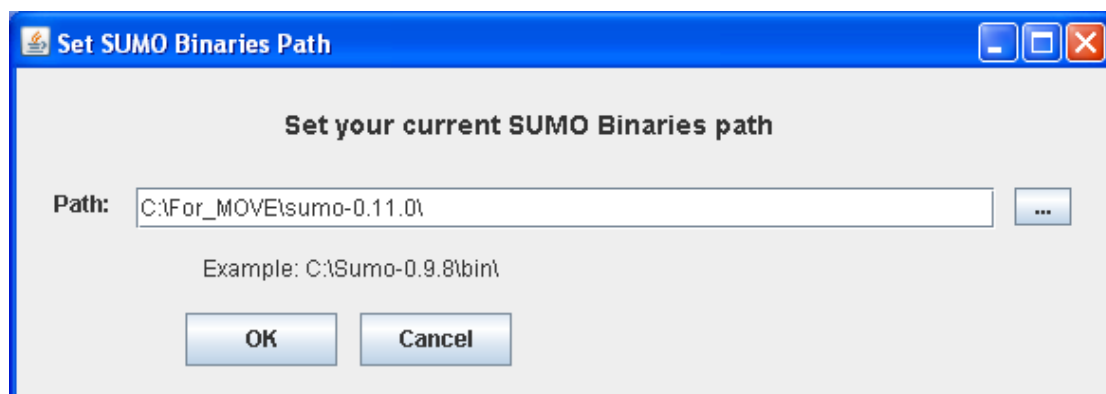
**[Set SUMO Binaries Path only use on windows]**

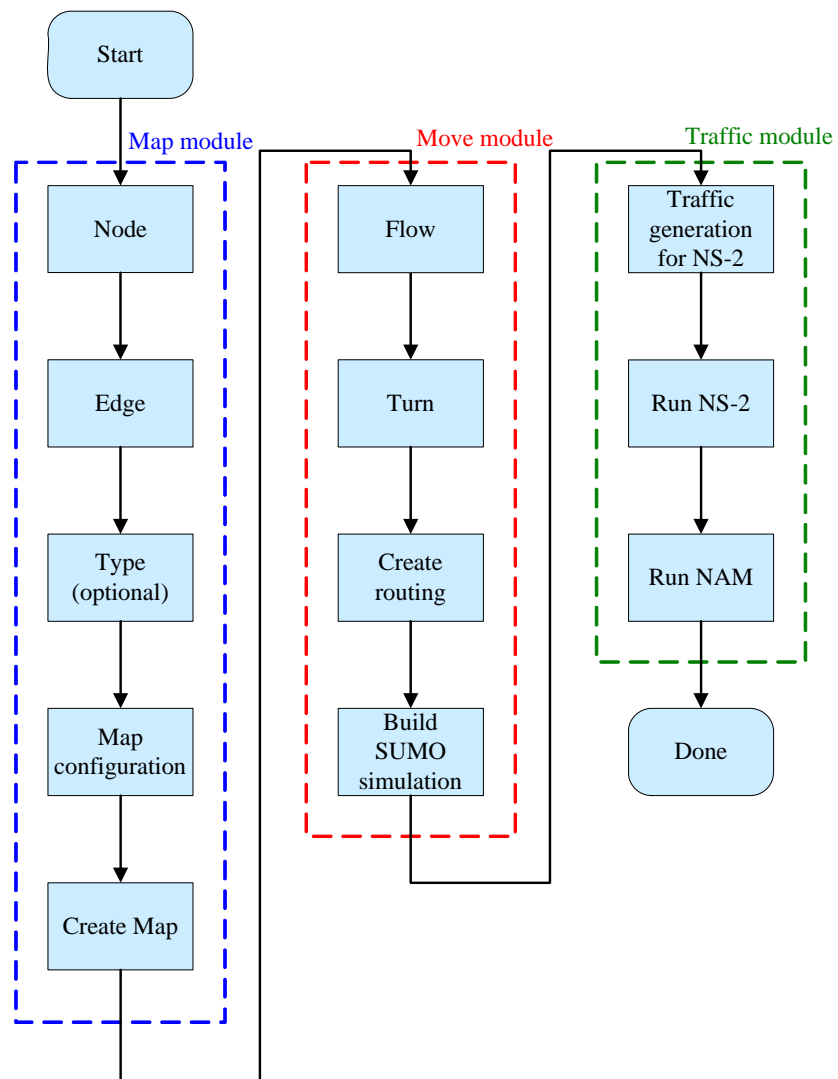
Step 1: To click “File” then choose “Set SUMO Binaries Path”.



Step 2: Please you set SUMO binaries path. For instance, we set sumo path that is located at “C:\For\_MOVE\sumo-0.12.0\

Hint: You only set the path once, MOVE will save the configuration.





This is roadmap in the VANET manual. It is main let user friendly using and practice VANET.



We use example and step-by-step description in figure 1.

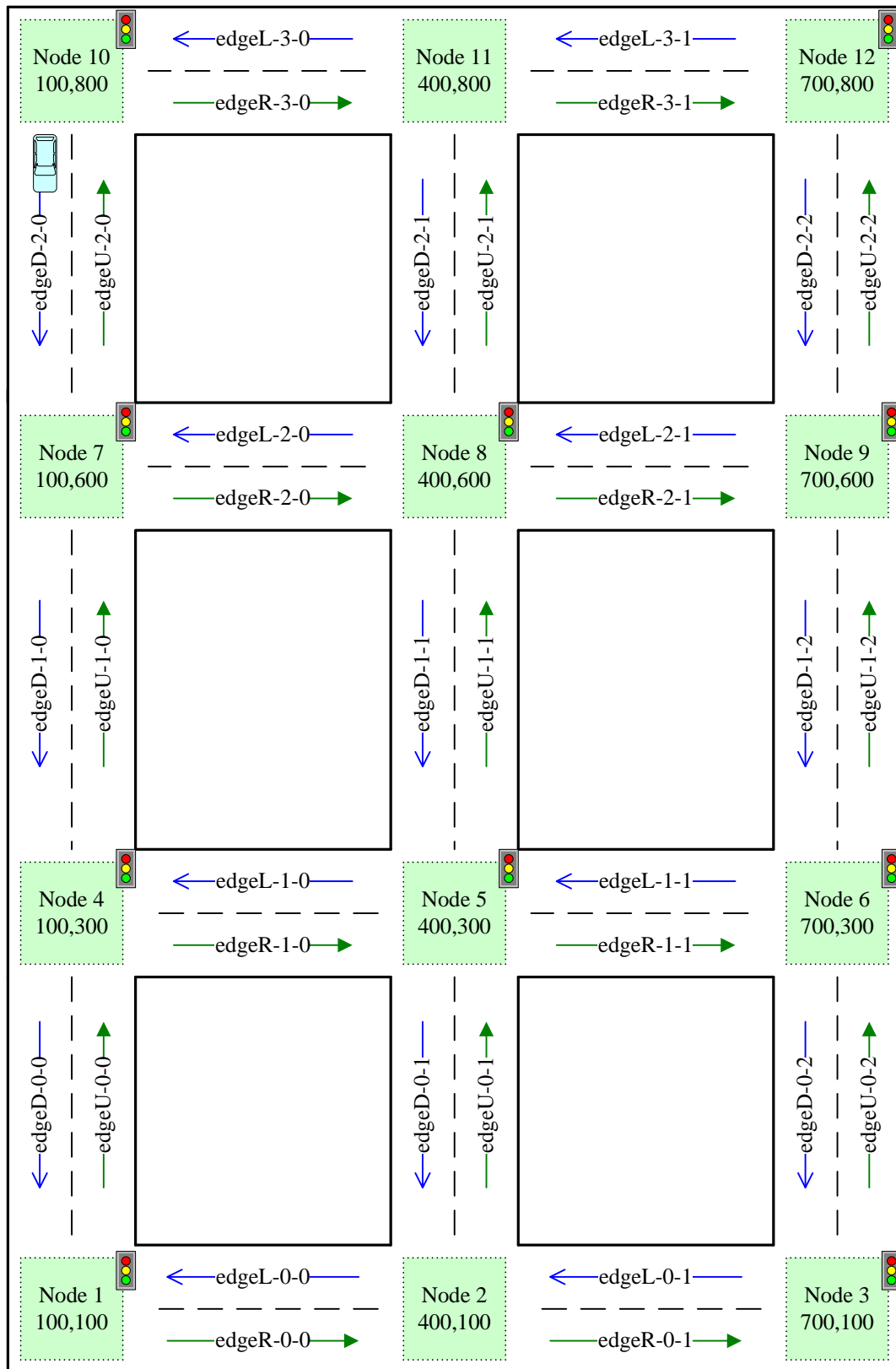
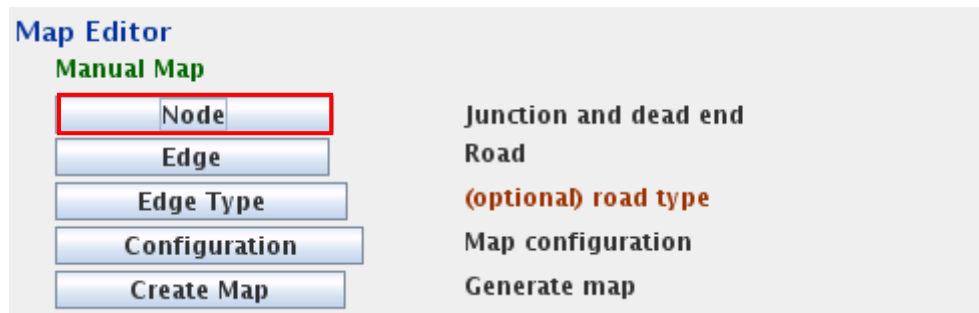


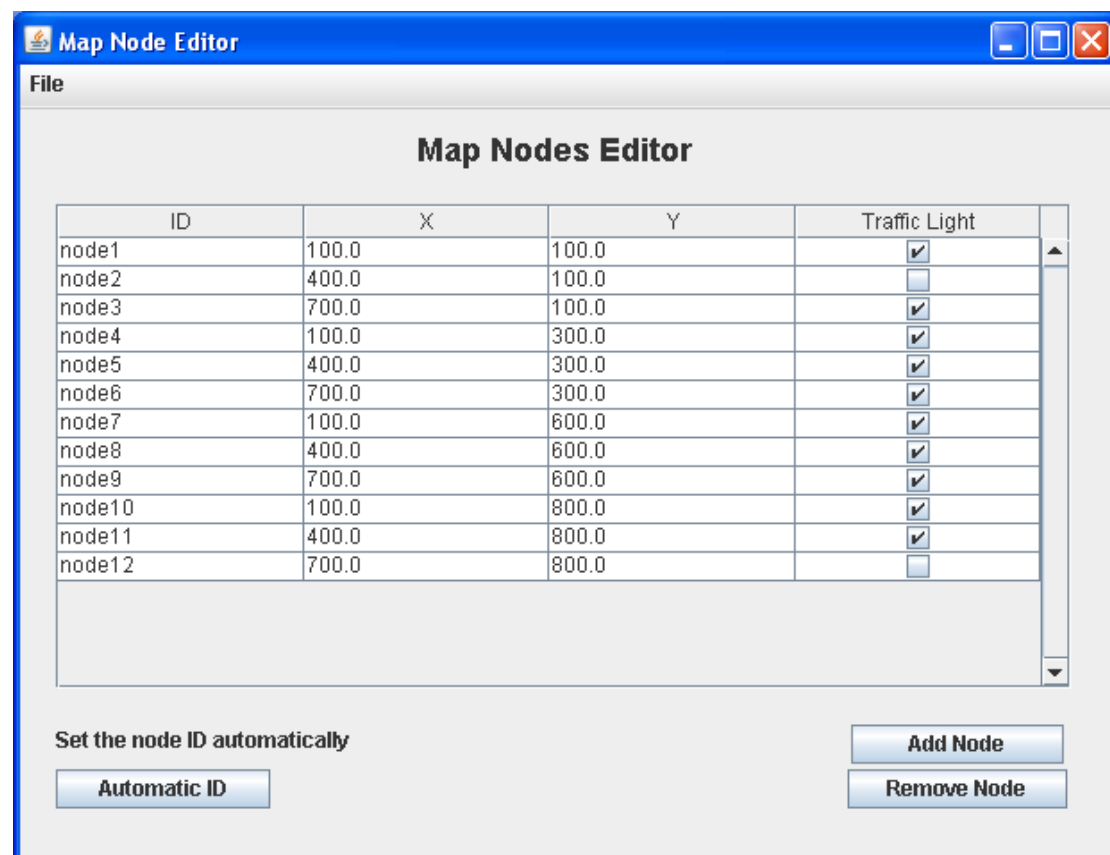
Fig.1. Example

## I.i Map generation

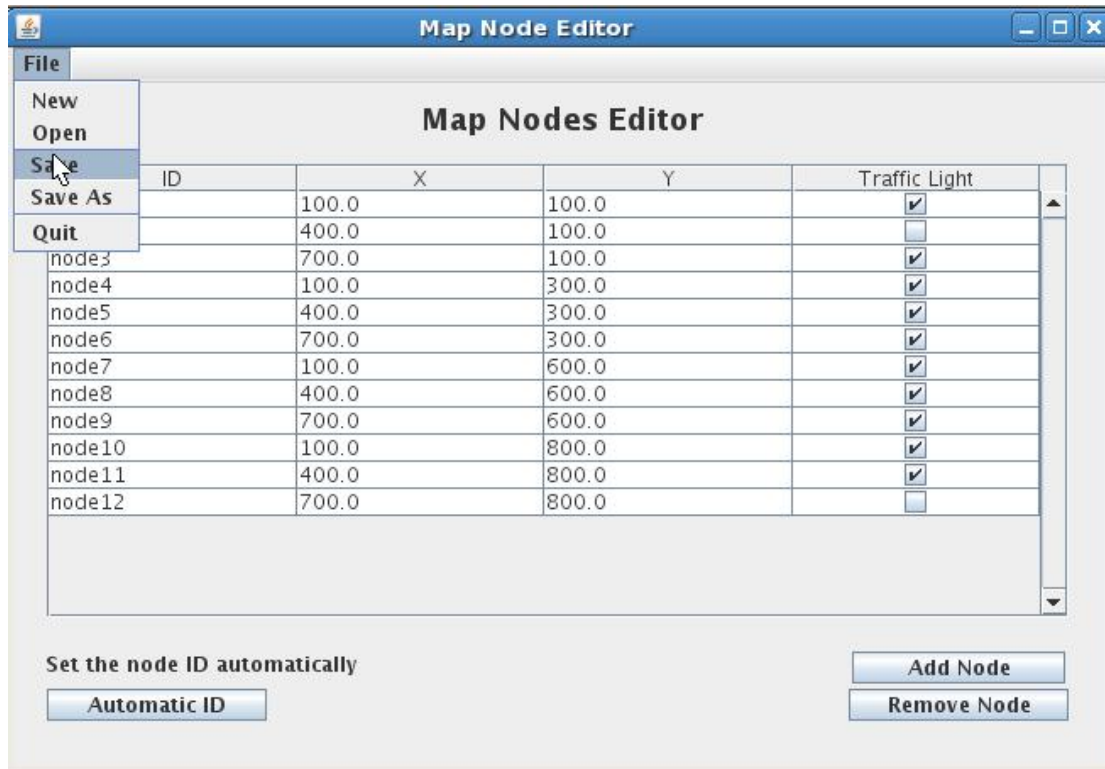
### Step 1: Manually create your own map nodes (ex\_NODE.nod.xml)



First, select “Node” from MOVE main menu. This is simply where all the map nodes are.



You can set node follow upper legend.



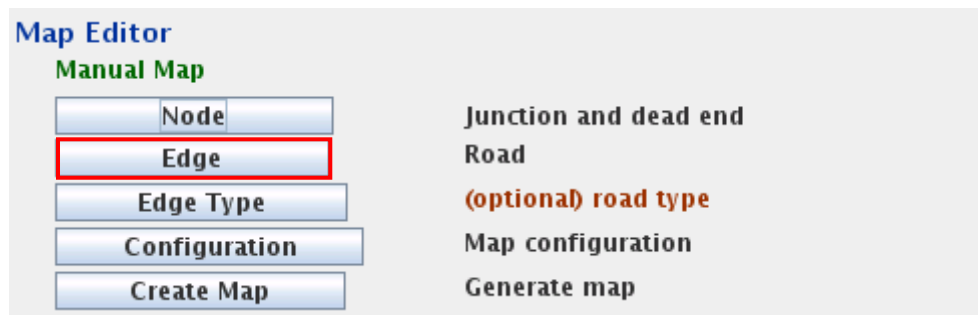
Then select 'File' -> 'save' or 'save as' when you are done editing.

Save the file as <name>.nod.xml (example.nod.xml).

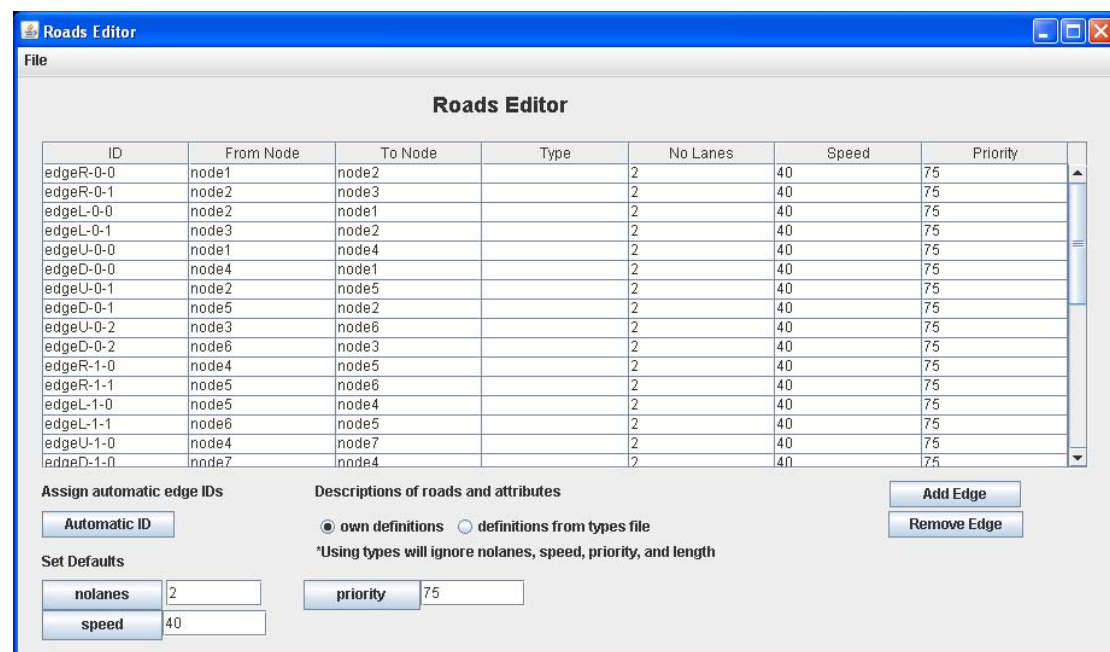
The example detail in ex\_NODE.nod.xml:

```
<nodes>
<node id="node1" x="100.0" y="100.0" type="traffic_light"/>
<node id="node2" x="400.0" y="100.0" type="priority"/>
<node id="node3" x="700.0" y="100.0" type="traffic_light"/>
<node id="node4" x="100.0" y="300.0" type="traffic_light"/>
<node id="node5" x="400.0" y="300.0" type="traffic_light"/>
<node id="node6" x="700.0" y="300.0" type="traffic_light"/>
<node id="node7" x="100.0" y="600.0" type="traffic_light"/>
<node id="node8" x="400.0" y="600.0" type="traffic_light"/>
<node id="node9" x="700.0" y="600.0" type="traffic_light"/>
<node id="node10" x="100.0" y="800.0" type="traffic_light"/>
<node id="node11" x="400.0" y="800.0" type="traffic_light"/>
<node id="node12" x="700.0" y="800.0" type="priority"/>
</nodes>
```

## Step 2: Manually create your own map edge (ex\_EDGE.edg.xml)



First, select “Edge” from MOVE main menu. This is similar to node editor. This is where you can specify all the roads (a road will create a connection between two nodes created previously).



You can set edge follow upper legend.

Save the file as <name>.edg.xml (example.edg.xml).

The example detail in ex\_EDGE.edg.xml:

```
<edges>
<edge id="edgeR-0-0" fromnode="node1" tonode="node2" priority="75"
nolanes="2" speed="40" />
<edge id="edgeR-0-1" fromnode="node2" tonode="node3" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-0-0" fromnode="node2" tonode="node1" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-0-1" fromnode="node3" tonode="node2" priority="75"
nolanes="2" speed="40" />
```

```

<edge id="edgeU-0-0" fromnode="node1" tonode="node4" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-0-0" fromnode="node4" tonode="node1" priority="75"
nolanes="2" speed="40" />
<edge id="edgeU-0-1" fromnode="node2" tonode="node5" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-0-1" fromnode="node5" tonode="node2" priority="75"
nolanes="2" speed="40" />
<edge id="edgeU-0-2" fromnode="node3" tonode="node6" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-0-2" fromnode="node6" tonode="node3" priority="75"
nolanes="2" speed="40" />

```

```

<edge id="edgeR-1-0" fromnode="node4" tonode="node5" priority="75"
nolanes="2" speed="40" />
<edge id="edgeR-1-1" fromnode="node5" tonode="node6" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-1-0" fromnode="node5" tonode="node4" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-1-1" fromnode="node6" tonode="node5" priority="75"
nolanes="2" speed="40" />

```

```

<edge id="edgeU-1-0" fromnode="node4" tonode="node7" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-1-0" fromnode="node7" tonode="node4" priority="75"
nolanes="2" speed="40" />
<edge id="edgeU-1-1" fromnode="node5" tonode="node8" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-1-1" fromnode="node8" tonode="node5" priority="75"
nolanes="2" speed="40" />
<edge id="edgeU-1-2" fromnode="node6" tonode="node9" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-1-2" fromnode="node9" tonode="node6" priority="75"
nolanes="2" speed="40" />

```

```

<edge id="edgeR-2-0" fromnode="node7" tonode="node8" priority="75"
nolanes="2" speed="40" />

```

```

<edge id="edgeR-2-1" fromnode="node8" tonode="node7" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-2-0" fromnode="node8" tonode="node9" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-2-1" fromnode="node9" tonode="node8" priority="75"
nolanes="2" speed="40" />

<edge id="edgeU-2-0" fromnode="node7" tonode="node10" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-2-0" fromnode="node10" tonode="node7" priority="75"
nolanes="2" speed="40" />
<edge id="edgeU-2-1" fromnode="node8" tonode="node11" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-2-1" fromnode="node11" tonode="node8" priority="75"
nolanes="2" speed="40" />
<edge id="edgeU-2-2" fromnode="node9" tonode="node12" priority="75"
nolanes="2" speed="40" />
<edge id="edgeD-2-2" fromnode="node12" tonode="node9" priority="75"
nolanes="2" speed="40" />

<edge id="edgeR-3-0" fromnode="node10" tonode="node11" priority="75"
nolanes="2" speed="40" />
<edge id="edgeR-3-1" fromnode="node11" tonode="node10" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-3-0" fromnode="node11" tonode="node12" priority="75"
nolanes="2" speed="40" />
<edge id="edgeL-3-1" fromnode="node12" tonode="node11" priority="75"
nolanes="2" speed="40" />
</edges>

```

### Step 3: Map configuration editor (ex\_Map.netc.cfg)



After that, choose “Configuration” from MOVE main menu.

Select you are previously created/edited ‘**nod**’ and ‘**edg**’ files location. There are some defaults here. They only work when you did not fill the properties in the “**Edge**” editor or the types file is missing.

Save the file as <name>.netc.cfg (**ex\_Map.netc.cfg**)

Example detail in **ex\_Map.netc.cfg**:

<configuration>

<input

*xml-node-files="C:\For\_MOVE\MOVE\_example\ex\_NODE.nod.xml"*

*xml-edge-files="C:\For\_MOVE\MOVE\_example\ex\_EDGE.edg.xml"*

*xml-connection-files=""*

*type-file=""*

/>

<output

*output-file="C:\For\_MOVE\MOVE\_example\ex\_Map.net.xml"*

/>

<defaults

*type="Unknown"*

*lanenumber="1"*

*speed="40"*

```

        priority="75"
        capacity-norm=""
    />

    <reports
        print-options="false"
    />

    <processing
        speed-in-kmh="false"
        no-turnarounds="false"
        remove-geometry="x"
    />

</configuration>

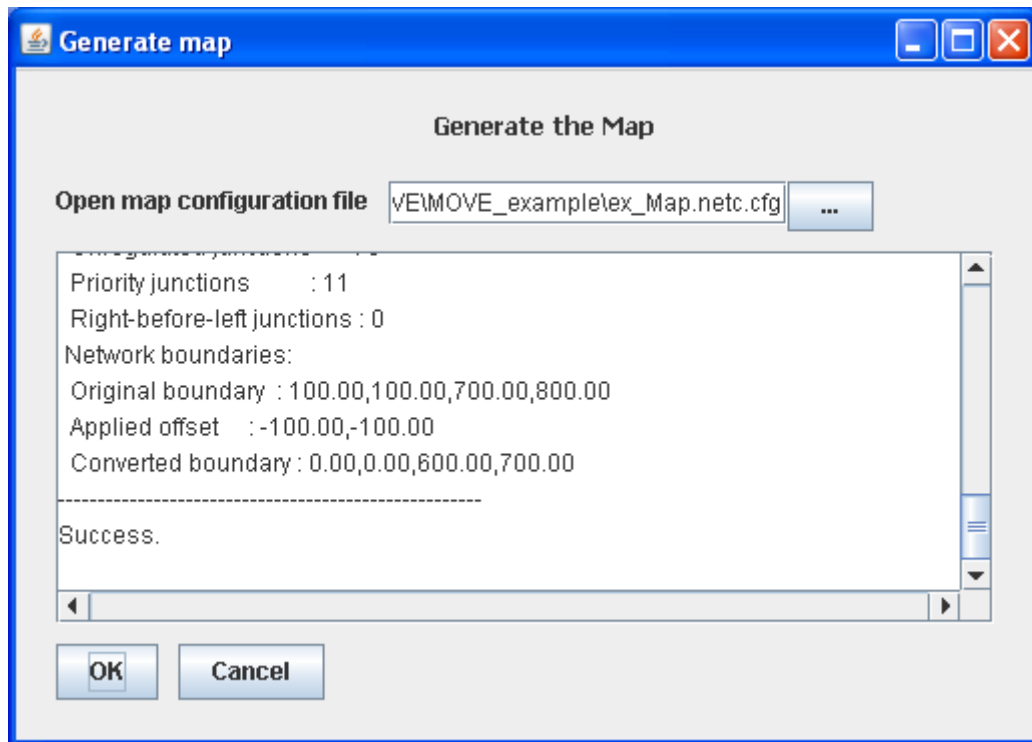
```

#### Step 4: Generate the Map (ex\_Map.net.xml)



Finally, select “Create Map” from MOVE main menu.

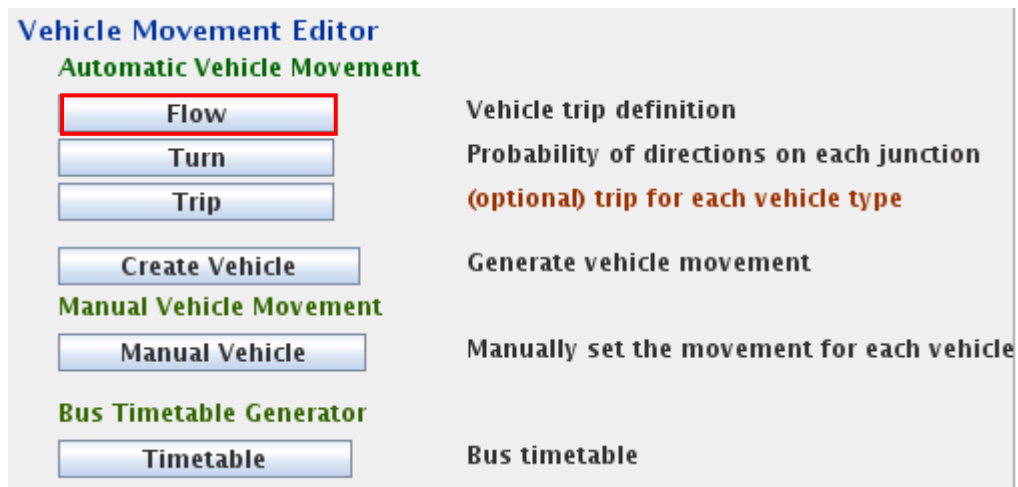




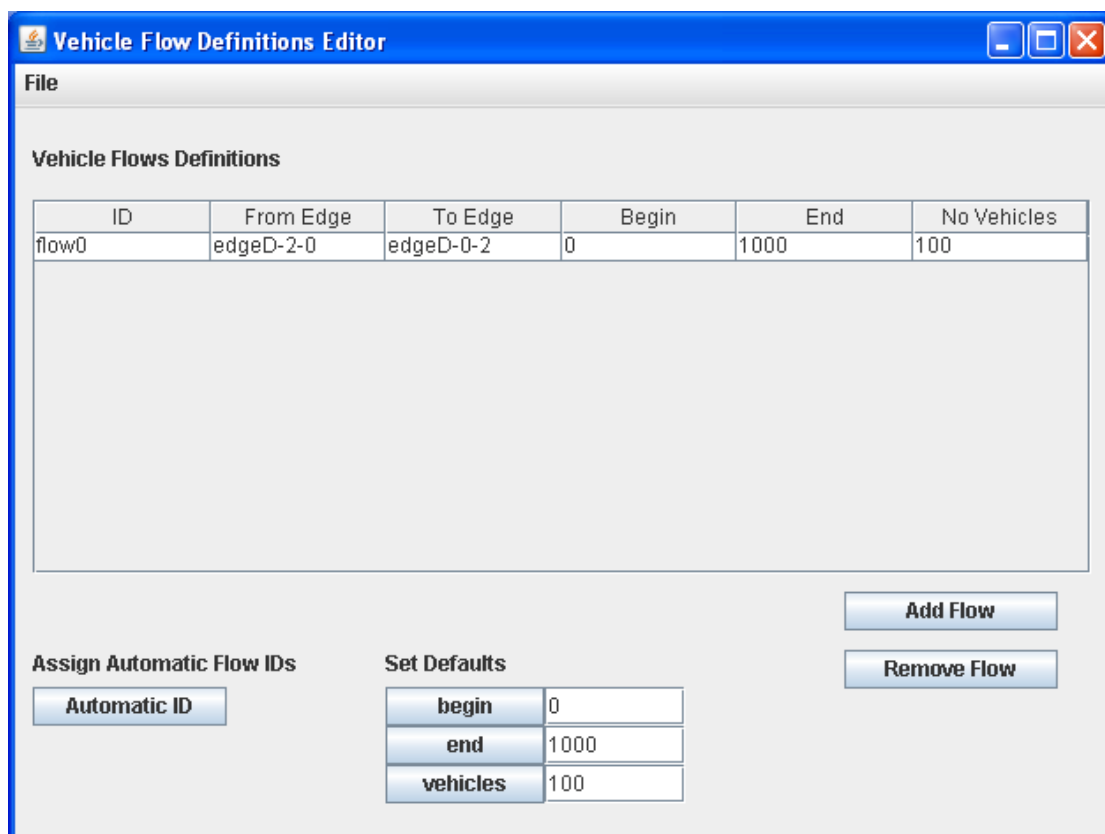
Simply select the netc file (**ex\_Map.netc.cfg**) and click OK. A <name>.net.xml (**ex\_Map.net.xml**) will be automatically generated. This is your map file.

## I.ii Vehicle movements generation

### Step 1: Using flow definition (ex\_FLOW.flow.xml)



Simply select “Flow” from MOVE main menu. This editor will specify the groups of vehicle movements flow on simulation.



You can set flow follow upper legend.

Using these groups of flows, movements of the vehicles will be generated between the two specified edges. For example if you have:

edge1 (connected to node 1 and 2)

edge2 (connected to node 2 and 3)

edge3 (connected to node 3 and 4)

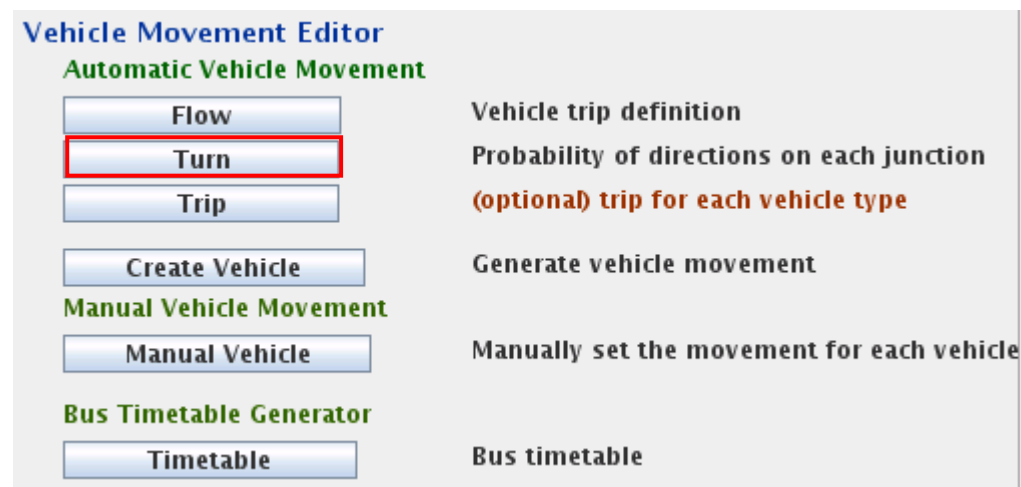
If you want a car to move from node 1 to 2 to 3 (1-2-3), you must assign the flow to be from edge1 to edge3 (NOT edge1 to edge2). If you assign it from edge1 to edge2, it will only move halfway (from 1- 2). Just remember that the movement pattern is all ways from the beginning node of a source edge to the beginning node of a destination edge.

Save the file as <name>.flow.xml (**ex\_FLOW.flow.xml**). You can also open and edit an existing .flow.xml file.

Example detail in **ex\_FLOW.flow.xml**:

```
<flows>
<flow id="flow0" from="edgeD-2-0" to="edgeD-0-2" begin="0" end="1000"
no="100" />
</flows>
```

## Step 2: Using junction turning ratio (not implement in the example)



Simply select “Turn” from MOVE main menu. This is when you to specify a turning ratio for each junction. You must also create flow definitions as previously explained for the vehicle movements. **In the editor, you must group percentages together so they sum up as 1.0 (100%)**

Note: If you want setting same probability in MOVE, you can try input as follows:

```
sumo-jtrrouter -n ex_Map.net.xml -f ex_FLOW.flow.xml -t ex_TURN.turn.xml -v -o
ex_ROU.rou.xml -b 0 -e 1000 --continue-on-unbuild --turn-defaults=30,30,40
```

*The probability format is “turning left, go straight, and turning right”*

**Junction Turning Probability Editor**

File

**Junctions Turning Ratios Editor**

Begin	End	From Edge	To Edge	Percentage
0	1000	edgeR-2-0	edgeR-2-1	0.4
0	1000	edgeR-2-0	edgeU-2-1	0.4
0	1000	edgeR-2-0	edgeD-1-1	0.2
0	1000	edgeD-1-1	edgeL-1-0	0.4
0	1000	edgeD-1-1	edgeR-1-1	0.2
0	1000	edgeD-1-1	edgeD-0-1	0.4

**Set Defaults**

begin

end

**Add Turn**

**Remove Turn**

You can set junctions turning ratios follow upper legend.

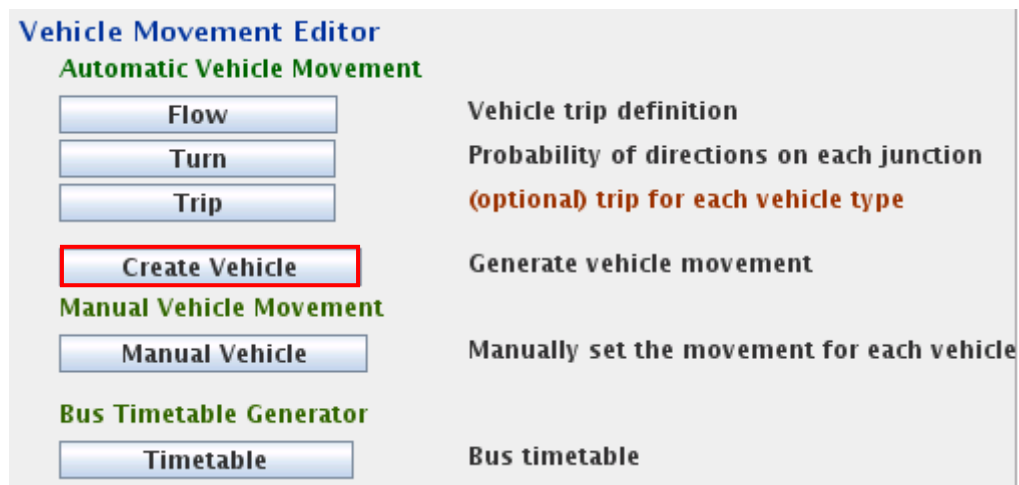
Example detail in (ex\_TURN.turn.xml)

```

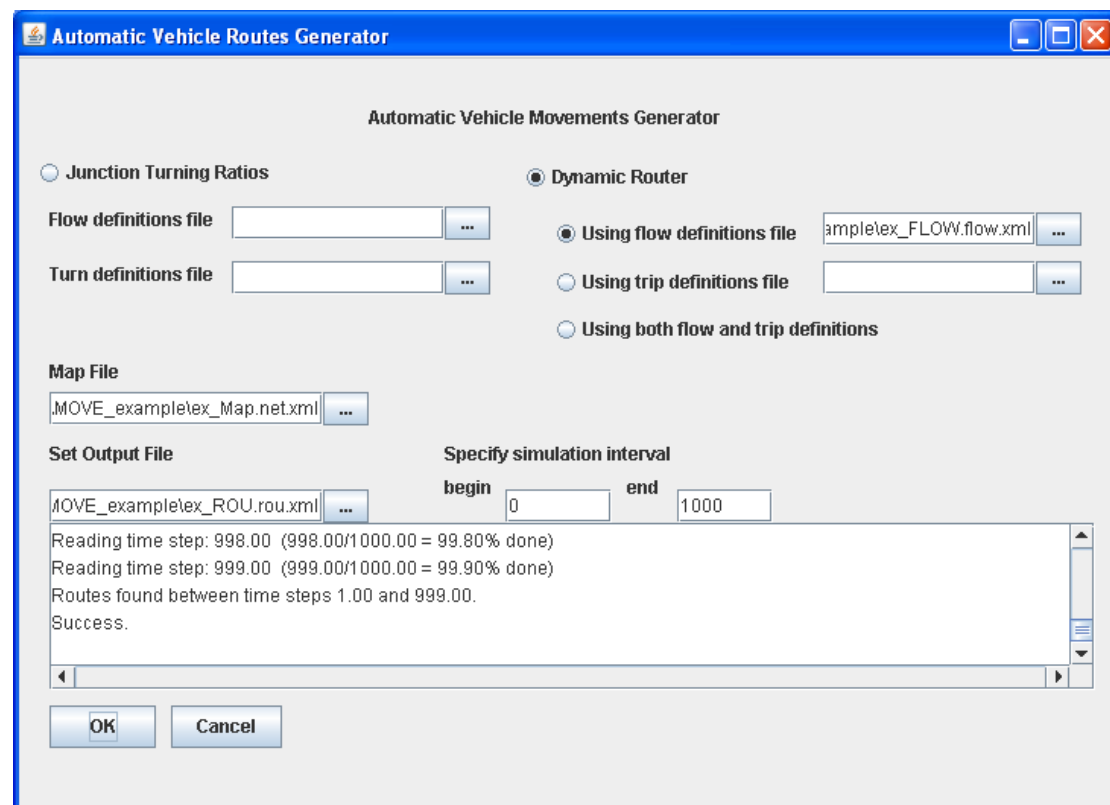
<turn-defs>
<interval begin="0" end="1000">
  <fromedge id="edgeR-2-0">
    <toedge id="edgeR-2-1" probability="0.4"></toedge>
    <toedge id="edgeU-2-1" probability="0.4"></toedge>
    <toedge id="edgeD-1-1" probability="0.2"></toedge>
  </fromedge>
  <fromedge id="edgeD-1-1">
    <toedge id="edgeL-1-0" probability="0.4"></toedge>
    <toedge id="edgeR-1-1" probability="0.2"></toedge>
    <toedge id="edgeD-0-1" probability="0.4"></toedge>
  </fromedge>
</interval>
</turn-defs>

```

### Step 3: Automatic vehicle movements (ex\_ROU.rou.xml)



Simply select “Create Vehicle” from MOVE main menu. This will simply create a number of vehicles at the start of simulation. For our example, we use junction turning ratios function.

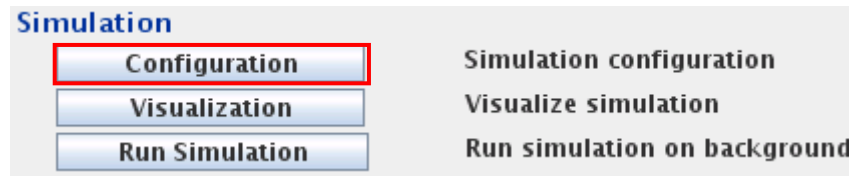


Select your previously created map file (e.g. **ex\_Map.net.xml**). Specify your output file location and name it as <name>.rou.xml (**ex\_ROU.rou.xml**). Set the beginning and end time of simulation. Finally click OK. The rou.xml file will be automatically generated.

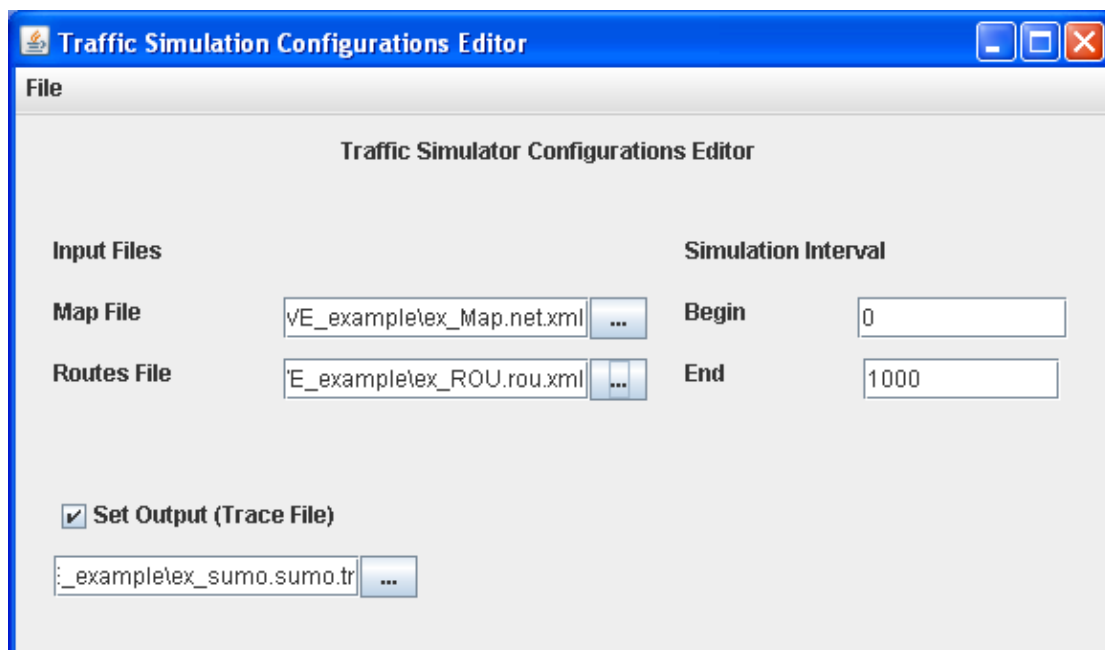
NOTE: this starting number of vehicles is the only way for generating movements for

random maps since the edge and nodes are unknown. May you occur error in sometime (e.g. short routing), when your map is small.

#### Step 4: Simulation setup (ex\_SUMOTRACE.sumo.tr and ex\_SUMO.sumo.cfg)



After the map and movement is complete, you will need to specify the configurations of the simulation. Select “Configuration” at the bottom on MOVE main menu.



Specify the <name>.net.xml (i.e. **ex\_Map.net.xml**) and <name>.rou.xml (i.e. **ex\_ROU.rou.xml**) location and specify the beginning and end time of simulation. If you want to create the trace file, don't forget to check the checkbox and specify your trace output name (e.g. <name>.**sumo.tr**). Then save the file as <name>.sumo.cfg (i.e. **ex\_SUMO.sumo.cfg**)

Example detail in (ex\_SUMO.sumo.cfg)

<configuration>

<input

net-file="C:\For\_MOVE\MOVE\_example\ex\_Map.net.xml"

route-files="C:\For\_MOVE\MOVE\_example\ex\_ROU.rou.xml"

additional-files=""

```

    junction-files=""
  />

  <output
    netstate-dump="C:\For_MOVE\MOVE_example\ex_sumo.sumo.tr"
    tripinfo-output="output-tripinfos.xml"
    emissions-output="output-emissions.xml"
    vehroute-output="output-vehroutes.xml"
  />

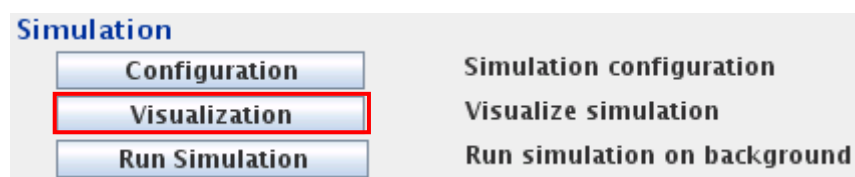
  <time
    begin="0"
    end="1000"
    time-to-teleport="-1"
    srand="23423"
    route-steps="-1"
  />

  <reports
    print-options="false"
  />

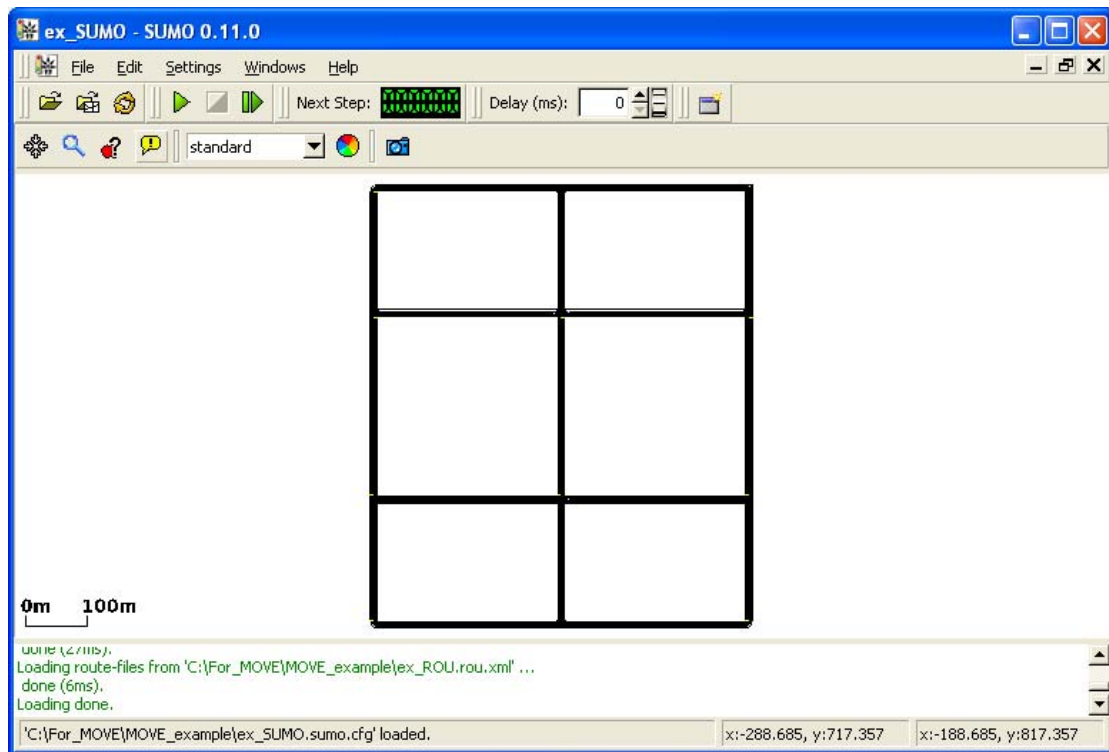
</configuration>

```

### Step 5: Visualize Simulation



Now, you can select “Visualization” see the actual movements on vehicle.



For details on using visualization with SUMO, please refer to the SUMO's manual.



### I.iii Automatically generated map (not implement in the example)

Random Map Generator

☐ Totally Random Layout

☐ Grid Layout

☐ Set number of junctions in both x-y directions 1

☐ Overrides number of junctions in x direction 1

☐ Overrides number of junctions in y direction 1

☐ Set length of roads in both horizontal and vertical directions 100

☐ Overrides length of horizontal roads 100

☐ Overrides length of vertical roads 100

☒ Spider Layout

☒ Set number of axes within the net 3

☒ Set number of circles within the net 3

☐ Set the distances between the circles 100

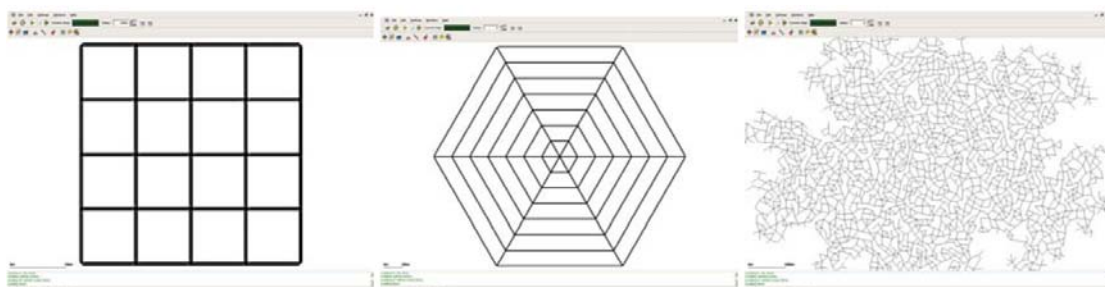
Set Output File ppertestspidermap1.net.xml ...

Computing step 10: Rechecking of lane endings  
Computing step 12: Computing edge shapes  
Computing step 13: Computing node logics  
Computing step 15: Computing traffic light logics  
Success.

OK Cancel

Choose "Random Map" from MOVE main menu to do this.

You can create a custom auto-generated map with three types of layout: grid, spider, or total random.



Select the options, then select your targeted <name>.net.xml file then click OK. A <name>.net.xml will be generated. This is your map file.

## Liv Creating map from a TIGER Line file

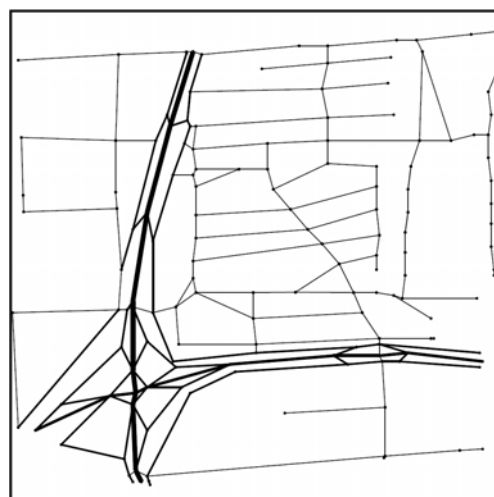
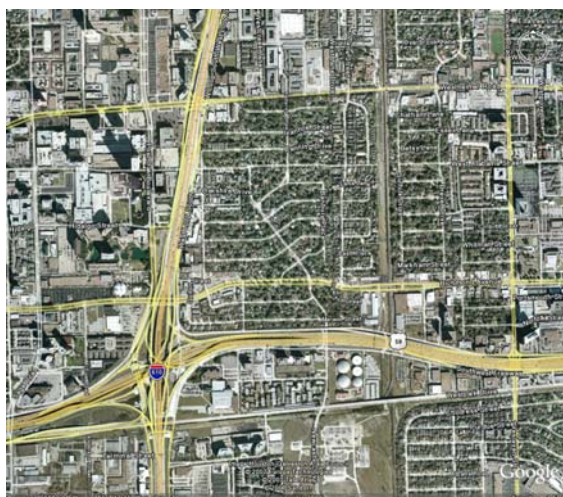


If you have a TIGER Line map file, you can convert this into a MOVE map. Choose "Convert Tiger" from MOVE main menu. Choose the tiger file (usually <name>.dat) Then select the target filename.

**WARNING:** do not put any extension for the filename. Simply put the first name. i.e. /home/tigeroutput (no extension). This will generate <name>.nod.xml, <name>.edg.xml, <name>.netc.cfg, and <name>.net.xml (your map file)

*Note: current version of MOVE will regard all types of roads specified in the TIGER Line file as one type of road (can be seen by opening the <name>.edg.xml file). To support more types of roads, the source file tiger.java*

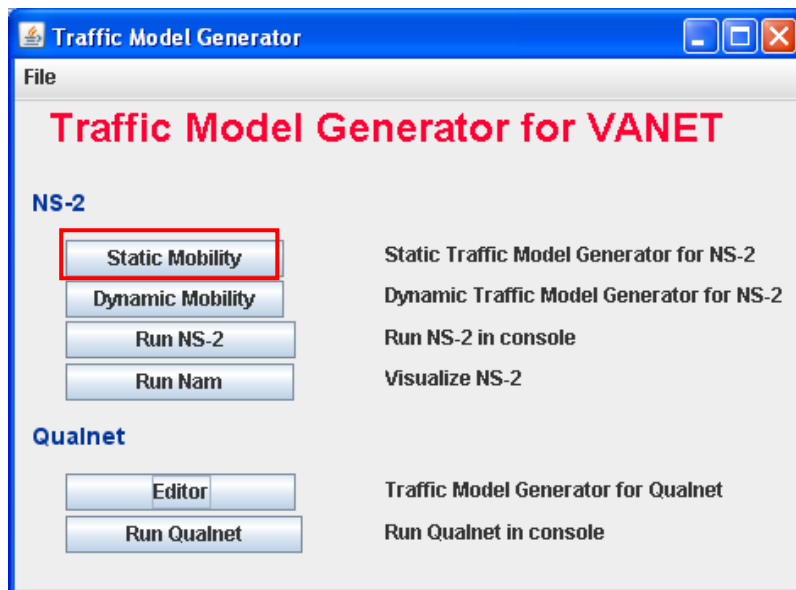
Under the figures show google map and TIGER line map.



## II Traffic Model Generation



First, Select “Traffic Model” from MOVE main menu entering traffic model,



The traffic model generator consists of two main sections: for NS-2 and Qualnet. In this example, we mainly use NS-2 in the example. Right now, MOVE only supports static mobility available. We will try to deploy dynamic mobility in future.

## II.i NS-2 editor (static mobility)

### Step 1: Traffic Model Generator for NS-2 (ex\_NS2.tcl)

The screenshot shows the 'Static Traffic Model Generator for NS-2' window. It has several sections: 'General Options' with fields for Channel Type, Network Interface Type, Interface Queue Type, Antenna Model, Ad hoc Routing Protocol, Radio Propagation Model, MAC Type, Max Packet in IFQ, and Link Layer Type. 'Mobile nodes starting positions' is a table with columns for Time, Node ID, and Initial Position. 'Agent's Options' includes UDP settings (Packet size, Start Time, Stop Time, Sending Rate, Maximum Packets, Introduce Random Noise) and TCP settings (Packet size, Window Size, Start Time, Stop Time, Maximum Burst, Maximum cwnd). 'Connections' is a table with columns for Source ID, Starttime, Destination ID, End time, and Transport Pr. There are buttons for 'Set TCP', 'Add Connection', 'Set UDP', and 'Del Connection'.

This editor will generate the traffic simulation file (a tcl file) for NS-2 simulation tool. First import MOVE Trace (i.e. **ex\_SUMOTRACE.sumo.tr**) and .net.xml (i.e. **ex\_Map.net.xml**) file for script generator.

The screenshot shows the same window with data entered. In the 'General Options' section, 'Channel Type' is 'ChannelWirelessChannel', 'Network Interface Type' is 'PhyWirelessPhy', 'Interface Queue Type' is 'QueueDropTailPriorityQueue', 'Antenna Model' is 'AntennaOmniAntenna', 'Ad hoc Routing Protocol' is 'AODV', 'Radio Propagation Model' is 'PropagationTwoRayGround', 'MAC Type' is 'Mac802\_11', 'Max Packet in IFQ' is '50', and 'Link Layer Type' is 'LL'. In the 'Mobile nodes starting positions' table, there are 10 rows with Node IDs from 0 to 9 and initial positions. In the 'Agent's Options' section, 'UDP' packet size is 1000, sending rate is 64kb, maximum packets is 2280000, and 'Introduce Random Noise' is checked. 'TCP' packet size is 1000, window size is 20, start time is 1, stop time is 1000, maximum burst is 0, and maximum cwnd is 0. In the 'Connections' table, there are two rows: one from flow0\_0 to flow0\_1 at time 10 using udp, and another from flow0\_2 to flow0\_3 at time 100 using udp. The 'Set Nam Trace File' is set to 'ex\_NAM.net' and 'Set Trace Output File' is set to 'ex\_NS2.tr'.

Wait for loading file (it will depend on **file size** and your **computer hardware**). After loading is done, specify the options of the TCL simulation. For more details about the options, please refer to the NS-2 manual.

For the example choose NAM trace file and specify the file location. Then we add the connections between each mobile node specified in the table on the left side and assign the transport protocol (e.g. tcp/udp).

In “mobile nodes starting positions” table, you can see different time, node ID, and

initial position. You can refer this information to add connection in right-side.

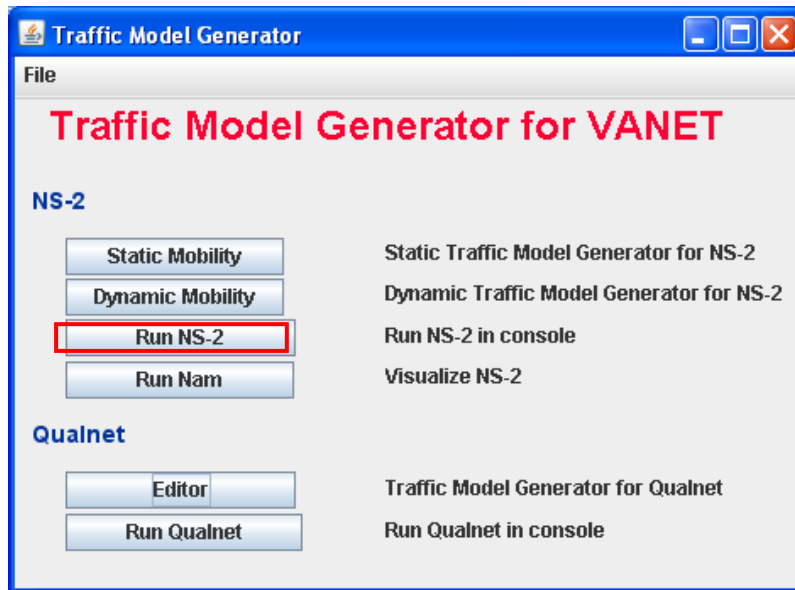
Note, the node ID depend on the name of flow in SUMO. For example, if a flow calls test0, the generated node ID should be called test0\_0, test0\_1, ..., test0\_x, where x is index in the flow.

Please check the node ID. If you input a non-exist node in the table, the TCL file will have not correct node ID.

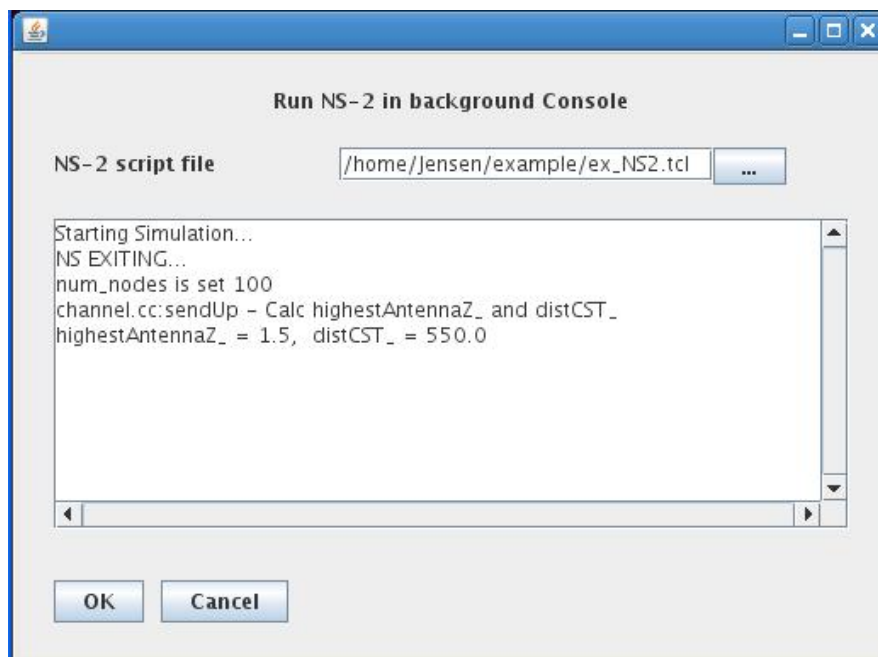
When you are done, select File->Save or Save As and put it as <name>.tcl  
(**ex\_NS2.tcl**)

NOTE: please use ns-2 in linux platform.

**Step 2: Run NS-2 in background console (only Linux platform and without TraCI)**  
(ex\_NAM.nam and ex\_NS2.tr)



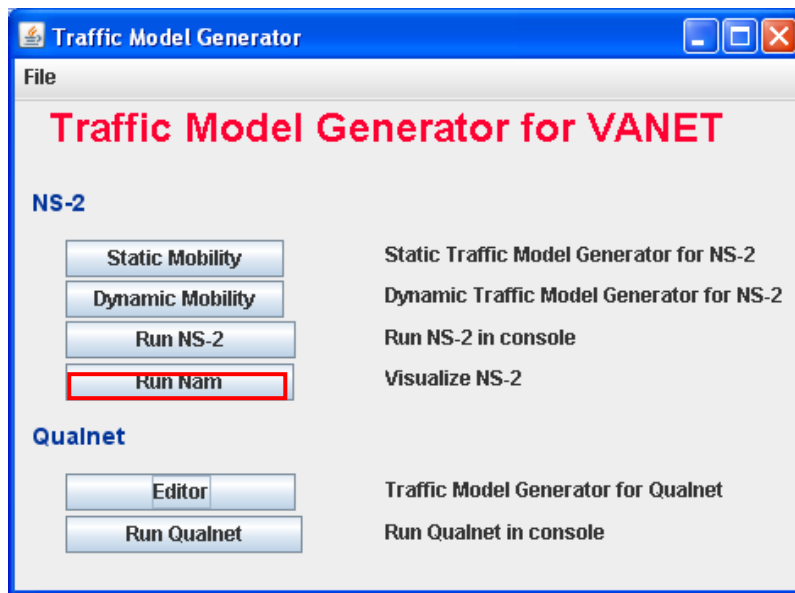
After, select "Run NS-2" runs NS-2 in console.



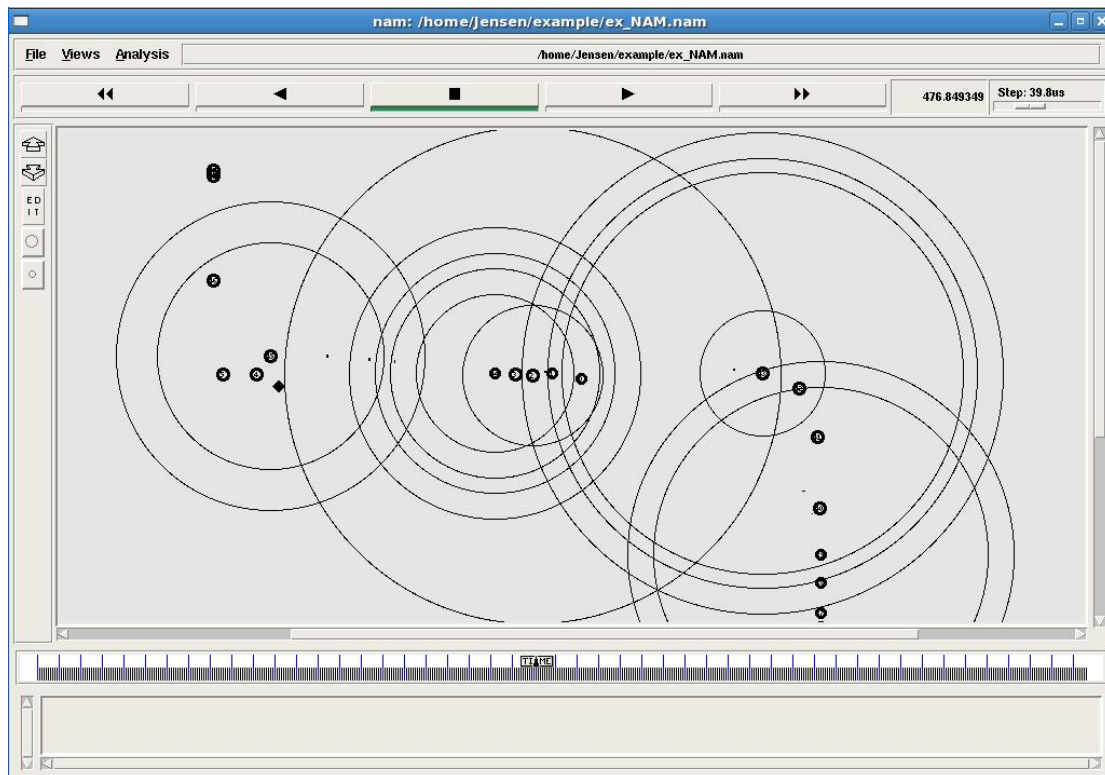
Finally, run the simulation with <name>.tcl (ex\_NS2.tcl)

After that, you can either run the NS-2 script in your own shell or using the program's NS-2 script runner (by choosing the Run NS-2 button from the traffic model main menu).

**Step 3: Run NAM visualize NS-2 in example (only Linux platform)**



Finally, you can call the NAM trace runner from the main menu.



You can play NAM and see the actual movements in example.



## II.ii Qualnet editor

### Step 1: Traffic Model Generator for Qualnet

The Qualnet Simulation Generator (4.0) interface is divided into several sections:

- General Options:** Experiment Name (text), Topology Boundary (x, y coordinates), Simulation Time (text), Number of Nodes (text), Routing Protocol (AODV dropdown), Physical Model (802.11a dropdown), Data rate (6000000 Hz), Temperature (290 K), Noise Factor (7.0), Antenna Model (OMNIDIRECTIONAL), Antenna Gain (0.0 dB), Antenna Height (1.5 m), Antenna Efficiency (0.8 dB), Antenna Mismatch Loss (0.3 dB), Antenna Cable Loss (0.0 dB), Antenna Connection Loss (0.2 dB).
- Wireless Options:** Frequency (2400000000 Hz), Propagation Limit (-111.0 dB), Propagation Pathloss Model (TWO-RAY dropdown), Propagation Shadowing Model (CONSTANT dropdown), Propagation Shadowing Mean (4.0), Propagation Fading Model (RAYLEIGH dropdown), Propagation Gaussian Components File (default.fading), Propagation Fading Max Velocity (10.0), Propagation Ricean K Factor (0.0), MAC802.11 Protocol Options (Short Packet Transmit Limit: 7, Long Packet Transmit Limit: 4, RTS Threshold: 0, Propagation Delay: 1US), IP Network Protocol Options (Fragmentation Unit: 2048, IP Priority Queue Size: 50000, IP Loopback Address: 127.0.0.1).
- Applications Layer Options:** CBR (selected), FTP, TELNET, Item to Send (100), Item Size (512 bytes), Interval (1 second), Start Time (0), End Time (1000 second), Session Duration (1 second), Precedence Bits (0), TCP Settings (Type: LITE, Delay ACKS, Delay Short Packet ACKS, Use Nagle Algorithm, Use Push, Use Keepalive Probes), Send buffer (16384 bytes), Receive buffer (16384 bytes), Maximum Segment Size (512 bytes).
- Mobile Nodes:** Table with ID and Initial Positions.
- Trace and Statistics Settings:** Table with Type and Settings for Application Stats, TCP Stats, Routing Stats, IGMP Stats, Exterior Gateway Pro..., Network Layer Stats, and Queue Stats.

This editor will generate the traffic simulation file (config files) for Qualnet simulation tool. First import MOVE Trace (i.e. **ex\_SUMOTRACE.sumo.tr**) and .net.xml (i.e. **ex\_Map.net.xml**) file for script generator.

The Qualnet Simulation Generator (4.0) interface is divided into several sections:

- General Options:** Experiment Name (text), Topology Boundary (x: 600.0, y: 700.0), Simulation Time (1000 second), Number of Nodes (100), Routing Protocol (AODV dropdown), Physical Model (802.11a dropdown), Data rate (6000000 Hz), Temperature (290 K), Noise Factor (7.0), Antenna Model (OMNIDIRECTIONAL), Antenna Gain (0.0 dB), Antenna Height (1.5 m), Antenna Efficiency (0.8 dB), Antenna Mismatch Loss (0.3 dB), Antenna Cable Loss (0.0 dB), Antenna Connection Loss (0.2 dB).
- Wireless Options:** Frequency (2400000000 Hz), Propagation Limit (-111.0 dB), Propagation Pathloss Model (TWO-RAY dropdown), Propagation Shadowing Model (CONSTANT dropdown), Propagation Shadowing Mean (4.0), Propagation Fading Model (RAYLEIGH dropdown), Propagation Gaussian Components File (default.fading), Propagation Fading Max Velocity (10.0), Propagation Ricean K Factor (0.0), MAC802.11 Protocol Options (Short Packet Transmit Limit: 7, Long Packet Transmit Limit: 4, RTS Threshold: 0, Propagation Delay: 1US), IP Network Protocol Options (Fragmentation Unit: 2048, IP Priority Queue Size: 50000, IP Loopback Address: 127.0.0.1).
- Applications Layer Options:** CBR (selected), FTP, TELNET, Item to Send (100), Item Size (512 bytes), Interval (1 second), Start Time (0), End Time (1000 second), Session Duration (1 second), Precedence Bits (0), TCP Settings (Type: LITE, Delay ACKS, Delay Short Packet ACKS, Use Nagle Algorithm, Use Push, Use Keepalive Probes), Send buffer (16384 bytes), Receive buffer (16384 bytes), Maximum Segment Size (512 bytes).
- Mobile Nodes:** Table with ID and Initial Positions.
- Trace and Statistics Settings:** Table with Type and Settings for Application Stats, TCP Stats, Routing Stats, IGMP Stats, Exterior Gateway Pro..., Network Layer Stats, and Queue Stats.

Wait for loading file (it will depend on **file size** and your **computer hardware**). After loading is done, specify the options of the simulation. For more details about the options, please refer to the Qualnet manual.

In here, we only generate mobility pattern in Qualnet. We also plan to create connection editor in future.



## Appendix I: Advance traffic light control

Reference link: Documentation is available at

[http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Advanced\\_Traffic\\_Lights\\_Simulation](http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Advanced_Traffic_Lights_Simulation)

First take your converted network, open it in a text editor, copy the traffic lights definition (look for ">YOUR\_TLS\_NAME<" and copy the section that is embedded in <tl-logic>), paste it into a new file, embedding it in a further xml-tag, "<additional>".

So you should have a file which looks like:

<additional>

**CUT\_TLS\_DEFINITION**

</additional>

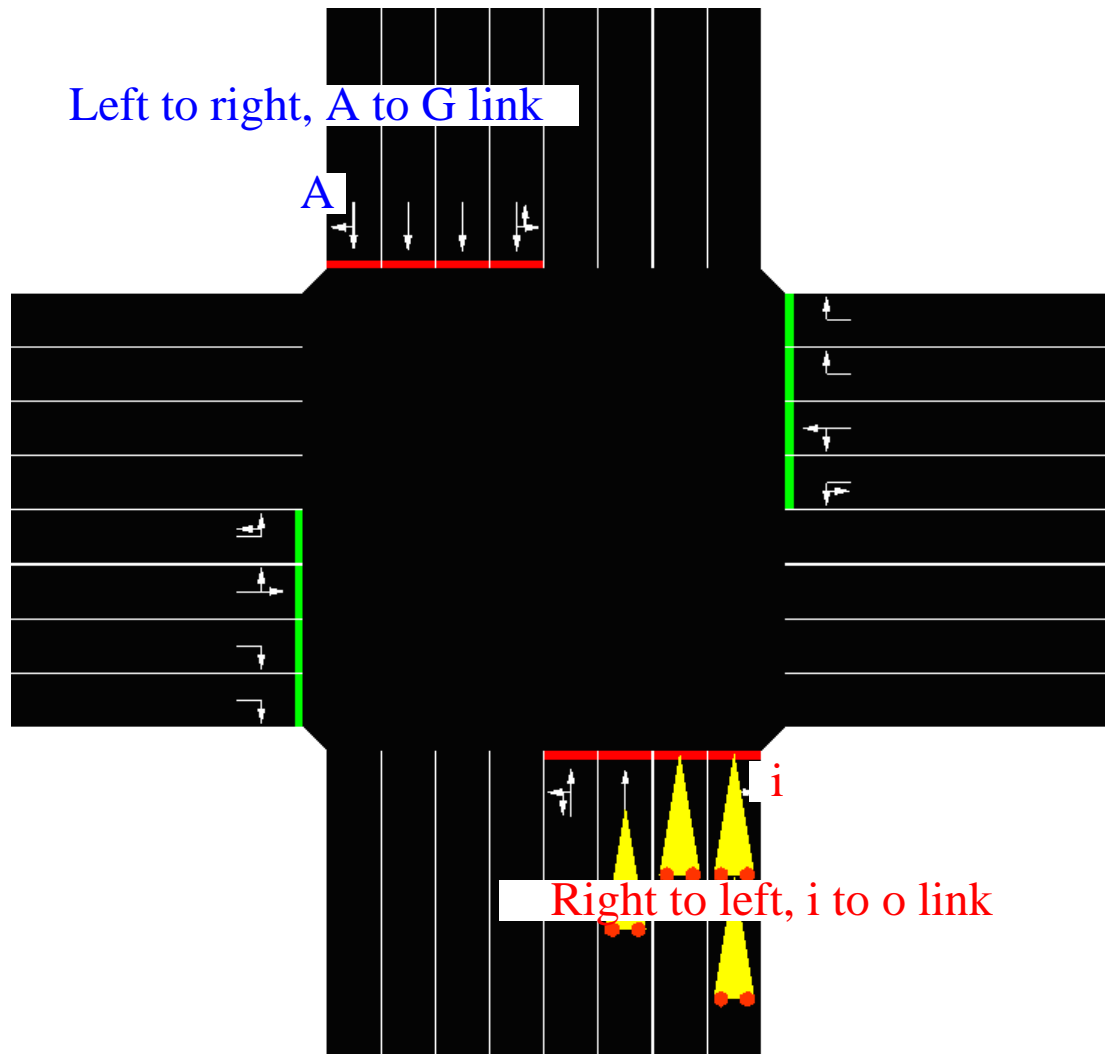
Ok, replace the "programID" by something else. You may now change the durations of all phases. You can load this file by adding its name to the additional-section of your configuration.

For example, we change period of the traffic light in node8. You can get information from map file.

File name: *traffic\_duration.add.xml*

<additional>

```
<tl-logic id="node8" type="static" programID="modify" offset="0">
  <phase duration="35" state="GGGGGggrrrrrrGGGGGggrrrrrr"/> //Green light Index A to
index G highlight with blue color as shown in under Figure. Index I to index O highlight with red color.
  <phase duration="5" state="yyyyygrrrrrryyyyygrrrrrr"/> //Yellow light
  <phase duration="6" state="rrrrGGrrrrrrrrrrGGrrrrrr"/> //Green light for turn left
  <phase duration="5" state="rrrryyrrrrrrrrrryyrrrrrr"/> // Yellow light for turn left
  <phase duration="31" state="rrrrrrGGGGgggrrrrrrGGGGggg"/> // Similar as above
configuration, just different directions.
  <phase duration="5" state="rrrrrryyyyggrrrrrryyyyggg"/>
  <phase duration="6" state="rrrrrrrrGGGrrrrrrrrGGG"/>
  <phase duration="5" state="rrrrrrrryyrrrrrrrryy"/>
</tl-logic>
```



File name: *ex\_SUMO.sumo.cfg*

<configuration>

<input>

<net-file value="/home/Jensen/example/ex\_Map.net.xml"/>

<route-files value="/home/Jensen/example/ex\_Map.net.xml"/>

<additional-files value="/home/Jensen/example/traffic\_duration.add.xml "/>

<junction-files value=""/>

</input>

<output>

<netstate-dump value="/home/Jensen/example/grid.sumo.tr"/>

<tripinfo-output value="output-tripinfos.xml"/>

<emissions-output value="output-emissions.xml"/>

<vehroute-output value="output-vehroutes.xml"/>

</output>

<time>

<begin value="0"/>

<end value="1000"/>

<time-to-teleport value="-1"/>

<srand value="23423"/>

<route-steps value="-1"/>

</time>

<reports>

<print-options value="false"/>

</reports>

</configuration>

PS: This part thanks SUMO developer Daniel to guide.

TAIWAN, NCKU university, CSIE, LENS lab.

Updater: Chine-Ming Chou

## Appendix II: Random trip

Due to SUMO remove random routes generation; introduced a [script for generating random trips](#) instead. The script requires Python.

- Python 2.7 or later - <http://www.python.org/download/>

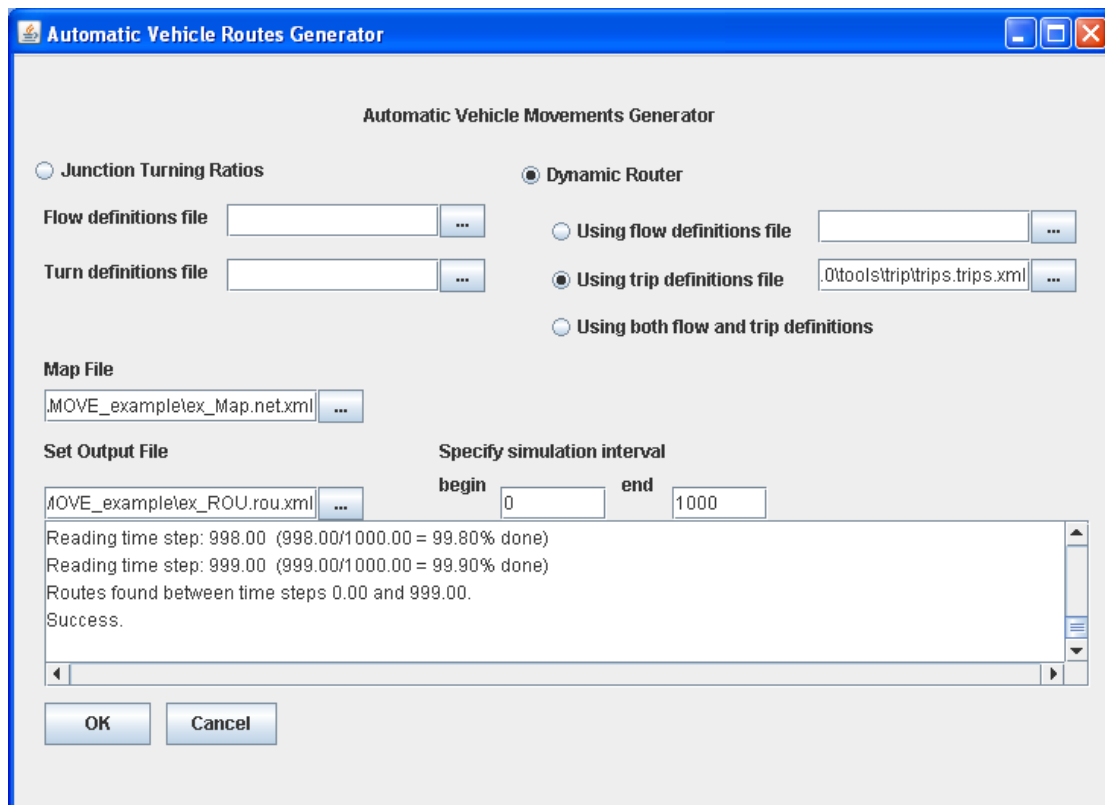
The script is located in: <SUMO>/tools/trip

"**randomTrips.py**" generates a set of random trips for a given network (option "**-n**"). It does so by choosing source and destination edge either uniformly at random or weighted by length (option "**-l**"), by number of lanes (option "**-L**") or both. The resulting trips are stored in an xml file (option "**-o**", default trips.trips.xml) suitable for the [DUAROUTER](#) which is called automatically if the "**-r**" option (with a filename for the resulting route file) is given. The trips are distributed evenly in an interval defined by begin (option "**-b**", default 0) and end time (option "**-e**", default 3600) in seconds. The number of trips is defined by the repetition rate (option "**-p**", default 1) in seconds. Every trip has an id consisting of a prefix (option "**-t**", default "t") and a running number. Example call:

```
randomTrips.py -n input_net.net.xml -e 1000 -l
```

Note that the script does not check whether the chosen destination may be reached from the source. This task is performed by the router.

When you generated trips file, you can feed it into MOVE.



This part refers website of SUMO:

[http://sourceforge.net/apps/mediawiki/sumo/index.php?title=AdditionalTools\\_Trip#randomTrips.py](http://sourceforge.net/apps/mediawiki/sumo/index.php?title=AdditionalTools_Trip#randomTrips.py)

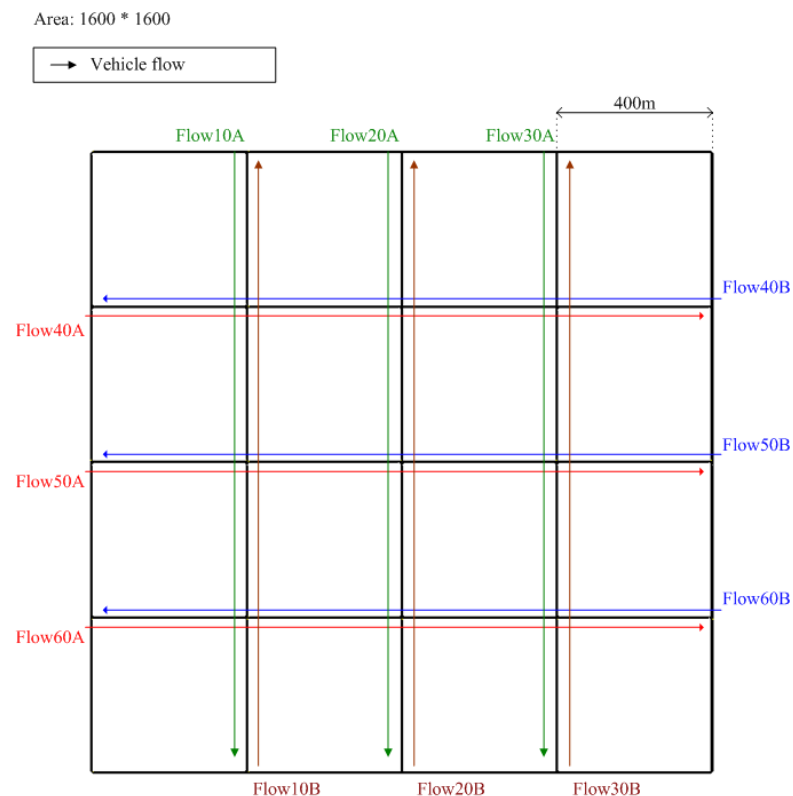
## Appendix III: TraCI using and example (for advanced user)

According to TraCI information, please refer SUMO official website.

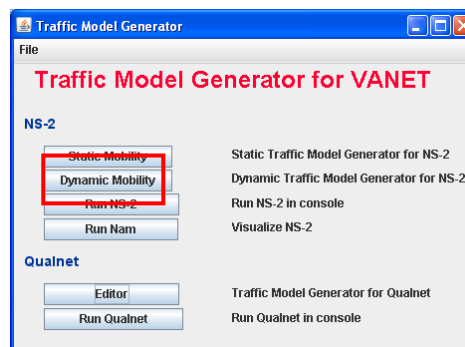
<http://sourceforge.net/apps/mediawiki/sumo/index.php?title=TraCI>

Our work is referred relation SUMO's material to implement.

Here we only provide some examples as us implemented in MOVE for TraCI. Please use our example files (for trace-version) to perform this case study. The scenario is employed a grid map as show in the following figure.



Step A: please select “dynamic mobility”



Note that, this GUI only generates TCL file for TraCI simulation.

TAIWAN, NCKU university, CSIE, LENS lab.

Updater: Chine-Ming Chou

Step B: Import your map file and route file.

**Dynamic Traffic Model Generator for NS-2**

**General Options**

Channel Type: Channel/WirelessChannel  
 Network Interface Type: Phy/WirelessPhy  
 Interface Queue Type: Queue/DropTail/PriQueue  
 Antenna Model: Antenna/OmniAntenna  
 Ad-hoc Routing Protocol: AODV  
 Radio Propagation Model: Propagation/TwoRayGround  
 MAC Type: Mac/802\_11  
 Max Packet in IFQ: 50  
 Link Layer Type: LL

**Mobile Nodes starting positions**

Time	Node ID	Initial Position
0.00	flow60A_0	30.65 410.05 0.0
0.00	flow50A_0	30.65 810.05 0.0
0.00	flow40A_0	30.65 1210.05 ...
0.00	flow10B_0	419.95 30.65 0.0
0.00	flow10A_0	410.05 1599.3...
0.00	flow20B_0	819.95 30.65 0.0
0.00	flow20A_0	810.05 1599.3...
0.00	flow30B_0	1219.95 30.65 ...
0.00	flow60B_0	1599.3500000...
0.00	flow50B_0	1599.3500000...

**Agents Options**

UDP  
 Packet size: 1000  
 Start Time: 0.00  
 Stop Time: 290.00  
 Sending Rate: 64kb  
 Maximum Packets: 280000  
☐ Introduce Random Noise

TCP  
 Packet size: 1000  
 Window Size: 20  
 Start Time: 0.00  
 Stop Time: 290.00  
 Maximum Burst: 0  
 Maximum cwnd: 0

**Connections**

Source ID	Start time	Destination...	End time	Transport ...
-----------	------------	----------------	----------	---------------

Buttons: Set TCP, Set UDP, Add Connection, Del Connection

In this example, we save this file to “ex\_traci.tcl”.

Before you run simulation, you should install ns2 trace-patch.

# directly copy all files into your ns-2 folder (e.g. tcpip, traci-client, and Makefile.in).

# in ns-2 folder input “./configure”

```

jensen@jensen-desktop: ~/tools/ns-allinone-2.34/ns-2.34
File Edit View Terminal Help
AllinOne    diffserv    mcast      sensor-nets
aodv        diffusion   mobile     src_rtg
aomdv       diffusion3  mpls       tcl
apps        doc         nam        tcp
asim        dsdv        nix        tcpip
autoconf.h  dsr         ns         test-all
autoconf.h.in empweb      ns.1       test-output
autoconf-win32.h emulate     nse        tmix
BASE-VERSION FILES       ns_tclsh.cc TODO.html
baytcp      gaf         nstl       tools
bin         gen         packmime   tora
bitmap      HOWTO-CONTRIBUTE pgm        trace
CHANGES.html html        plm        traci-client
classifier  imcp       pushback   update.out
common      indep-utils qs          validate
conf        install-sh  queue      validate.out
config.guess INSTALL.WIN32 rap         VERSION
config.h    lib        README     webcache
config.log  LICENSES   realaudio  wpan
config.status link       release_steps.txt xcp
config.sub  linkstate routealgo
configure   mac       routing
configure.in Makefile  rtproto
jensen@jensen-desktop:~/tools/ns-allinone-2.34/ns-2.34$ ./configure

```

# and then “make”

TAIWAN, NCKU university, CSIE, LENS lab.

Updater: Chine-Ming Chou

In this example, we will show some TraCI API as follows:

1. command-GetRoadID [nsID] [sumoID]

# Returns the id of the edge the named vehicle was at within the last step; error value:  
""

# \$ns\_ at 3.0 "\$mobilityInterfaceClient command-GetRoadID 6 flow20A\_0"

2. command-SetMaxSpeed [nsID] [sumoID] [max speed]

# Sets the vehicle's maximum speed to the given value

# \$ns\_ at 5.0 "\$mobilityInterfaceClient command-SetMaxSpeed 0 flow60A\_0 5"

3. command- changeTarget [nsID] [sumoID] [target edge]

# The vehicle's destination edge is set to the given. The route is rebuilt.

# \$ns\_ at 5.0 "\$mobilityInterfaceClient command-ChangeTarget 0 flow60A\_0 4344-2"

4. command-changeRoute [nsID] [sumoID] [number of edges] [edges lists]

# Assigns the list of edges as the vehicle's new route assuming the first edge given is the one the vehicle is currently at: The first occurrence of the edge is currently at is searched within the new route; the vehicle continues the route from this point in the route from. If the edge the vehicle is currently does not exist within the new route, an error is generated.

# \$ns\_ at 5.0 "\$mobilityInterfaceClient command-changeRoute 0 flow60A\_0 7 0111-1,1011-2,1020-1,2021-1,2131-1,3141-1,3141-2"

### Step C: Execution TraCI server (Important)

Before you run ns-2 simulation, you need to execute TraCI server side every time.

The command format is

# Sumo -n [your map file] -r [your route file] --remote-port 8888 --penetration 1 --route-steps -1

For our example, you should input

# Sumo -n /yourPath/TraCI\_work\_grid\_4\_4/grid.net.xml -r /yourPath/TraCI\_work\_grid\_4\_4/grid.rou.xml --remote-port 8888 --penetration 1 --route-steps -1

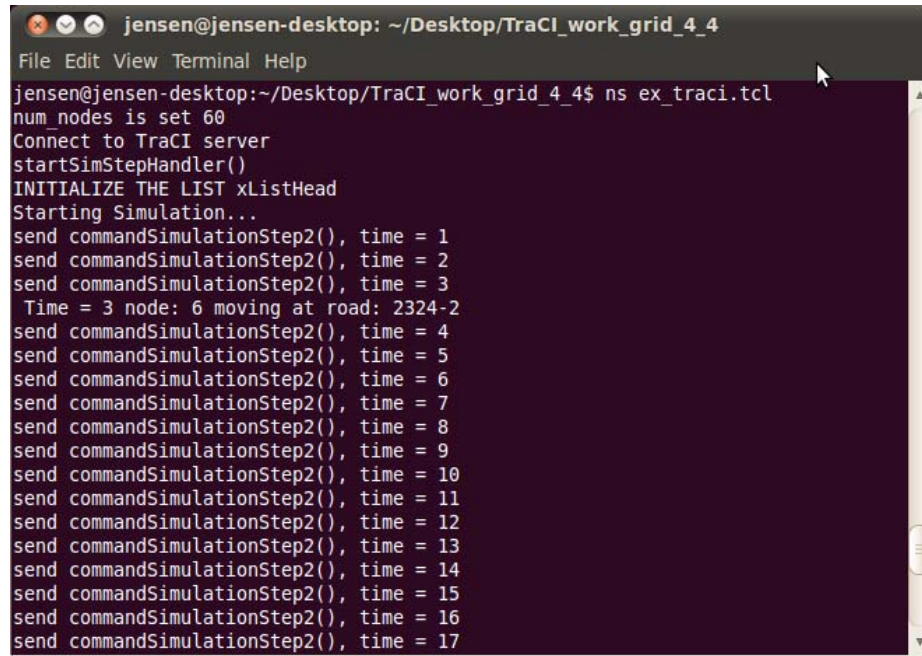


#### Step D: Test command-GetRoadID

Please remove comment (i.e. #) in “ex\_traci.tcl”.

\$ns\_ at 3.0 "\$mobilityInterfaceClient command-GetRoadID 6 flow20A\_0"

You will get road ID “2324-2” as show in the following figure.



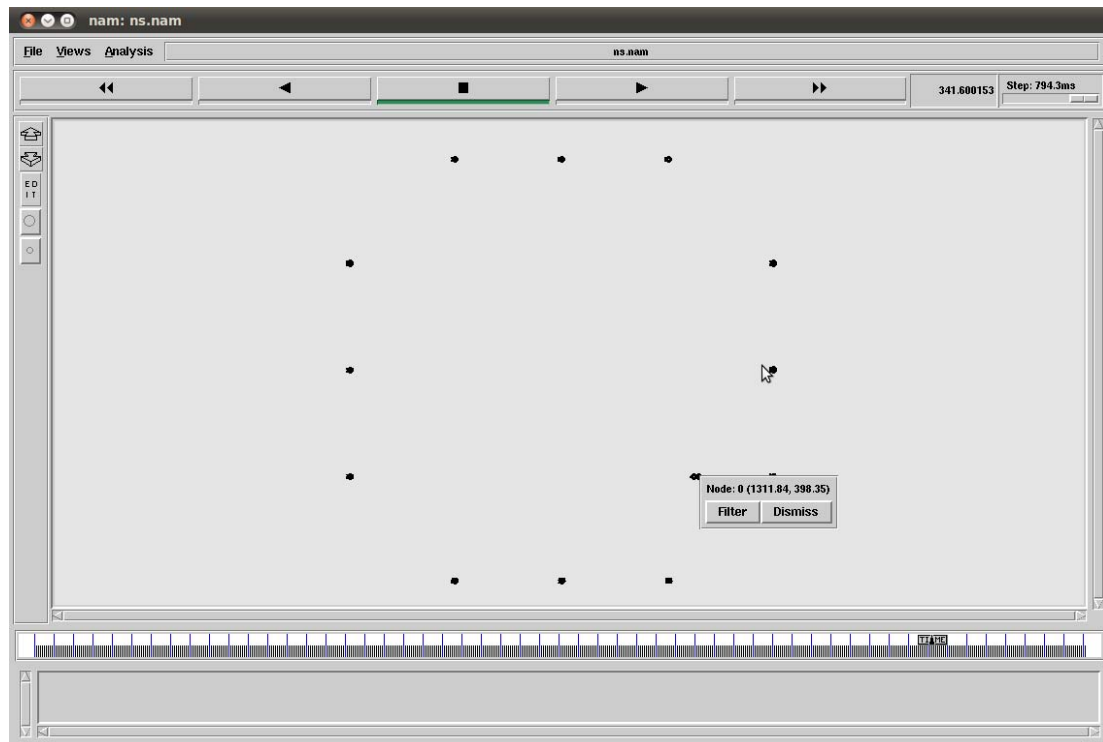
```
jensen@jensen-desktop: ~/Desktop/TraCI_work_grid_4_4
File Edit View Terminal Help
jensen@jensen-desktop:~/Desktop/TraCI_work_grid_4_4$ ns ex_traci.tcl
num_nodes is set 60
Connect to TraCI server
startSimStepHandler()
INITIALIZE THE LIST xListHead
Starting Simulation...
send commandSimulationStep2(), time = 1
send commandSimulationStep2(), time = 2
send commandSimulationStep2(), time = 3
Time = 3 node: 6 moving at road: 2324-2
send commandSimulationStep2(), time = 4
send commandSimulationStep2(), time = 5
send commandSimulationStep2(), time = 6
send commandSimulationStep2(), time = 7
send commandSimulationStep2(), time = 8
send commandSimulationStep2(), time = 9
send commandSimulationStep2(), time = 10
send commandSimulationStep2(), time = 11
send commandSimulationStep2(), time = 12
send commandSimulationStep2(), time = 13
send commandSimulationStep2(), time = 14
send commandSimulationStep2(), time = 15
send commandSimulationStep2(), time = 16
send commandSimulationStep2(), time = 17
```

### Step E: Test command-SetMaxSpeed

Please remove comment (i.e. #) in “ex\_traci.tcl”.

```
# $ns_ at 5.0 "$mobilityInterfaceClient command-SetMaxSpeed 0 flow60A_0 5"
```

Once you change the vehicle speed, which node will by specific speed until you change it again. The results show in the following figure. Node 0 moves slow in this simulation due to maximum speed is 5 meter/sec.

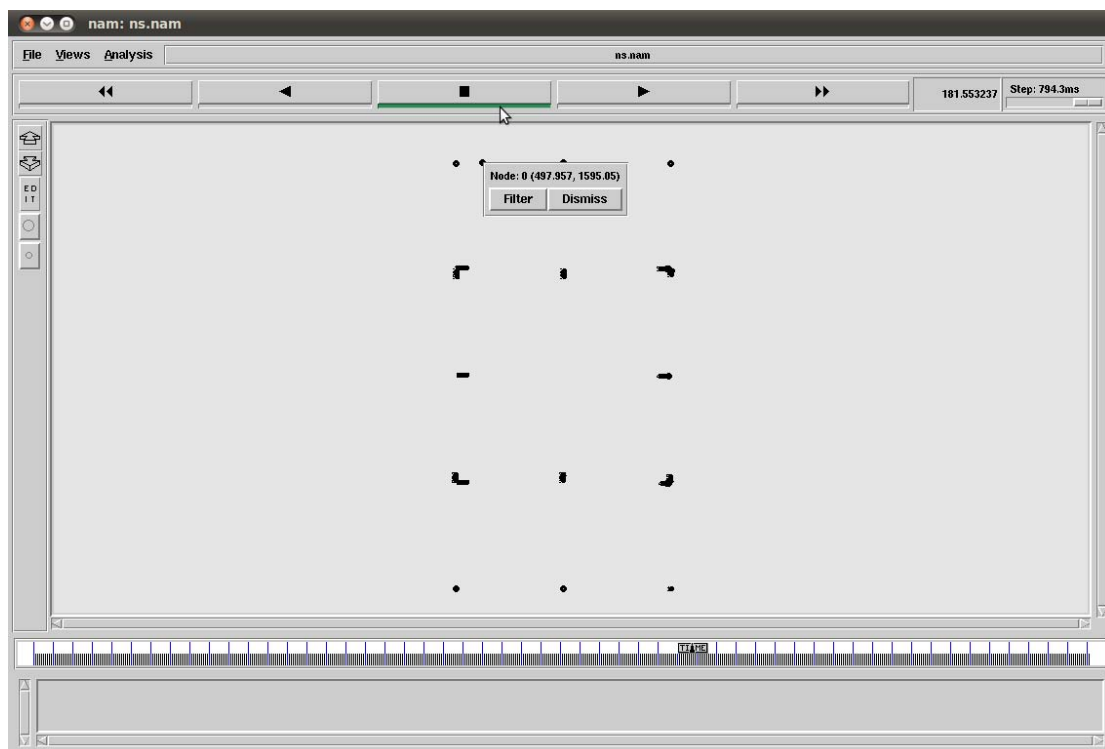


### Step F: Test command-ChangeTarget

Please remove comment (i.e. #) in “ex\_traci.tcl”.

```
# $ns_ at 5.0 "$mobilityInterfaceClient command-ChangeTarget 0 flow60A_0 4344-2"
```

When you perform this command, node will change route by the shortest path (auto calculated by SUMO). In this example, node 0 (i.e. flow60A\_0 will move different path to target edge “4344-2”.



### Step G: Test command-changeRoute

Please remove comment (i.e. #) in “ex\_traci.tcl”.

```
# $ns_ at 5.0 "$mobilityInterfaceClient command-changeRoute 0 flow60A_0 7  
0111-1,1011-2,1020-1,2021-1,2131-1,3141-1,3141-2"
```

For this command, you need to input the vehicle “current edge” for first edge (i.e. 0111-1), and then input each edge from current edge to destination edge. For example, 0111-1,1011-2,1020-1,2021-1,2131-1,3141-1,3141-2, here 0111-1 is current edge and 3141-2 is destination edge.

The results show in the following. Node 0 is moving to the other way (highlight with red circle)

