TuST: from Raw Data to Vehicular Traffic Simulation in Turin

TuST: from Raw Data to Vehicular Traffic Simulation in Turin

Marco Rapelli
FULL Interdepartmental Center
Politecnico di Torino
Turin, Italy
marco.rapelli@polito.it

Claudio Casetti
FULL Interdepartmental Center
Politecnico di Torino
Turin, Italy
casetti@polito.it

Giandomenico Gagliardi
ITS Technology & Architecture Department
5T Srl
Turin, Italy
giandomenico.gagliardi@5t.torino.it

***Abstract*—Traffic simulations are becoming a standard way to study urban mobility patterns, to evaluate new traffic policies and to test modern vehicular technologies. For this reason, in recent years, mobility projects pushed towards an increase in the demand of traffic simulators and towards an extension of their area of investigation, aiming at covering a whole city and its suburbs. In this paper we describe the methodology we followed in the creation of a large-scale traffic simulation of a 400-Km^2 area around the Municipality of Turin. Our preliminary results demonstrate that a complete modeling of such a wide tool is possible at the expense of minor simplifications.**

***Index Terms*—Urban Mobility, Transportation Modeling, Road Traffic Simulation, Large-Scale Traffic Simulation.**

I. Introduction

According to UNICEF data [1], today more than 50% of the Earth's population lives in urban areas and by the half of this century this rate will exceed two-thirds. It is estimated that urban population increases by 60 million people every year. This continuous growth has significant environmental, economic and quality-of-life impact. For this reason, it is essential that urban planners develop adequate transport services and infrastructure systems that meet this growing demand. To this end, more and more mobility studies rely on simulation tools, which, thanks to the continuous progress in this field, are now capable of analyzing increasingly larger areas.

Urban mobility models provide the analysis of current traffic patterns and extracting data from them can be useful for predictive models of transport investments and new traffic policies. Indeed, simulation systems can produce a multitude of qualitative and quantitative information regarding traffic behavior, congestion management and emissions. Furthermore, those models are extremely important for the evaluation of Intelligent Transport System (ITS) applications, emerging vehicular technologies and solutions for Smart Cities, which would be not feasible or too expensive to test in a real urban scenario. However, building an urban model is not a trivial task. The larger the scale of the simulator, the more complex the project.

In the literature there are just few examples of traffic simulators that can be defined as 'large scale'. A very good example of a large-scale urban traffic simulator is the one using the TAPASCologne dataset representing the greater urban area of the city of Köln, Germany [2]. In this project, a 400-Km^2 area was modeled with a total of 1.2 million individual trips for a full-day scenario. Many other projects that aim to test vehicular technologies or Smart Cities solutions on large-scale scenarios chose to exploit this model as baseline because of its high level of realism and also due to the fact that it is a complete open-source project. It is one of the best and wider example for large-scale simulators to this day. Another case of urban mobility model is the traffic simulator of Bologna, Italy [3]. In this analysis, a set of traffic sensors was used in order to synthesize a typical weekday traffic pattern. It is not focused on the entire city, but it models 28 Km^2 around the city center, using an initial set of 22,000 vehicles for the morning rush hour. More recent studies are those of LuST (Luxembourg SUMO Traffic) scenario [4] for highways and primary roads of the City of Luxembourg and MoST (Monaco SUMO Traffic) scenario [5] for the Principality of Monaco and some adjacent areas. Those projects focus, respectively, on C-ITS applications and a parking-manager optimizer. In the LuST scenario a map of a 156-Km^2 area was exploited and, from an original dataset of 14,000 vehicles over 16 hours, only the morning and afternoon traffic peak hours were simulated. The MoST scenario represents a little coastal area following 35,000 vehicles in the morning rush hour. On a completely different scale, the ITS Austria West scenario [6] presents a huge real-life traffic monitoring system in SUMO with an infrastructure made of around 245,000 nodes and 320,000 edges with a total length of 27,000 Km. Its dataset is comprehensive of 1.2 million routes and 1.6 million vehicles for a full-day scenario, but, unfortunately, this scenario is not freely available to the community as it relies on proprietary information.

This paper presents TuST (Turin SUMO Traffic) an ongoing study to model the traffic pattern of the Turin metropolitan area and its neighboring districts over 24 hours, using the SUMO tool. We aim at contributing to the discussion on how to build an open-source, large urban simulator, by proposing a detailed description of the construction of a very large area model. In particular, we focus on aspects of map construction and traffic assignment from raw, aggregated traffic data provided by 5T [7], a public agency working on ITS and Info-mobility in the Italian region of Piemonte and partner of the City of Turin and of its public transport agency,

GTT [8].

The paper is organized as follows: in Section II, we present the use case we studied and the data from which we started. A complete and detailed description of the methodology and a discussion of the various options we faced is given in Section III. Then, in Section IV the results we obtained for the model are shown. Conclusions and future work are presented in Section V.

II. Case Study

The results presented in this paper were obtained from data owned by 5T. The dataset provided by 5T is comprehensive of all vehicle movements in the Province of Turin and constitutes the so-called Origin/Destination (O/D) matrix. Such an O/D matrix was created from a set of surveys in 2011 and then integrated and expanded with data from street sensors and traffic light sensors. It is now part of a more complex mobility system, called 5T-Supervisor. The matrix itself is a heterogeneous object: some parts are periodically updated with data collected by sensors, which cover large part of the active O/D relations, but not its totality; other parts have remained untouched since the original survey of 2011. Despite that, it is considered reliable and it is currently used by 5T and other agencies as the main instrument for mobility studies in the Province of Turin.

The O/D matrix file is a database where each record reports the traffic flow described as the number of vehicles per hour (Veh/h) for each origin/destination pair. Depending on the specific day, different O/D matrices can be observed. In particular, a distinction is made between weekdays, saturdays (including days before holidays) and sundays (including holidays), in three main periods of the year: school days, non-school days and "summer closure" days (typically 3 weeks in August). Therefore, the 5T Supervisor model consists of 9 different O/D matrices. Origins and Destinations in the matrix are outlined as Traffic Assignment Zones (TAZs), an aggregation of census areas largely used in mobility and transportation modeling. The Province of Turin, which is also the area monitored by 5T-Supervisor, counts 387 TAZs.

The case study we focused on is a 400 Km^2 area around the Municipality of Turin. The districts wholly or partially included in our map are Turin, Moncalieri, Nichelino, Borgaretto, Orbassano, Beinasco, Grugliasco, Collegno, Cascine Vica, Pianezza, Druento, Venaria Reale, Borgaro Torinese, Mappano, Settimo Torinese, San Mauro and Revigliasco for a total of 257 TAZs and more than 1,200,000 estimated population.

III. Methodology

A. Road Graph creation

In order to create a graph representing the road network of the study area, we exploited the JOSM tool [9], which is an extensible editor for OpenStreetMap in Java 8+. Using JOSM, it is possible to download all the data from the OpenSteetMap database [10], associated to a selected area of user's choice, obtaining an OSM file. The OSM file has an eXtensible Markup Language (XML) format and it contains all the information that can be geo-referenced and linked to a bidimensional topography map. Regarding the topography we can find shapes of buildings, parks and rivers. The represented infrastructure includes highways, roads and pathways as well as railways for trams or trains, bike paths and waterways for ships. We can also find points of interest such as monuments, tourist attractions, stadiums, bus stops. OSM files are the primary input for the creation of network files used by different simulation tools as SUMO, UrbiSim, MatSim, VISUM and many others.

B. Map construction

The mobility simulator we used in this project is SUMO (Simulator of Urban Mobility) [11]. SUMO is an open source, highly portable, microscopic and time-continuous road traffic simulation package designed to handle large road networks. In particular, we used the version 1.1.0, developed in December 2018. Being a microscopic simulator, each vehicle is modeled explicitly with its own route and it moves individually through the network. For this reason, it is also the most used tool for testing applications on vehicular technologies.

We used NETCONVERT [12] (one of SUMO's aid tool) to create a network file in SUMO-readable XML format from the OSM file of the area of interest. Since this study is focused on vehicular traffic only, the infrastructure considered in the OSM file was filtered. Some manual post-processing was needed for the occasional incorrect tagging. The resulting network is composed of almost 33,000 nodes, 66,000 edges and more than 6,500 Km of roads (more than 7,600 Km considering multiple-lanes roads).

Unfortunately the procedure so far described leads to a network with many errors. Some of them are due to imprecision in the OSM file (like the one described above regarding incorrect street tags), some others to the NETCONVERT conversion process. It is important to underline that, although our network aims to create a representation of the real Turin road map, it remains a model and it carries errors like all models do. Trying to fix all the inconsistencies is a mammoth task and it would be too time consuming. The only corrections we applied in the map, either manually or through Python scripts, are the ones that created extremely high, unrealistic traffic congestion situations.

The first editing required was regarding the speed limits assigned to network edges. SUMO assigns a nominal speed limit to streets, as imposed by traffic rules. This clearly differs from the average speed of real vehicles on that street. Indeed, a real vehicle often has to contend with asphalt conditions or lane restrictions due to, e.g., double-parked cars. Conversely, a real vehicle may travel at a speed greater than the limit, if traffic and road conditions allow it. For these reasons the speed limit of a network edge was set to the average speed that would be experienced by a vehicle if it were the only one in that street section. This situation is called Flow 0 traffic condition and it is the input we introduced in order to study how speed decreases in congestion periods. To extract the

Flow 0 average velocities we used an API of Google Maps called Distance Matrix [13]. Google API Distance Matrix yields the time in seconds that a vehicle would experience from a given Origin to a Destination choosing the depart time and the traffic conditions (optimistic, medium, pessimistic). Since we are interested in an ideal situation with no vehicles at all, we used the optimistic traffic condition option, starting at 4:30am, a very low congestion situation for Turin (and for most other cities). To automatize the process, a Python script was used to extract Flow 0 travel times from the Google API Distance Matrix and then to compute Flow 0 average speeds for all road segments in our map.

After that, a manual correction of the number of lanes was performed. Indeed, some primary roads had a smaller than expected number of lanes per direction of travel. This is mainly because the number of lanes in a SUMO street often depicts incorrect information: there is a large number of roads in Turin that have no lane-dividing line, but they still allow the presence of more than one vehicle in parallel because of their width. It is clear that squeezing traffic meant for a two-lane street onto a single lane leads to unrealistic congestion in the model. Unfortunately, no online database reporting the exact number of lanes was found. In order to obtain a more precise model, the full length of some primary road was checked using Google Maps Street View and the precise number of nominal or theoretical lanes was updated. Specifically, we manually edited more than 100 primary roads.

Regarding the connectivity, other problems presented themselves: SUMO provides only right-before-left priority for intersections and not also left-before-right as happens in roundabouts. For this reason, the connectivity of more than 500 roundabouts was manually fixed.

Another manual correction was required for the phases of traffic lights. Indeed NETCONVERT creates surreptitious traffic lights phases in intersections tagged as regulated by a traffic light in the OSM file. The phases created by SUMO often do not suit well the traffic pattern of Turin and a too long or too short traffic light cycle could cause very long queues at the intersection. For this purpose, data for more than 900 traffic lights were provided by the 5T-Supervisor. Phases of actual traffic lights in Turin change throughout the day following a phase-adaptation algorithm called UTOPIA based on live traffic patterns. For the sake of simplicity, at least for the moment, in our model we used fixed traffic light phases. In order to do this, phases of light were averaged over the day and then manually inserted in more than 700 traffic-light-regulated intersections.

Additionally, some synchronization was needed in order to have the so called green-wave effect on a road. SUMO provides a Python script called `tlsCoordinator.py` that manages the traffic light synchronization, given the traffic pattern of a single hour. The `tlsCoordinator.py` script works pretty well but it nevertheless needs some manual corrections. After the described correction, the map we ended up with is shown in Fig. 1.

Traffic lights managing in SUMO is not a simple task and it still represents an ongoing work in our model. Consequently, unless stated otherwise, the results presented in the next section were obtained using a network where there are no traffic lights and all intersections are priority-based.

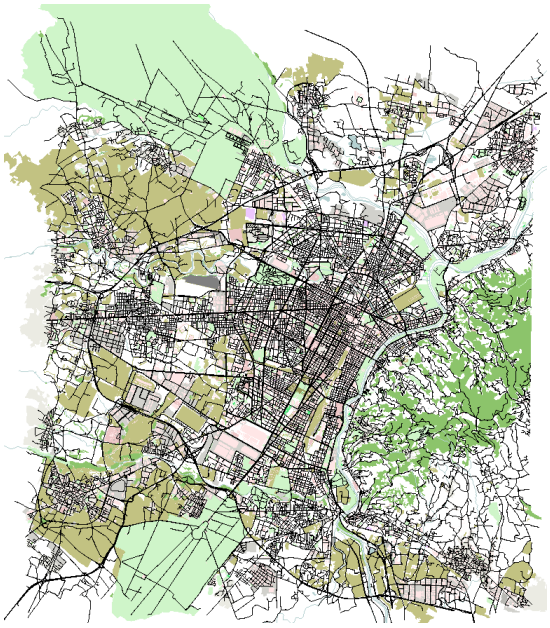

Fig. 1: Case study map.

C. Traffic Assignment algorithms

Following the creation of a realistic map, the next step is the definition of a realistic input for our model, derived from the traffic patterns in the 5T O/D matrix. From the original matrix, only the traffic flows with Origin and Destination in a TAZ inside the map described above were extracted, thus creating a new O/D matrix focused on our study area. The traffic pattern chosen for the analysis is a typical school weekday, which results in approximately 2,200,000 car trips for the analyzed map. Those numbers make our model one of the most extended examples in the literature for a large-scale microscopic traffic simulator. Comparing it with the ITS Austria West scenario, our model aims to simulate about twice the number of vehicles in a much smaller area, resulting thus in a much greater complexity.

Through appropriate SUMO input files, we translated the O/D matrix into SUMO trips, i.e., to map them into origin edge, destination edge and suitable depart times (the latter generated randomly within each hour).

Generating an O/D edge pair and a depart time according to the flow is quite a trivial task. The complexity is, given an O/D pair and a time of departure for each vehicle, choosing a route that leads the system to a stable situation even in congestion conditions. The algorithms in charge to do this are called Traffic Assignment algorithms. There are many examples in the literature of Traffic Assignment algorithms and many studies that compare their performance. There is no optimal Traffic Assignment method, and its performance depends on

the use case. For this reason in this study different Traffic Assignment algorithms were tested to find the one that best suits our model. Below, we provide a quick overview of some basic Traffic Assignment algorithms [14], their implementation in SUMO, and the suitability to our model.

- The All-or-Nothing assignment is the simplest one. It is basically a Dijkstra algorithm for shortest path search in a graph with weights. In our case the used weights are the edge travel times. All-or-Nothing assignment is implemented in SUMO and called DUAROUTER [15]. It computes the shortest path for all vehicles at time 0. This of course leads to unrealistic results. For example, we can imagine to have two alternate paths linking an O/D pair. The travel times of those two paths are very similar, but one is slightly greater than the other. Running a DUAROUTER for this example will result in having all vehicles choosing the path with a slightly better travel time and no one choosing the other, which is clearly a solution far from reality. Nevertheless the All-or-Nothing assignment is used as a building block for all other Traffic Assignment algorithms.
- The User Equilibrium (UE) assignment is based on Wardrop's first principle [16] which states that, after a user equilibrium condition is reached, no drivers can unilaterally reduce their travel time by shifting to another route. This assignment algorithm leads to an equilibrium solution, however, it is not feasible for large-scale models: for every new vehicle, all drivers should recompute the travel times of all the paths present in the map, which in our case is an extremely large number. To overcome this problem, SUMO introduced a parameter in order to choose the number of alternate paths a vehicle should take into consideration. Those paths are the n-shortest ones computed with DUAROUTER. The UE assignment algorithm works under the assumptions that the travel time on a given link is a function of the flow in that link only and that drivers have the same perception of the travel time on a path. For this reason, it is not a realistic solution even if it leads to an optimum equilibrium scenario.
- The Stochastic User Equilibrium (SUE) assignment extends the UE assignment, introducing randomization in the route-choice algorithms for the selection among alternate paths. This is due to the fact that travel time perception differs among drivers. The SUMO version supports two different route-choice algorithms: Gawron and Logit [17]. Gawron computes the probability of choosing a route based on the travel time along the used route in the previous simulation step, the sum of the edge travel times over the alternate paths and the previous probability of choosing a route. Logit, instead, applies a fixed formula to each route to calculate the new probability. It ignores old costs and old probabilities and takes the route cost directly as the sum of the edge costs from the last simulation step. The probabilities are calculated from an exponential function scaled by the sum over all route values. Similar to UE, the SUE assignment leads to an equilibrium solution, however it seems more appropriate for low-congestion traffic conditions such as off-peak periods or lightly travelled rural areas.
- The Incremental assignment, instead, does not aim to achieve an equilibrium solution as UE or SUE. It is a much simpler greedy algorithm in which fractions of traffic volumes are assigned in steps based on All-or-Nothing assignment. To execute it, SUMO computes a DUAROUTER assignment for each vehicle, but at its depart time instead of at the beginning of the simulation. In this way a vehicle will compute its optimal route based on the current traffic condition, which is actually a very good approximation of what happens in real life.

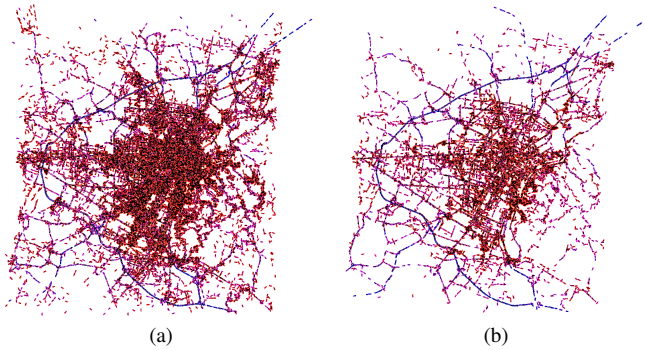

Fig. 2: Comparison between Traffic Assignment methods. Red vehicles are slow or stopped, blue ones are moving fast. (a) All-or-Nothing Assignment on single iteration. (b) Incremental Assignment after 50 iterations.

D. Iterative Traffic Assignment

None of the Traffic Assignment algorithms described above are optimal if used on a single run. Indeed, in order to achieve a good assignment, they are all used in an iterative way. The difference between a single run method and an iterative one is depicted in Fig. 2, where in (a) we have an All-or-Nothing Assignment on a single iteration, in (b) an Incremental Assignment at the 50^{th} iteration is shown.

There are two methods for Iterative Assignment: Microscopic Traffic Assignment method and Macroscopic Traffic Assignment method. In Microscopic Traffic Assignment method a simulation is run for each iteration and travel times on network edges are measured. Then travel times obtained in this way are given as input to the Traffic Assignment algorithm on the next iteration step and the cycle goes on until the maximum number of iterations is reached. SUMO performs this method with the DUAITERATE tool [18], which runs DUAROUTER for an entire simulation in each step. This is the best method for Traffic Assignment, unfortunately it is too time consuming: in order to have an optimum Traffic Assignment, a great number of simulations have to be run.
The Macroscopic Traffic Assignment aims to imitate the

Microscopic one, but in a faster way. Indeed, instead of running a simulation for each step, it leverages mathematical resistive functions that approximate the travel time increase when the flow increases. In SUMO this method is exploited by MAROUTER [19], a tool that uses a hard-coded capacity-constraint function based on speed limits, lane numbers and edge priorities to compute travel times and flows from traffic density. The Macroscopic Traffic Assignment method is of course less precise than the Microscopic one: due to approximations of mathematical functions, travel times are computed instead of measured. Nevertheless the output reached can be considered good enough.

MAROUTER implements a Macroscopic Traffic Assignment method supporting two Traffic Assignment algorithms: Stochastic User Equilibrium and Incremental (User Equilibrium is not yet implemented). SUE is the best among the equilibrium algorithms and can be used in MAROUTER with two route-choice algorithms: Gawron or Logit. For both these route-choice algorithms it is possible to select the number of alternate paths to consider and the number of iterations to perform before stopping the equilibrium search. It is also possible to stop the iterations when the difference between two consecutive runs is below a given tolerance, instead of after a fixed number of iterations. Unfortunately in large-scale networks, this condition is very difficult to reach in a reasonable time, so we decided to use a fixed number of iterations that does not lead to high processing times. Indeed the SUE algorithm has two levels of optimization implemented in nested loop cycles, namely inner and outer iterations. It is possible to set the maximum number of iterations for either of those cycles, in order to have a trade-off between the level of optimization reached and the processing time. In particular we tested SUE-Gawron and SUE-Logit both with 50 iterations: 5 inner iterations and 10 outer iterations and vice versa.

For each of those solutions we choose 2 or 5 alternate paths as sets for the route-choice algorithms. The processing times for those options are already quite large: in order to complete all iterations 20 to 35 minutes are required.

Incremental Assignment, instead, has only few options that can be used. In particular we tested its performances for 5, 10, 50 and 100 iterations. The first outcome to notice is that processing time of Incremental Assignment is significantly lower compared to SUE: for 50 iteration Incremental stops after 5 minutes, while 10 minutes are required to complete 100 iterations. This is because Incremental Assignment is a greedy algorithm that does not search for an optimum equilibrium condition, as SUE does. Thus it does not explore all the solutions set but looks for a local-optimum in each step.

In the next section the performances of all the options described above are compared and commented and a brief explanation of some open issue of our model is given.

IV. Results

Before going on to detail the results, it is important to underline that the modeling of the presented system is still ongoing, so the results presented here are not conclusive.

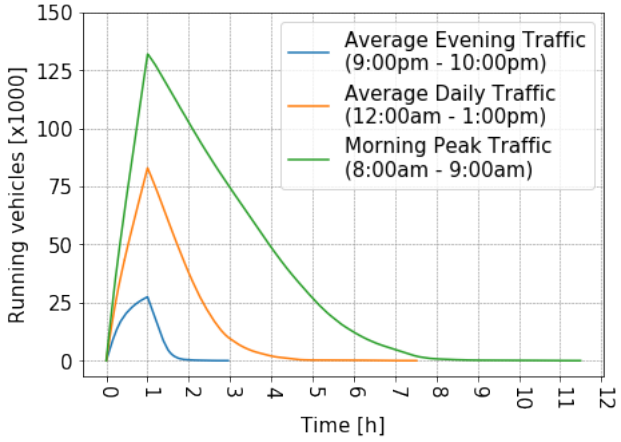

Fig. 3: All-or-Nothing Assignment over single iteration. Comparison among three time periods.

Rather, they are intermediate results that lead to a stable point, but are still not the solution we are looking for, that is, the one that more closely resembles the traffic pattern in Turin, on an average weekday, over a 24-hour period. All outputs presented were obtained by averaging over 5 simulations.

In order to evaluate the goodness of our model and to compare the performance of different Traffic Assignment methods, we conducted a stability study. It consists in introducing a certain traffic pattern in our model with the aim of analyzing the time needed to let all drivers complete their journey. The main metric to inspect is thus the time needed to empty the network.

A first example of stability study in shown by the plot of Fig. 3: the model used was an All-or-Nothing Assignment over a single iteration. Three different traffic patterns are represented: one hour of the morning peak in green, an average daily hour in yellow and, in blue, one hour of the evening traffic pattern. We can see that the maximum reached by the three lines differs because of the distinct traffic loads during those periods of the day. It is important to underline how the blue line begins to flatten out before reaching its maximum: this behavior indicates that the number of vehicles exiting from the network is becoming similar to the one entering it and a situation of stability is going to be reached. Such a stability is not reached in the other two lines. Indeed we can read from the graph that more than 7 hours were needed to serve all drivers of an average daily hour of simulation, while more than 11 hours were requested to empty the network if a morning peak hour is injected. These underwhelming results are due to the choice of the Traffic Assignment method: we will see later on how they can be improved with different Traffic Assignment algorithms and with multiple iteration runs. For the next results we will always focus on the morning peak hour (the green line in the previous plot), from 8:00am to 9:00am, because it covers the maximum number of vehicles injected in a daily traffic pattern.

Next, we show and compare the performance of the SUE

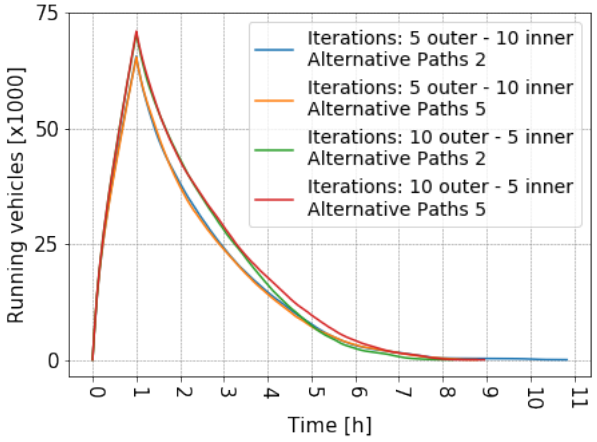

Fig. 4: SUE Assignment over 50 iterations with Gawron model.

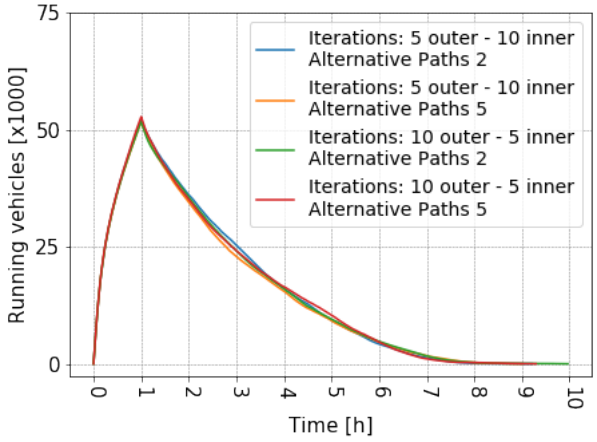

Fig. 5: SUE Assignment over 50 iterations with Logit model.

Traffic Assignment: in Fig. 4, we used the Gawron mathematical model for the route-choice algorithm, while in Fig. 5 the Logit method was explored. For both cases the test was performed using either 5 outer and 10 inner iterations or 10 outer and 5 inner iterations and 2 or 5 alternate paths. First of all, it can be noticed that there is basically no difference between all the options: there is just a little discrepancy between different iteration choices in the Gawron method. There is instead a significant improvement between the two mathematical methods. Indeed, while for Gawron we reached almost 75,000 vehicles in the map at the end of injection, we had only 50,000 for the same traffic pattern using Logit. This resulted in a shorter time needed in order to empty the map. Furthermore it is possible to see how the Logit curve bends more than the Gawron one before its peak, showing that the situation is more stable with respect to Gawron. However those results are anyway unsatisfying: still more than 9 hours were needed in order to serve all drivers of a single peak hour. It is important to underline that both SUE-Gawron and SUE-Logit are good Traffic Assignment algorithms. However the performance of such algorithms is strongly related to the model they are applied on.

Fig. 6 reports the output of the tests run using Incremental Traffic Assignment. Different number of iterations (5, 10, 50 and 100 iterations) were used for the performance evaluation. We can notice how the maximum number of vehicles reached after the injection is slightly lower than the SUE Logit case (less than 50,000 vehicles), while, more remarkably, the number of hours needed to empty the network is now down to 3 and a half. Furthermore, it can be appreciated how the difference between the curves decreases when the number of iterations increases and also how the curves bend before their maximum. Those signals indicate that a stability condition is reached and it is notable that 50 iterations are sufficient in order to obtain it. Drivers in real traffic in Turin during the morning peak hour can run into situations of very high congestion, especially on the outer ring road, easily leading

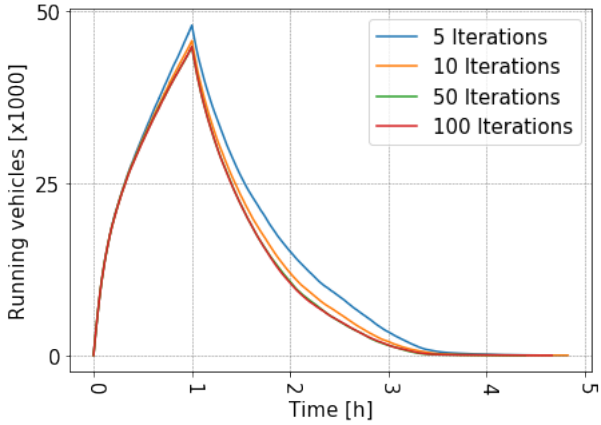

Fig. 6: Incremental Assignment.

to hour-long trips across town. The situation simulated here showed a moderately worse situation, which can be anyway claimed as good enough due to model complexity. Further improvements are of course still being sought.

The results presented so far all refer to a single hour of traffic. When we try to inject a full-day traffic pattern, the problems that crop up in a single hour are further compounded. Fig. 7 shows in blue the number of vehicles running in the model for a full-day traffic pattern of a typical weekday of Turin, while the red line displays the traffic injected according to the O/D Matrix. Those results, as well as the others presented below, were obtained using 50 iterations of the Incremental Traffic Assignment method. It is easy to notice how the simulator here leads us to an unrealistic situation, with 125,000 vehicles still running in the map at midnight. A worse result is obtained if traffic lights are introduced in the model. As shown in Fig. 8, the number of vehicles grew to 150,000 at the end of a 24-hours simulation for a model with traffic lights. It is to be remarked that these traffic lights, though set up with real average phase cycles derived from 5T data and synchronised usign `tlsCoordinator.py`, still lack the dynamic quality and actual synchronization present in

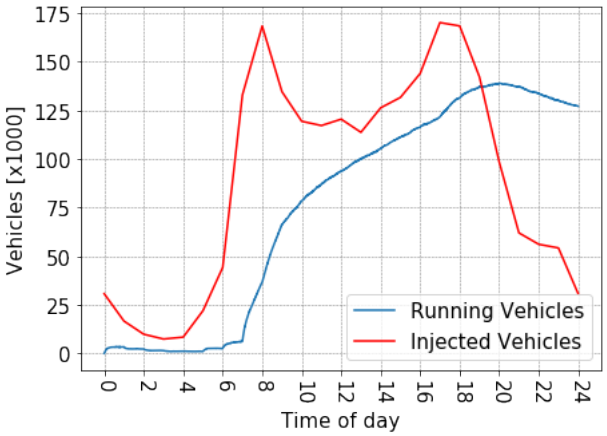

Fig. 7: Full-day traffic pattern.

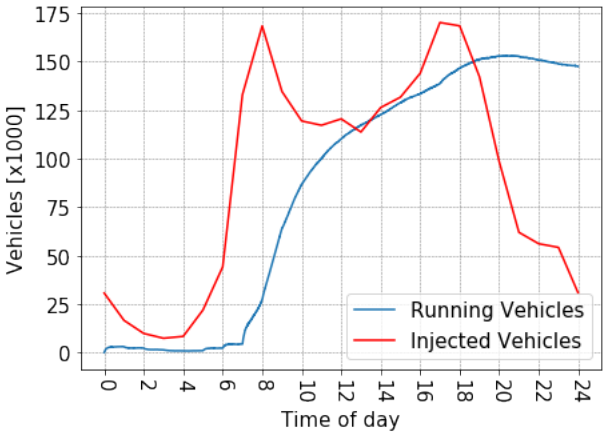

Fig. 8: Full-day traffic pattern with traffic lights.

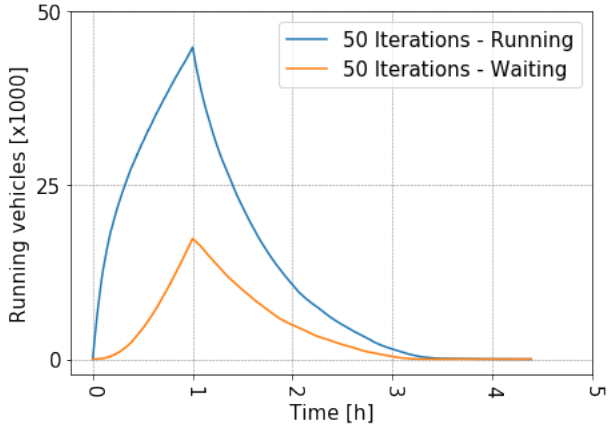

Fig. 9: Waiting vehicles over single morning peak hour.

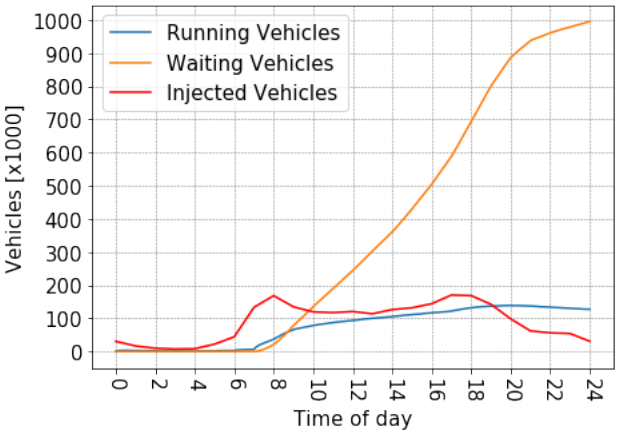

Fig. 10: Waiting vehicles over full-day traffic pattern.

Turin's real traffic lights.

The large residual traffic present at the end of the 24-hour cycle is due to the effect of waiting vehicles. The waiting vehicles are those that are scheduled to enter at a given time, but, due to congestion in the origin area, do not find room in the map. Those vehicles have to wait for the congestion to alleviate in that area, and will be inserted later than it was planned. If we consider a very complex scenario with many vehicles to be inserted in a large period of time, like in the full-day scenario, it can be argued that there are parts of the map where vehicles are always introduced and congestion never has the time to decrease. This leads to a situation in which the cars that cannot enter basically wait forever and vehicles stuck in queue never exit from the model, so we see the number of running vehicles growing enormously. This effect is mitigated if simulations of a single hour of traffic are performed, as shown in Fig. 9, while the cars waiting increase exponentially for the full-day scenario in like in Fig. 10. In this case approximately 1 million vehicles (out of the 2 millions that should be injected in total) cannot enter the map.

In order to study the portion of traffic we can handle *at this stage of model development*, the number of injected vehicles was scaled down to 20% of the total daily pattern. In this condition, the simulator can correctly manage the full-day scenario and the number of vehicles waiting to enter is negligible as displayed in Fig. 11.

V. Conclusions

As the continuous growth of Earth's population pushes the urbanization process, urban planning for infrastructures and traffic analysis are becoming crucial. More and more research relies on large-scale simulation tools to conduct mobility analysis or evaluation of ITS applications for vehicular technologies and Smart Cities applications. However, the literature concerning models that can investigate wide-ranging scenarios, up to entire cities, is not as rich as one would expect.

In this work we presented how to build complex mobility model using open-source tools and which kind of data are necessary for its realization. We focused on a 400-Km^2 area around the municipality of Turin and showed how the crucial parts for a well-functioning system rely on sound network construction and an optimal Traffic Assignment method choice.

In the results we compared the performance of our model using different Traffic Assignment algorithms with distinct

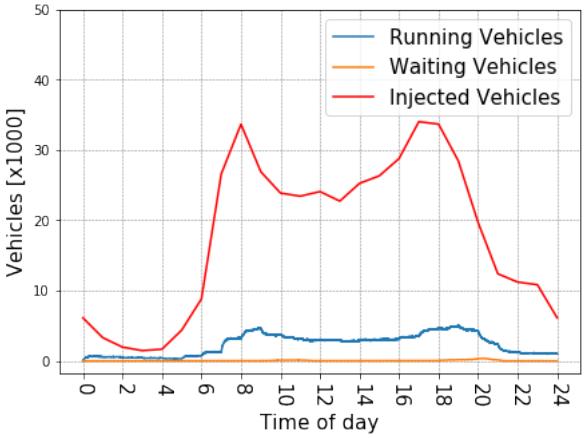

Fig. 11: Full-day traffic pattern scaled by 20%.

options. The outputs highlight how the options of each method leads to similar outcomes, while the choice of an appropriate Traffic Assignment algorithm is indeed essential. Our model is shown to be capable of addressing and managing a single peak-hour traffic based on real Turin traffic pattern data. Some problems arise when a full-day traffic is injected in the system.

In future works we intend to continue to improve our model, in order to be capable to deal with a wider time window. In this way we could claim to have a perfect traffic mobility model, whose traces can be used for simulation studies involving vehicular communication and even urban planning.

ACKNOWLEDGMENT

The present research was conduct in the Future *Urban Legacy* Lab (F*UL*L), an interdisciplinary and interdepartmental laboratory of Politecnico di Torino.

REFERENCES

[1] UNICEF, "Condition of childhood in the world 2012 – Sons of cities," original version "La condizione dell'infanzia nel mondo 2012 – Figli delle città," http://www.unicef.it/Allegati/SOWC_2012_ITA.pdf, 2012.
[2] S. Uppoor, O. Trullols-Cruces, M. Fiore, and J. M. Barcelo-Ordinas, "Generation and analysis of a large-scale urban vehicular mobility dataset," IEEE Transactions on Mobile Computing, vol. 13.5, pp. 1061–1075, 2013.
[3] V. Caiati, L. Bedogni, L. Bononi, F. Ferrero, M. Fiore and A. Vesco, "Estimating urban mobility with open data: A case study in Bologna," IEEE International Smart Cities Conference (ISC2), pp. 1–8, September 2016.
[4] L. Codecá, R. Frank, S. Faye and T. Engel, "Luxembourg sumo traffic (lust) scenario: Traffic demand evaluation," IEEE Intelligent Transportation Systems Magazine, vol. 9.2, pp. 52–63, 2017.
[5] L. Codecá, J. Erdmann and J. Härri, "A SUMO-Based Parking Management Framework for Large-Scale Smart Cities Simulations," IEEE Vehicular Networking Conference (VNC), pp. 1–8, December 2018.
[6] K. H. Kastner and P. Pau, "1 Experiences with SUMO in a Real-Life Traffic Monitoring System," SUMO 2015–Intermodal Simulation for Intermodal Transport, vol. 1, 2015.
[7] "5T Torino," http://www.5t.torino.it.
[8] "GTT Gruppo Torinese Trasporti," http://www.gtt.to.it/cms.
[9] "JOSM wiki," https://josm.openstreetmap.de.
[10] "OpenStreetMap Wikipedia," https://it.wikipedia.org/wiki/OpenStreetMap.
[11] "Simulation of Urban MObility – Wiki," https://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-_Wiki.
[12] "NETCONVERT," https://sumo.dlr.de/wiki/NETCONVERT.
[13] "Google Maps Platform – Distance Matrix API Documentation," https://developers.google.com/maps/documentation/distance-matrix.
[14] T. V. Mathew and K. V. Krishna Rao, "Introduction to Transportation Engineering," in Traffic Data Collection, NPTEL, 2007.
[15] "DUAROUTER," https://sumo.dlr.de/wiki/DUAROUTER.
[16] J. G. Wardrop, "Wardrop of Some Theoretical Aspects of Road Traffic Research-Road Paper," in Road Engineering Division, 1952.
[17] "Demand/Dynamic User Assignment," https://sumo.dlr.de/wiki/Demand/Dynamic_User_Assignment.
[18] "Tools/Assign," https://sumo.dlr.de/wiki/Tools/Assign.
[19] "MAROUTER," https://sumo.dlr.de/wiki/MAROUTER.