Using a Traffic Simulator for Navigation Service

Abdullah AlDwyish, Hairuo Xie, Egemen Tanin, Shanika Karunasekera and Kotagiri Ramamohanarao
The University of Melbourne, Australia
aldwyish@student.unimelb.edu.au,{xieh,etanin,karus,kotagiri}@unimelb.edu.au

ABSTRACT

Traffic congestion is a serious problem that is only expected to get worse in the future. Statistics shows that half of traffic congestion is caused by temporary disruptions like accidents. These events have dramatic impact on road network availability and cause huge delays for commuters. Also, they are usually unexpected and hard to manage by traffic authorities. State-of-the-art navigation systems started to provide real-time information about traffic conditions to help users make better routing decisions. However, traffic in the road network changes rapidly and the advice calculated now may not be valid after few minutes. This is especially critical in the presence of traffic incidents, where the impact of the incident could cause traffic to propagate to nearby roads. Thus, it is important for navigation systems to consider the evolution and future impact of traffic events. In this work, we present a navigation system that uses faster than realtime simulations to predict the evolution of traffic events and help drivers proactively avoid congestion caused by events. The system can subscribe to real-time traffic information and forecast the traffic conditions using fast simulations. We evaluate our approach through extensive experiments to test the performance and accuracy of the simulator with real data obtained from TomTom Traffic API. Also, we test the quality of navigation advice in realistic settings and show that our solution is able to help drivers avoid congested areas in cases where even real-time update methods lead drivers to congested routes.

CCS CONCEPTS

•**Information systems** →*Location based services;*

KEYWORDS

Spatial Databases, Location-based Social Networks

ACM Reference format:
Abdullah AlDwyish, Hairuo Xie, Egemen Tanin, Shanika Karunasekera and Kotagiri Ramamohanarao. 2017. Using a Traffic Simulator for Navigation Service. In *Proceedings of SIGSPATIAL'17, Los Angeles Area, CA, USA, November 7–10, 2017,* 4 pages.
DOI: 10.1145/3139958.3139998

DOI: 10.1145/3139958.3139998

1 INTRODUCTION

Traffic congestions can have a significant impact on economy and environment of large cities. According to Texas A&M Transportation Institute, congestion cost America $160 billion in 2014 [1]. Due to traffic congestion, American drivers traveled an extra 6.9 billion hours, around 42 hours on average per commuter in 2014. These estimates are only expected to get worse. For example, in Australia, congestion cost was estimated to be 16 billion (AUD) in 2015 and expected to jump up to 37 billion [2] (AUD) in 2030. Traffic incidents (such as accidents, emergency maintenance or public events) are responsible for more than 50% of delays caused by congestion [3, 6]. Such traffic incidents have a dramatic impact on road capacity and availability. They also have an inconsistent impact that depends on many spatial and temporal variables which makes them hard to be embedded in any generic decision making method.

To combat traffic congestion, navigation systems started to provide information about the traffic conditions of the road network in real-time. However, these navigation systems route drivers based on current traffic conditions without taking into account future evolution of an incident. Traffic in road networks can change rapidly and the advice calculated now may not be valid after few minutes for incidents such as car accidents. This is especially critical in the presence of non-recurrent incidents where the impact of the incident introduces significant variation in the traffic conditions around the incident. Thus, routing based on current traffic conditions will risk leading drivers to unexpected congestion. To overcome this shortcoming, navigation services need to consider the future impact of traffic events. Knowing how events evolve in traffic is key to bringing more realism and improving the navigation advice [7].

In this work, we propose a system to forecast and quantify the future impact of traffic incidents using fast simulations to provide real-time navigation advice. In our approach, we use real-time traffic updates to estimate the current traffic flow and forecast the future traffic by using a faster-than-real-time microscopic traffic simulator, called SMARTS [8]. Using a snapshot of current traffic state on the road network we can use traffic simulation models to predict the future conditions. We retrieve traffic snapshots periodically and use the snapshots to validate and calibrate the simulation. Route navigation advice generated by the simulator can be more effective than historical route guidance or advice based on current conditions because it considers the spatio-temporal evolution of traffic.

To illustrate the benefit of foreseeing the evolution of traffic events with our novel system, consider the following scenario: a driver who needs to avoid a traffic incident on the way to work.

[1] https://mobility.tamu.edu/ums/
[2] https://bitre.gov.au/publications/2015/is_074.aspx

The driver uses a navigation system, and the system based on current traffic conditions advises the driver to take a short detour to avoid the road with the incident. However, this detour is close to the incident location and by the time the driver reaches an intersection adjacent to the incident, the driver might get stuck due to the propagation of traffic congestion. In contrast, using a traffic simulator we can forecast how traffic would evolve over time and compute a better alternative route to successfully avoid the event.

To predict incident impact, the literature has focused on machine learning techniques [1, 2, 7]. However, these techniques are based on historical data and they can not adapt to new situations. On the contrary, the simulation-based approach does not require historical data and is more adaptive to particular road and traffic situation. There has been a significant body of work on transportation simulations [4, 5, 9]. However, popular open-source and commercial simulators are not designed for use in real-time navigation. In this work, we use SMARTS [8] a distributed large-scale microscopic traffic simulator that is able to scale up and perform fast simulations allowing for short-term estimations to be done fast. SMARTS performs simulation in a continuous spatial domain in real-time. Using traffic simulation allow us to predict the immediate future state of the road network without the need for historical data. Also, we can adapt and adjust simulation scenarios based on real-time traffic data. This is a novel approach for navigation services.

2 SYSTEM OVERVIEW

An overview of the system is presented in Figure 1. We propose a client-server architecture, where the client runs on a modern mobile phone or a navigation device. Using the client, the user can send a routing request providing the source and destination. The server is organized based on three main functions: (i) event monitoring, (ii) traffic prediction based on simulation and (iii) navigation advice generation. The system uses real-time traffic information provided by traffic authorities or third party API. Also, the system can receive traffic event reports from traffic authorities or from an external event detection component.

The **Event Monitor** keeps track of active traffic events, and manage the simulation during the lifetime of a traffic event. When the system receives an event report, it stores the event information in the active event memory and sends a simulation request to the simulation center. When the traffic event is cleared, the Event Monitor will send a request to the simulation center to terminate the simulation.

Once the **Simulation Center** receives a simulation request, it uses the most recent snapshot of the road network conditions and the event information to run a simulation and stream the output of the simulation to the graph store. The flow estimation component determines the current traffic conditions of the road network given current traffic data. The traffic prediction component forecast the future traffic by using our microscopic traffic simulator SMARTS [8]. The simulator employs multiple traffic models (such as car-following and lane-changing), and the simulated vehicles obey various road rules. The simulator is used to forecast and estimate future travel time for roads.

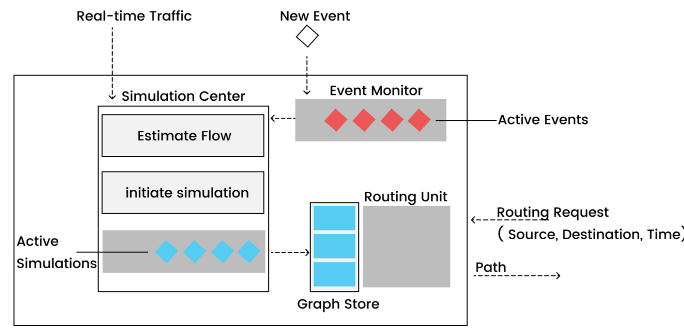

Figure 1: Overview of the System

The navigation advice component consists of the **Routing Unit** and the **Graph Store**. This component will perform route planning using the predicted travel times that are generated by the simulator. When the server receives a routing request, the **Routing Unit** will initiate a graph search and use the graph store to assign costs to edges while searching for a route. The **Graph Store** is an indexed data store with the goal of providing fast access to relevant travel costs to be used during the search process. It uses the connectivity of the road network as an indicator of locality.

3 TRAFFIC PREDICTION VIA SIMULATION

In this section, we describe how the system uses the SMARTS [8] simulator to forecast traffic conditions in the road network and generate navigation advice.

For every traffic event report, the system starts a simulation session for the event area. A simulation session lasts until the event is cleared. In each session, the system executes a repeated cycle of flow estimation and simulation in a sliding window mode. For example, assume the system receives an event report at 12:00 pm and starts a simulation session. First, the system performs flow estimation using the 11:55 am snapshot of the traffic in the area. Then, the simulation will run and complete within 5 minutes. The simulation generates the travel time for a 30 minute window into the future (until 12:30 pm) and streams the estimated output to the graph store. Each simulation session is run separately and we can run multiple simulation sessions in parallel for different areas based on incoming events reports.

The first step in the simulation process is to estimate the current traffic flow. The estimate depends on the real-time traffic information maintained by the system. We assume the data we receive from a service provider is either a vehicle volume or speed per road. We first calibrate traffic by filling up road links with random vehicles. The number of vehicles inserted to a link is determined by the average speed of vehicles on the link, which is extracted from service provider data. The number of vehicles is computed with Formula 1, in which N is the number of vehicles, l_{edge} is the length of the link, n_{lane} is the number of lanes on the link, $l_{vehicle}$ is the average length of vehicles, v is the average speed of vehicles and T is the safety headway in the car-following model.

$$N = \frac{l_{edge}}{l_{vehicle} + v \times T} \times n_{lane} \tag{1}$$

Once the simulation is started, we maintain the rate that new traffic enters to the simulation area from the border of the area.

During the simulation, the simulator periodically estimates the speed of each road based on the simulated vehicles on the road at a given time interval. For example, for a 5-minute interval, the simulator computes the average road speed every five minutes based on all the vehicles that used the road during the past five minutes. This estimate is exported at the end of the simulation for every road to the graph store.

The navigation advice is generated by traversing the graph store using a time-dependent graph search algorithm similar to Dijkstra algorithm, called SPA. In SPA, we modify Dijkstra's edge relaxation step to use the graph store to find the edge cost while exploring nodes. The algorithm maintains a group of nodes in a priority queue Q to be processed. These nodes are the frontier of paths from the source node s that have been explored so far. The cost of each frontier is the sum of each edge travel time from the source node to the frontier. It is important here that each frontier in Q maintains its own timeline. Therefore, we calculate the timestamp used to query the graph store based on the cost of the frontier $C[u]$ and the query start time qt at each iteration. During edge relaxation, we extract the frontier u with the minimum cost and update the timestamp to $t = qt + C[u]$. For every node v adjacent to u that has not been settled, we query the graph store to find out the cost $w(u, v, t)$ of traversing edge (u, v) at the timestamp t. The new cost from source is calculated using $C[u] + w(u, v, t)$. If this value is lower than the previous cost $C[v]$, then we update the cost of v to $C[u] + w(u, v, t)$ and set u as the predecessor node of v. This process is repeated until the destination node is extracted from the queue.

4 EXPERIMENTAL STUDY

In this study, we provide a set of experiments to test the simulator speed and prediction accuracy, as well as the quality of navigation advice. Experiments in this section were executed on a desktop computer with a 2.7 GHz Intel Core i5 processor and 16GB RAM. The OS is macOS 10.12.4 and the code was implemented in Java 1.8.

4.1 Evaluation of the Simulator Speed

In this section we show that the simulator can run significantly faster than real-time. We measure the simulator performance using the real-time factor, which is calculated by dividing prediction interval over the length of the execution time. For example, if it takes the simulator 5 minutes to simulate 20 minutes of traffic, the real-time factor is 4. Table 1 shows the parameters for this experiment (default in bold), for each setting we run the experiment 5 times and average the results. Note that the number of vehicles remains constant during the simulation, a number of vehicles of 1000 implies that there will be 1000 vehicles at any time during the simulation.

First, we evaluate the speed of the simulation for different prediction intervals. We found that the real-time factor for simulating 1000 vehicles for 30 minutes is 51.7, for 60 minutes is 54.5 and for 90 minutes is 57.1. This means that the simulator can simulate 30 minutes of traffic in under one minute and about 70 seconds to simulate traffic for 60 minutes and less than 2 minutes to simulate for 90 minutes. Furthermore, we increase the number of the vehicles to 5000, 10000. The real-time factor for simulating 5000 vehicles is 16.1 and for 10000 vehicles is 8.1. The results show that the simulation is still capable of simulating 5000 vehicles in about 2.5 minutes and 10000 vehicles in about 4 minutes. Next, we test the simulator performance on larger road networks. We selected two other suburbs with different road network layouts: Hawthorn (an inner suburb of Melbourne) and Ringwood (an outer suburb). The size of these areas are $2.4km^2$, $18km^2$ and $28km^2$ for the Melbourne CBD, Ringwood and Hawthorn respectively. Ringwood simulation has a real-time factor of 42.9 and Hawthorn simulation is 37.6. The real-time factor decreases slightly for bigger areas since the number of roads that need to be processed increases. However, the simulator is still fast enough to simulate 30 minutes of 1000 vehicles under one minute.

Table 1: Parameter settings for simulator speed experiment

Parameter	Range
Map	{ **Melbourne CBD**, Ringwood, Hawthorn}
Number of Vehicles	{**1000**, 5000, 10000}
Prediction Interval	{**30 min**, 60 min, 120 min}

4.2 Evaluation of Simulation Accuracy

In this section, we evaluate the simulation prediction accuracy using real data obtained from TomTom Traffic API [3]. We conduct multiple tests on two major highways in Melbourne and measure the accuracy of the prediction. The accuracy of the simulation is the average accuracy of all simulated road links. During the simulation, we collect the predicted speed $\hat{v}$ for each road link and compute the accuracy following the definition in Formula 2, in which v_r is the actual speed of the link that is extracted from the real data, and $\hat{v}$ is the average speed of vehicles during the simulation for that specific link.

$$accuracy = \begin{cases} 1 - \frac{\hat{v} - v_r}{v_r}, & \text{if } |\hat{v} - v_r| < v_r \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

In this test, we predict the traffic state for two major highways in Melbourne: Eastern Freeway (10.2 km) and City Link road (9.2 km). We measure the accuracy while simulating for four prediction intervals (2m, 30m, 60m, 90m). The traffic is calibrated only once at the start. For example, we can get TomTom data at 8am and calibrate the simulation using the 8am data. Then compare simulated traffic with the real data for 8.30 am. We ran the simulation during the morning peak hours for 90 minutes in real time. The results for the prediction accuracy are (86.2%, 71.6%, 73.5%, 79.1%) with standard deviation of (3.1, 20.5, 17.9, 25.9) for the intervals (2m, 30m, 60m, 90m) respectively. The results shows that the simulator can achieve a high accuracy for near future interval (2m). The accuracy for predicting distant future is lower but is still higher than 70%. Overall the simulator achieves 77.5% on average for all prediction intervals. We deem these four intervals an acceptable minimum to maximum travel time in a city. Also, deviation increases when the prediction interval increases. This is understandable as the traffic can be affected by a number of random factors in the real world during a longer period of time.

[3] http://developer.tomtom.com

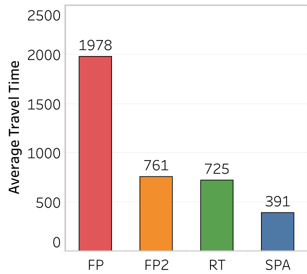

Figure 2: ATT values for the incident on Melbourne CBD location

4.3 Evaluation of Navigation Advice

In this section, we run a simulation-based evaluation to test the effectiveness of the navigation advice produced by our system. In this experiment we create a traffic event, and test the ability of the routing method to re-route a set of test vehicles away from the event.

Metrics and baseline: We use the simulated *Average Travel Time* (ATT) of test vehicles as the measure to evaluate the effectiveness of our approach. We compare our approach against two baseline methods: Fastest Path (FP) and Real-Time (RT) based methods. Both methods use Dijkstra's algorithm to compute routes and use travel time as edge weight, but they differ in the way they estimate the travel time. In the FP method, the travel time is calculated based on the maximum allowed speed of the road. This is a basic approach and does not get traffic updates. The RT method estimates and updates graph weights every five minutes based on speed samples collected during the past five minutes. Note that the FP approach represents the worst case scenario, since the weight is static and is not affected by traffic. Also, to test whether blocking the event location is enough to re-route from the event, we use a variation of FP approach (FP2) that is aware of the speed drop at the event location and re-route vehicles.

Simulation scenario: The simulation scenario is designed to test the effectiveness of a given routing method in helping drivers avoid traffic events. In this simulation scenario, the traffic event is modeled using a set of parameters such as (location, impact and duration). We create a set of 100 test vehicles whose initial routes go through the event location, and use a given routing method to re-route those vehicles. Then, we measure the simulated travel time of test vehicles. We fix simulation time to one hour and the event always starts at minute 5 and lasts for 30 minutes. During the simulation, the traffic event will be activated on a selected road, and this will reduce the road speed according to the given event impact parameter. The speed drop caused by the traffic event is visible to all routing methods (except for FP) and when test vehicles are released into the road network, they will be re-routed using the routing method under test (FP, FP2 RT, SPA). During the simulation, the travel times for vehicles are recorded and exported. We run this scenario for the four routing methods under test, and use the travel time of test vehicles to measure and compare the methods. In this scenario, the simulation was performed on Melbourne CBD, with the event starting at minute 5, test vehicles are released after 8 minutes, the event impact will reduces the road speed by 90% and the number of vehicles is set to 1000 vehicles during peak time.

Results: The result is shown in Figure 2. The figure shows that FP2 has reduced the ATT to 761 seconds, which means blocking the event location helps a part of the test vehicles but not all of them as the high ATT indicates that some vehicles still experienced a significant delay. Similarly, the RT method has ATT close to FP2 (725 seconds), which shows that using the RT method has a small improvement over blocking the event location. Furthermore, the figure shows that our method reduced ATT significantly to 391 seconds, which means our method reduced the travel time by 47% on average when compared to the RT method. The RT method did not perform well in this experiment because the propagation of traffic event was faster than the update rate of the RT method. This leads the congestion to spread to alternative routes used by the RT method and blocked the vehicles re-routed via these alternative routes.

5 CONCLUSION

We presented a navigation system with a traffic simulator in its core to help drivers avoid traffic events effectively. The system can run faster than real-time to forecast the future traffic conditions for traffic events. We presented an extensive experimental evaluation to study the speed and accuracy of the simulator as well as the quality of the navigation advice. The speed experiments shows that the simulator can run 50 times faster than real-time which means the simulator takes less than 1 minute to predict 30 minutes into the future for 1000 vehicles. In addition, the simulator can achieve an average accuracy of 77.5% when compared with real-data from TomTom traffic API for common city travel scenarios. Furthermore, we tested the effectiveness of the navigation advice using the simulator with a realistic scenario. Our method was able to suggest alternative routes that avoid the evolution of the traffic event and achieved up to 47.5% reduction in travel time on average. For this experiment, we assume our estimate of traffic conditions and event impact is accurate. In the future, we will study the effectiveness our methods when this estimate degrades. Also, we will focus on enhancing the flow estimation component to achieve better accuracy.

REFERENCES

[1] Abdullah AlDwyish, Egemen Tanin, and Shanika Karunasekera. 2015. Location-based Social Networking for Obtaining Personalised Driving Advice. In *SIGSPATIAL'15*. ACM, 91:1–91:4.
[2] Abdullah Aldwyish, Egemen Tanin, and Shanika Karunasekera. 2017. Follow The Best: Crowdsourced Automated Travel Advice. In *MobiQuitous'17*. ACM.
[3] John C. Falcocchio and Herbert S. Levinson. 2015. *Managing Nonrecurring Congestion*. Springer International Publishing, 197–211.
[4] Daniel Krajzewicz, Georg Hertkorn, Christian Rössel, and Peter Wagner. 2002. SUMO (Simulation of Urban MObility)-an open-source traffic simulation. In *MESM20002*. 183–187.
[5] Kai Nagel and Marcus Rickert. 2001. Parallel implementation of the TRANSIMS micro-simulation. *Parallel Comput.* 27, 12 (2001), 1611–1639.
[6] US Department of Transportation. 2017. Reducing Non-Recurring Congestion. (2017). https://ops.fhwa.dot.gov/program_areas/reduce-non-cong.htm [Online; accessed 25-May-2017].
[7] Bei Pan, Ugur Demiryurek, Chetan Gupta, and Cyrus Shahabi. 2015. Forecasting spatiotemporal impact of traffic incidents for next-generation navigation systems. *KAIS* 45, 1 (2015), 75–104.
[8] Kotagiri Ramamohanarao, Hairuo Xie, Lars Kulik, Shanika Karunasekera, Egemen Tanin, Rui Zhang, and Eman Bin Khunayn. 2016. SMARTS: Scalable Microscopic Adaptive Road Traffic Simulator. *ACM TIST* 8, 2 (2016), 26.
[9] Rashid A Waraich, David Charypar, Michael Balmer, and Kay W Axhausen. 2015. Performance improvements for large-scale traffic simulation in MATSim. In *Computational Approaches for Urban Environments*. Springer, 211–233.