SUMO - More on...
Dynamic User Assignment

Daniel Krajzewicz

Danilot Tete Boyom

SUMO - More on... Dynamic User Assignment

Daniel Krajzewicz
Danilot Tete Boyom

$Revision: 3772 $

Table of Contents

1. Introduction 1
Document Information 1
Document Scope 1
Document Audience 1
Document Structure 1
2. DUA on the City of Magdeburg from the INVENT Project 2
Prerequisites 2
Evaluations of the DUA 3
Development of Vehicle Emissions 3
3. Tools 12

List of Figures

2.1. The network of City of Magdeburg used within INVENT as shown by SUMO 2
2.2. Development of running vehicles over time and dua iteration step (iteration step number is increased by 5) (tbd) 3
2.3. Development of running vehicles over time in steps 30-39 (tbd) 4
2.4. Development of running vehicles over time in steps 30-39, step 34 and 35 excluded (tbd) 5
2.5. Development of ended vehicles over time and DUA iteration step (iteration step number is increased by 5) (tbd) 6
2.6. Development of waiting vehicles over time and dua iteration step (iteration step number is increased by 5) (tbd) 7
2.7. Development of the mean waiting time over time and dua iteration step (iteration step number is increased by 5) (tbd) 8
2.8. Development of the mean travel time over time and dua iteration step (iteration step number is increased by 5) (tbd) 9
2.9. Development of the mean travel time over time and dua iteration step for the last iteration steps 10
2.10. Development of the mean travel time over time and dua iteration step for the last and valid iteration steps 11

Chapter 1. Introduction

Document Information

This document is a part of *SUMO* - *S*imulation of *U*rban *MO*bility. SUMO is an open source microscopic road traffic simulation. It can be found under http://sumo.sourceforge.net/.

This document fits to version 0.9.5. We will try to keep it up to date with the next versions.

Document Scope

This chapter is meant to be an add-on to the user documentation [http://sumo.sourceforge.net/docs/gen/user_chp06.shtml]. We will not discuss how to perform the DUA using SUMO, herein, this is described within chapter "Dynamic User Assignment [http://sumo.sourceforge.net/docs/gen/user_chp05.shtml#user_chp05-dua]" of the user documentation [http://sumo.sourceforge.net/docs/userdocs.shtml]. What we would like to do instead is to talk some deeper aspects using a examples.

Document Audience

This document contains information for *users* of SUMO.

Document Structure

The first chapter discusses a real-world DUA example on the City of Magdeburg using data from the INVENT project.

Chapter 2. DUA on the City of Magdeburg from the INVENT Project

Prerequisites

The next picture shows one of the networks we have used within the project "INVENT". It was given to us as an VISUM network. As VISUM-routes normally start at explicite feeding edges, we could reduce the network by removing geometry nodes. The routes were given as OD-matrices in a ptv-Format. They define the flows for the time between 5am and 12pm. After converting the matrices using OD2TRIPS into trips we were able to begin with the DUA. The assignment was made using the `iterate.pl` - script described here (tbd).

Figure 2.1. The network of City of Magdeburg used within INVENT as shown by SUMO

Evaluations of the DUA

The DUA has been run 40 times in order to check the behaviour of the algorithm. As output, we generated the vehicle emissions (see chapter tbd), vehicle trip duration (see chapter tbd) and edge-based dumps (see chapter TBD), the latest with aggregation times of 150s, 300s and 900s.

We will now examine the data generated, starting with the vehicle emissions file.

Development of Vehicle Emissions

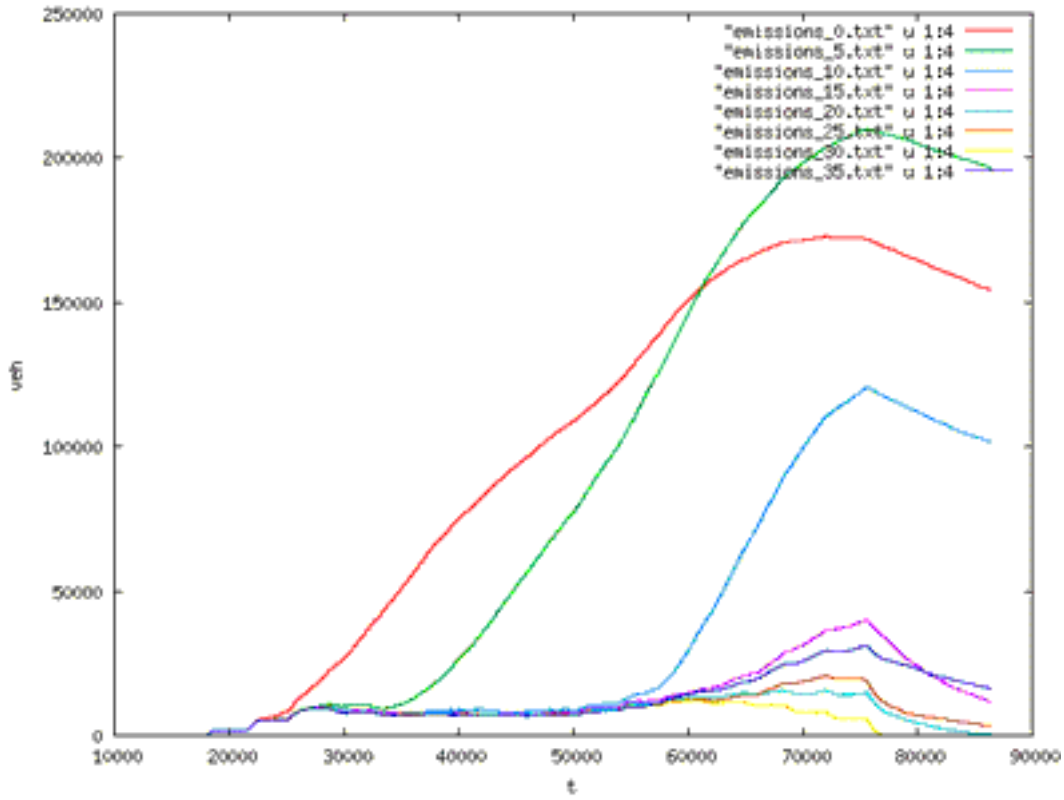
The vehicle emissions file (see chapter tbd) gives information about how many vehicles were loaded/runnning and have started/ended for each time stemp. Additionally, the time the simulation needed to compute the step is written.

Running Vehicles

The next picture shows the development of the number of vehicles within the simulation over the dua iteration process. Every 5th iteration is shown. The plot was done using the following gnuplot command after converting the xml-output into files readable by gnuplot:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "veh"
gnuplot> plot "emissions_0.txt" u 1:4 w l, "emissions_5.txt" u 1:4 w l, \
   "emissions_10.txt" u 1:4 w l, "emissions_15.txt" u 1:4 w l, \
   "emissions_20.txt" u 1:4 w l, "emissions_25.txt" u 1:4 w l, \
   "emissions_30.txt" u 1:4 w l, "emissions_35.txt" u 1:4 w l
```

Figure 2.2. Development of running vehicles over time and dua iteration step (iteration step number is increased by 5) (tbd)



Lets review what we see:

1. The number of running vehicles decreases with each step

   What we see here in fact is exactly the reason for using a dynamic user assignment. The large number of vehicles within the first iterations indicates that new vehicles are inserted into the network while the previously inserted have not yet left. The reason why previously inserted vehicles are still in the net is that they got stocked in jams. And the jams occured because all vehicles are trying to use the same route.

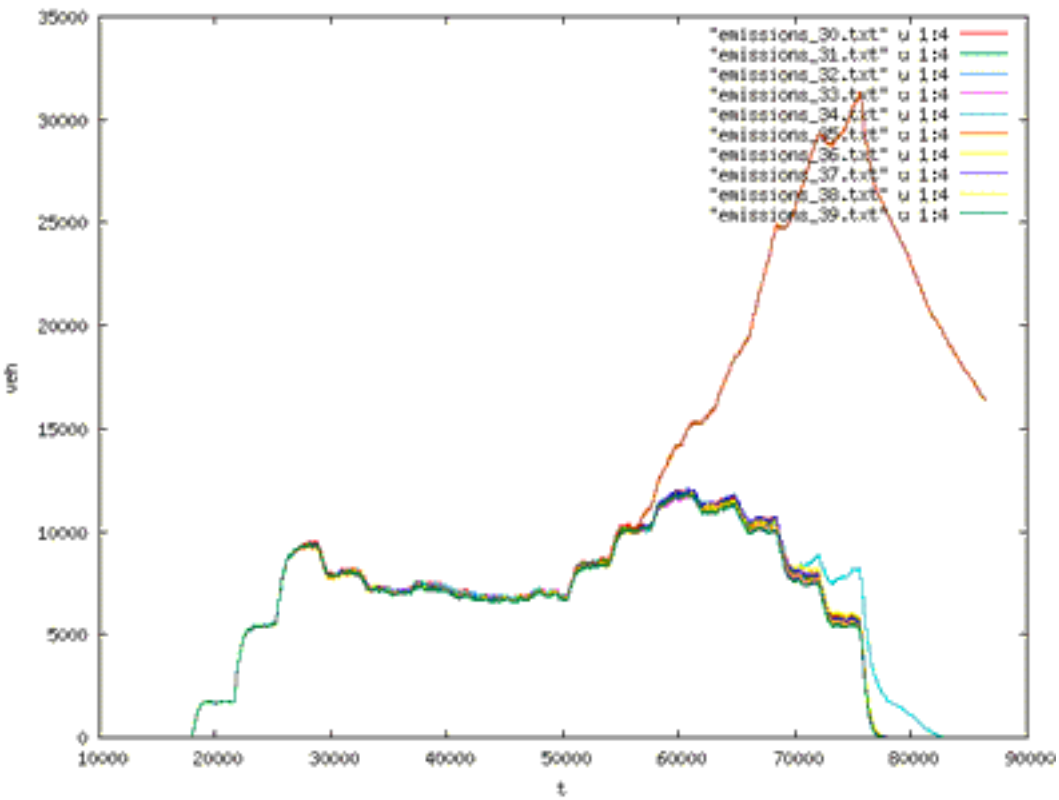
2. Not every following iteration step is better than the previous

   It may be supposes that this may only be due to the dynamic changes within the simulation. That means, that both small changes within the routes' distribution and the stochasticity in the microscopic model may yield in a larger jam which breaks down the network's capacity. Nevertheless, it is also possible that the microscopic model is not flexible enough not to cause the network to be free of jams and grid-locks as the reality is.

Now let's recheck the last steps (step 30-39) using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "veh"
gnuplot> plot "emissions_30.txt" u 1:4 w l, "emissions_31.txt" u 1:4 w l, \
   "emissions_32.txt" u 1:4 w l, "emissions_33.txt" u 1:4 w l, \
   "emissions_34.txt" u 1:4 w l, "emissions_35.txt" u 1:4 w l, \
   "emissions_36.txt" u 1:4 w l, "emissions_37.txt" u 1:4 w l, \
   "emissions_38.txt" u 1:4 w l, "emissions_39.txt" u 1:4 w l
```

Figure 2.3. Development of running vehicles over time in steps 30-39 (tbd)

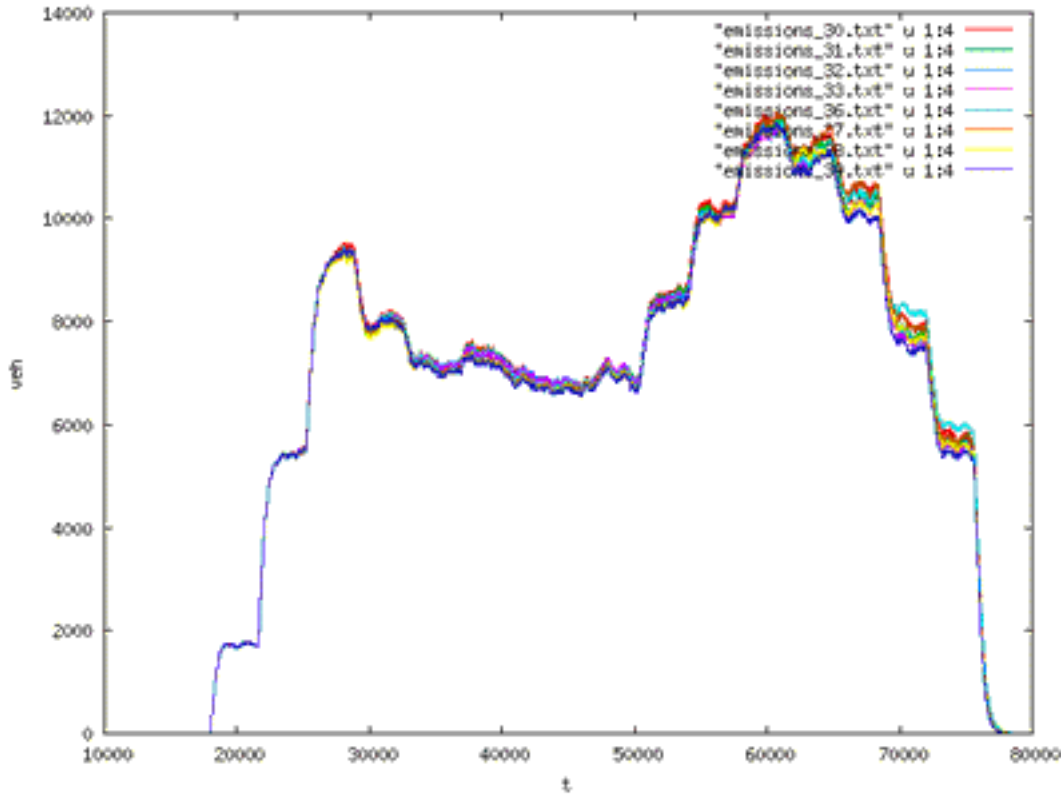


If we remove those "broken" assignments, we can see that the development is almost closed at step 30 and only a small advantage can is done:

```
gnuplot> set xlabel "t"
```

```
gnuplot> set ylabel "veh"
gnuplot> plot "emissions_30.txt" u 1:4 w l, "emissions_31.txt" u 1:4 w l, \
   "emissions_32.txt" u 1:4 w l, "emissions_33.txt" u 1:4 w l, \
   "emissions_36.txt" u 1:4 w l, "emissions_37.txt" u 1:4 w l, \
   "emissions_38.txt" u 1:4 w l, "emissions_39.txt" u 1:4 w l
```

Figure 2.4. Development of running vehicles over time in steps 30-39, step 34 and 35 excluded (tbd)



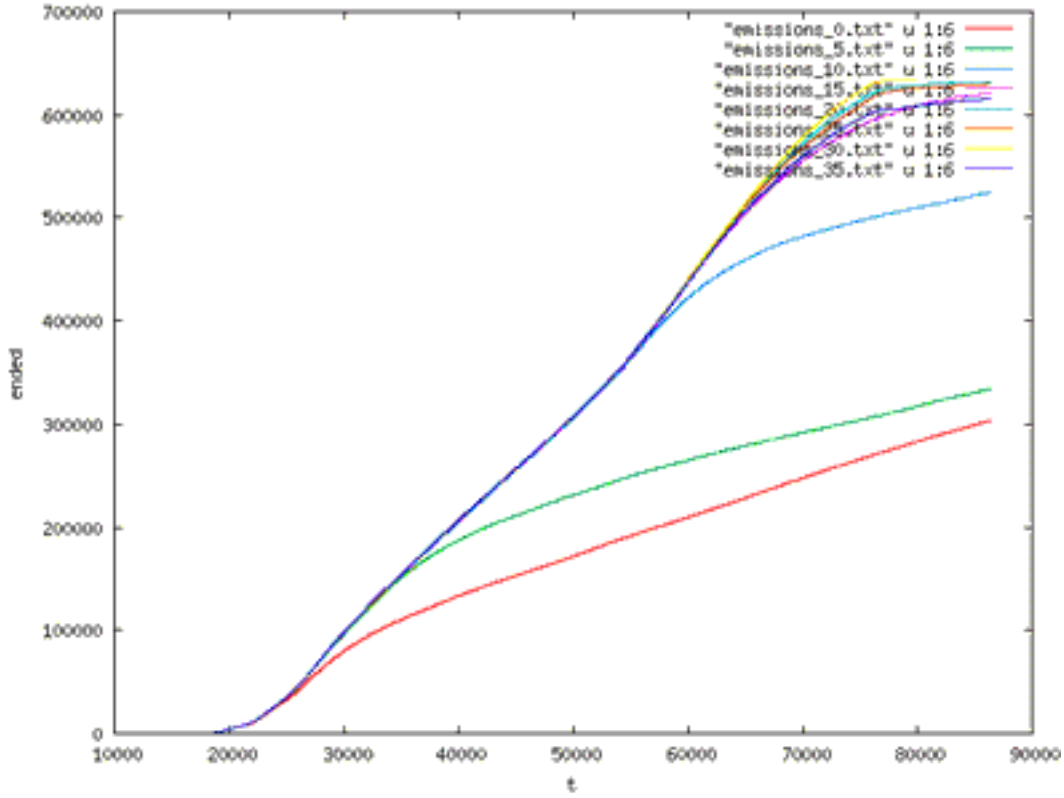
It is not clear whether it is necessary to try the really optimal approach. One can assume that in reality, some drivers are still passing major roads they know without asking about a shorter possibility. As the next picture shows, the dua of step !!! is fitting very well to the data we got from detectors within Magdeburg.

One can state, that the dua is finished if the vehicles running in a simulation step is not changing over some iteration step dramatically. A real function or algorithm for evaluating this but is still missing.

Ended Vehicles

Almost the same information can be gained using the information about the number of vehicles that have left the simulation stored within the emissions file. As one may see within the next picture, the number of vehicles which have left the simulation is increasing with the iteration step. This of course has the same reasons as described within the previous chapter: the vehicles get stocked in the simulation due to jams.

Figure 2.5. Development of ended vehicles over time and DUA iteration step (iteration step number is increased by 5) (tbd)



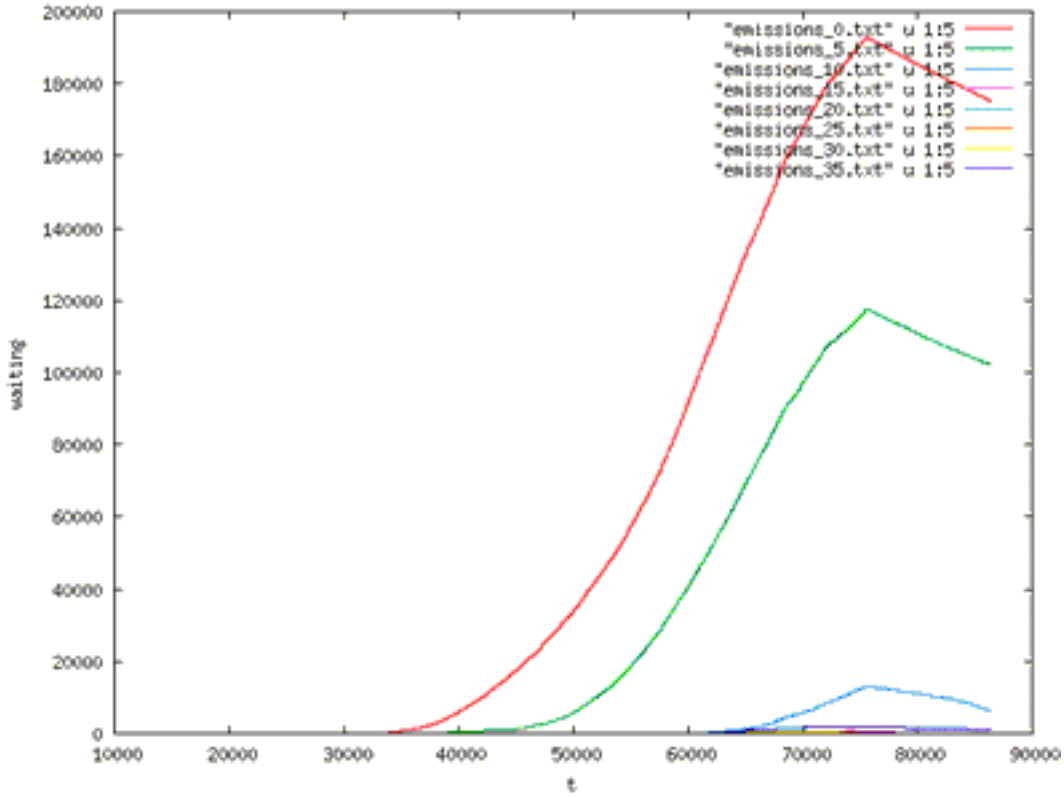
This picture was done using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "ended"
gnuplot> plot "emissions_0.txt" u 1:6 w l, "emissions_5.txt" u 1:6 w l, \
   "emissions_10.txt" u 1:6 w l, "emissions_15.txt" u 1:6 w l, \
   "emissions_20.txt" u 1:6 w l, "emissions_25.txt" u 1:6 w l, \
   "emissions_30.txt" u 1:6 w l, "emissions_35.txt" u 1:6 w l
```

Waiting Vehicles and the Mean Waiting Time

That vehicles are stocked may be also visualised using the number of waiting vehicles. Vehicles are declared as waiting if they can not be emitted into the network, because their departure edge is full. The next picture shows the development of this value during the simulations and the dua-steps.

Figure 2.6. Development of waiting vehicles over time and dua iteration step (iteration step number is increased by 5) (tbd)

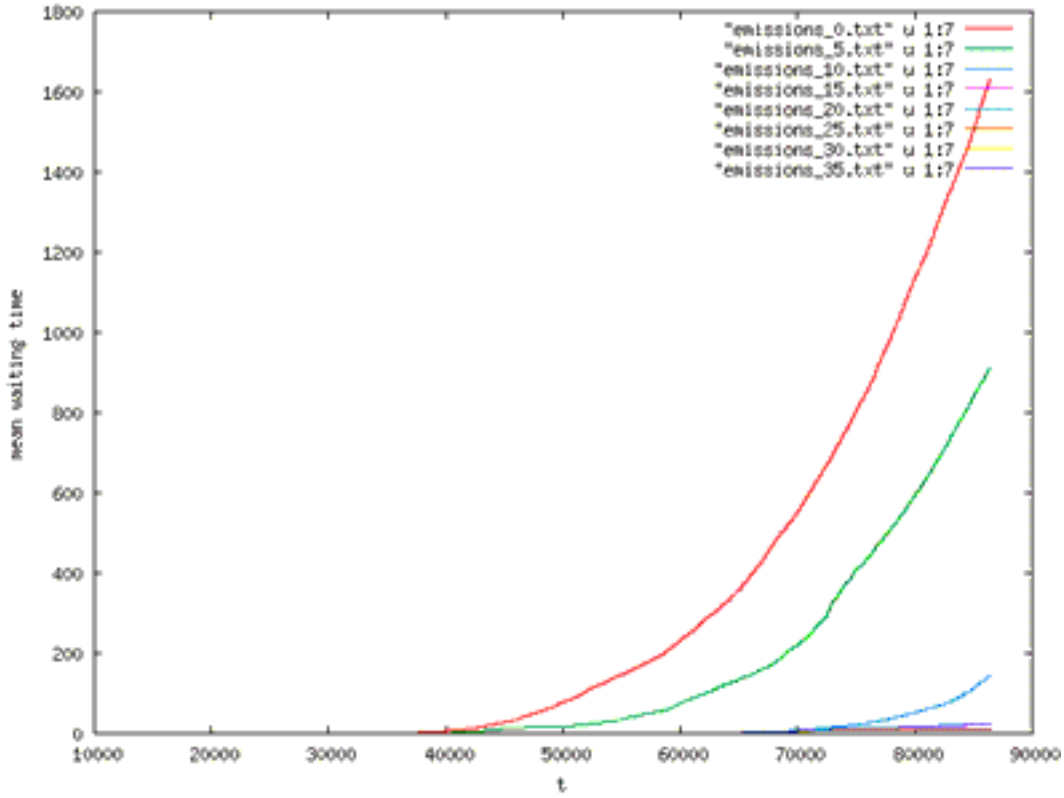


This picture was done using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "waiting"
gnuplot> plot "emissions_0.txt" u 1:5 w l, "emissions_5.txt" u 1:5 w l, \
   "emissions_10.txt" u 1:5 w l, "emissions_15.txt" u 1:5 w l, \
   "emissions_20.txt" u 1:5 w l, "emissions_25.txt" u 1:5 w l, \
   "emissions_30.txt" u 1:5 w l, "emissions_35.txt" u 1:5 w l
```

This is also visible when evaluating the mean waiting time:

Figure 2.7. Development of the mean waiting time over time and dua iteration step (iteration step number is increased by 5) (tbd)



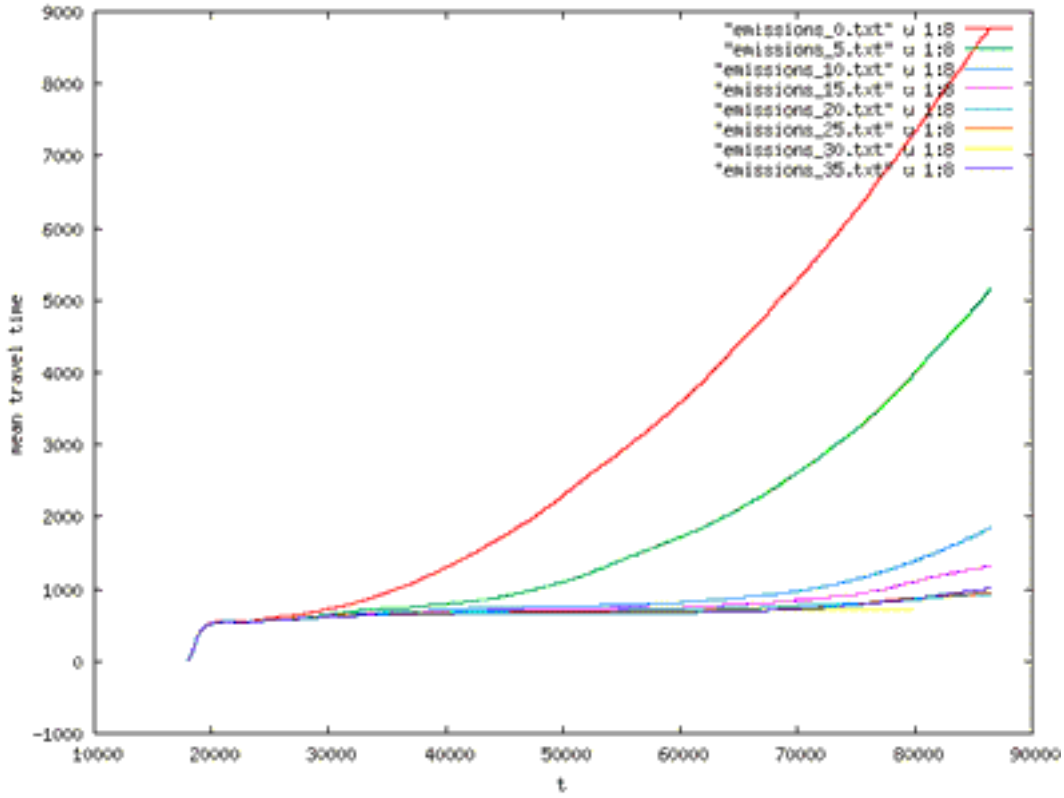
This picture was done using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "mean waiting time"
gnuplot> plot "emissions_0.txt" u 1:7 w l, "emissions_5.txt" u 1:7 w l, \
   "emissions_10.txt" u 1:7 w l, "emissions_15.txt" u 1:7 w l, \
   "emissions_20.txt" u 1:7 w l, "emissions_25.txt" u 1:7 w l, \
   "emissions_30.txt" u 1:7 w l, "emissions_35.txt" u 1:7 w l
```

Mean Travel Time

The next picture shows how dramatically the dua changes the mean travel time.

Figure 2.8. Development of the mean travel time over time and dua iteration step (iteration step number is increased by 5) (tbd)

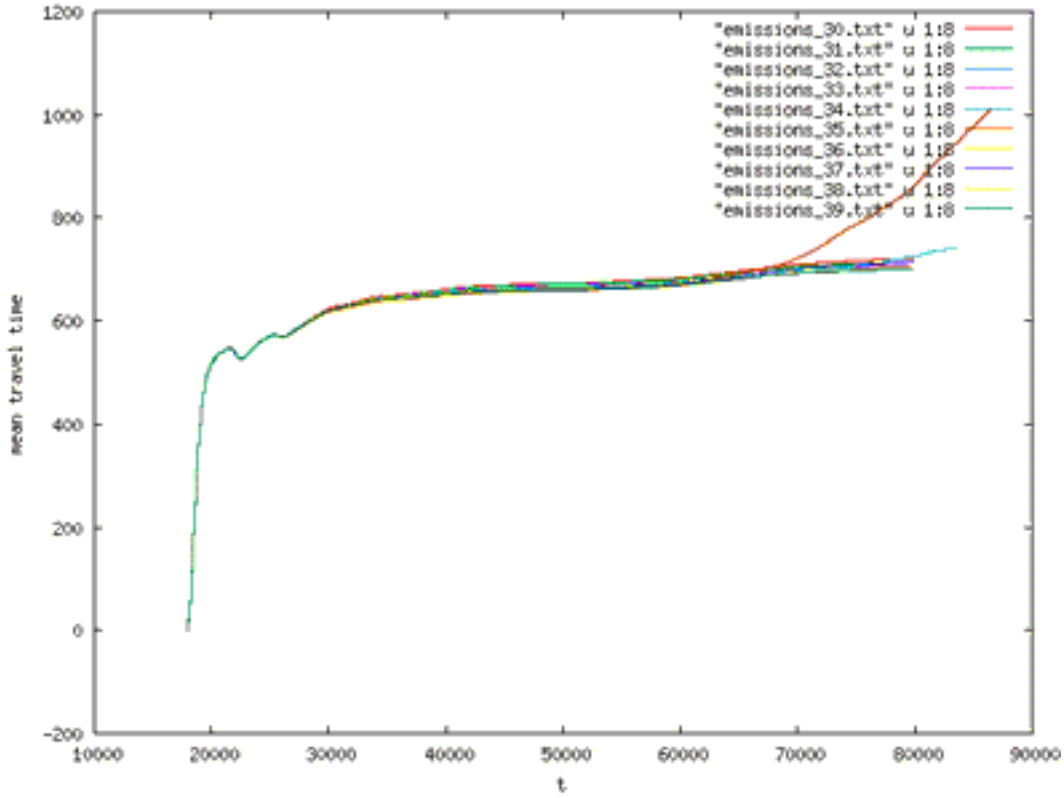


This picture was done using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "mean travel time"
gnuplot> plot "emissions_0.txt" u 1:8 w l, "emissions_5.txt" u 1:8 w l, \
   "emissions_10.txt" u 1:8 w l, "emissions_15.txt" u 1:8 w l, \
   "emissions_20.txt" u 1:8 w l, "emissions_25.txt" u 1:8 w l, \
   "emissions_30.txt" u 1:8 w l, "emissions_35.txt" u 1:8 w l
```

... and for the last 10 steps:

Figure 2.9. Development of the mean travel time over time and dua iteration step for the last iteration steps

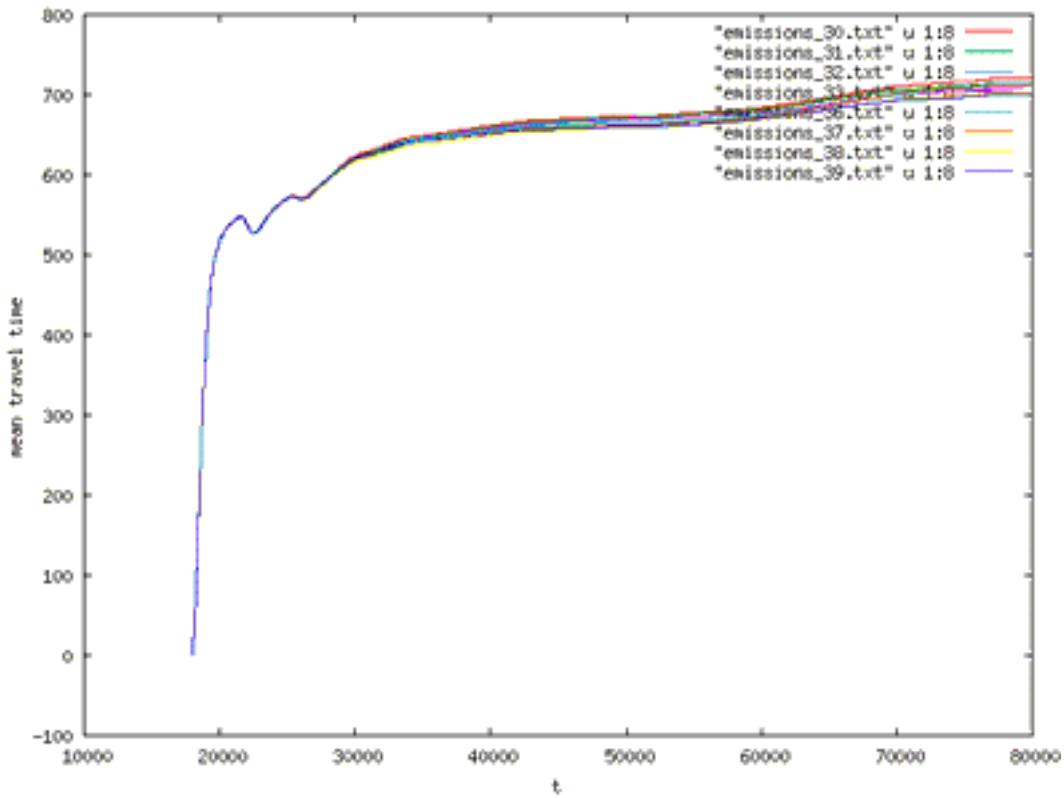


This picture was done using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "mean travel time"
gnuplot> plot "emissions_30.txt" u 1:8 w l, "emissions_31.txt" u 1:8 w l, \
   "emissions_32.txt" u 1:8 w l, "emissions_33.txt" u 1:8 w l, \
   "emissions_34.txt" u 1:8 w l, "emissions_35.txt" u 1:8 w l, \
   "emissions_36.txt" u 1:8 w l, "emissions_37.txt" u 1:8 w l, \
   "emissions_38.txt" u 1:8 w l, "emissions_39.txt" u 1:8 w l
```

... and for those last steps where no vehicle blocked:

Figure 2.10. Development of the mean travel time over time and dua iteration step for the last and valid iteration steps



using:

```
gnuplot> set xlabel "t"
gnuplot> set ylabel "mean travel time"
gnuplot> plot "emissions_30.txt" u 1:8 w l, "emissions_31.txt" u 1:8 w l, \
   "emissions_32.txt" u 1:8 w l, "emissions_33.txt" u 1:8 w l, \
   "emissions_36.txt" u 1:8 w l, "emissions_37.txt" u 1:8 w l, \
   "emissions_38.txt" u 1:8 w l, "emissions_39.txt" u 1:8 w l
```

One may see that for the last steps of the last iterations, the mean travel time differs only by 30s. This is really not that important...

Chapter 3. Tools

A tool which performs the Dynamic User Assignement automatically is discussed within the user documentation (see. "Automatic Iteration using 'dua-iterate.pl [http://sumo.sourceforge.net/docs/gen/user_chp05.shtml#user_chp05-dua-iterate]'").