A Prioritized Routing Assistant for Flow of Traffic

by

Gaurav Gupta, Kamishetty Sreeja, Praveen Paruchuri

in

The 21st IEEE International Conference on Intelligent Transportation Systems
(*ITSC-2018*)

Maui, Hawaii, USA

Report No: IIIT/TR/2018/-1

Centre for Visual Information Technology
International Institute of Information Technology
Hyderabad - 500 032, INDIA
November 2018

A Prioritized Routing Assistant for Flow of Traffic

Gaurav Gupta, Sreeja Kamishetty and Praveen Paruchuri[1]

***Abstract*— Traffic management during emergency evacuation throws up a different set of challenges than a regular traffic management. We focus here on one particular challenge, namely prioritized routing. Prioritized routing may need to happen even during normal times but stands out during emergencies, since emergency vehicles, police vehicles and vehicles (such as buses) that carry a lot more people may need to have a higher priority in terms of evacuation. It is also reasonable to assume that traffic police may need to perform a centralized control of traffic since they typically have a global view of the emergency and possibly have accurate real-time updates. We therefore make the following contributions in this paper: (a) We map the prioritized routing problem to the minimum-cost maximum-flow problem, a standard problem formulation in network flow theory. (b) We then develop the Prioritized Routing Assistant for Flow of Traffic (PRAFT) that casts the prioritized routing problem which includes a notion of priority of vehicles and priority of routes into the minimum cost maximum flow problem. (c) Through a series of experiments performed using the well-known traffic simulator SUMO, we could establish that PRAFT indeed maps higher priority vehicles to better quality routes and is monotonic in the sense that decreasing priority order of vehicles maps to a decreasing route quality.**

I. INTRODUCTION

Management of traffic during emergency evacuation [1], [2] presents a different set of challenges as compared to a regular traffic management. Given that the traffic police would typically have a better idea of the nature of the emergency and potentially have accurate (or reliable) real-time updates, the police may need to perform a centralized control of traffic in such situations. During a typical evacuation some vehicles would need to be evacuated faster than others e.g., emergency vehicles or large vehicles carrying a lot more people [3]. Many times, for an emergency vehicle on road, saving time in the order of minutes or even seconds can make a life saving difference for certain patients [4]. Hence, the notion of priority becomes very useful. Apart from emergency vehicles, we would also need to evacuate a number of other vehicles in a prioritized fashion where priority refers to priority of routes (i.e., path composed of roads and intersections in the road network) as well as priority of vehicles. Priority of route can be a function of user specified factors while priority of vehicle can be an assignment based on the domain requirements.

To account for all these factors, we develop the **Prioritized Routing Assistant for Flow of Traffic (PRAFT)** that: (a) Enables the police to maximize the traffic flow while (b) Providing the ability to account for priorities of vehicles and routes while performing the routing. PRAFT maps the prioritized routing problem to the minimum-cost maximum-flow Problem (MCMFP), a standard problem formulation in network flow theory. A solution to MCMFP preserves the maximum flow property of the unconstrained problem (i.e. max flow problem without the cost constraints) while routing the traffic based on priority of routes and vehicles. Hence it creates distinct traffic patterns as compared to a solution that does not include priorities (i.e., a max flow solution computed using Ford Fulkerson Algorithm (FFA)). In particular, PRAFT maps higher priority vehicles to better quality routes and is monotonic in the sense that decreasing priority order of vehicles maps to a decreasing route quality.

Traffic simulations facilitate the evaluation of infrastructure and policy changes before implementation on roads [5]. Simulation of Urban Mobility (SUMO) is one such free and popular traffic simulator [6], [7] which is used here. We use it in combination with OpenStreetMap (OSM) [8] which is a open source tool to model a real-world map and traffic settings. In particular, we use the traffic models provided by OSM for New York city and import into SUMO for simulation purposes . We then use PRAFT to identify the routes that the different vehicles should take and let SUMO handle the low level dynamics of how the vehicle should traverse along the given route [7] (e.g., pick lanes within a road, acceleration, deceleration, overtake etc). Note that we make no changes to the dynamics that SUMO decides apart from prescribing the route a vehicle should take. We then show through experimentation that our assistant is able to route the high priority vehicles in a better fashion compared to FFA.

II. FORD FULKERSON ALGORITHM (FFA)

The **Ford-Fulkerson Algorithm (FFA)** [9], [10] is a popular algorithm to compute maximum flow in a flow network be it water flow [11], liquid flow or flow of traffic [12] i.e., to solve the **Maximum Flow Problem (MFP)**. [13] uses FFA with modifications to model speed dependent capacities to study the Bangkok traffic. [14], [15] propose that the police use FFA to maximize flow of traffic for evacuation purposes. However, FFA does not have a notion of priorities for roads and vehicles, hence cannot handle the issue of prioritized routing. The following steps are involved in FFA:

- As long as we have a **augmenting path** (identified using the Breadth First Search (BFS) or Depth First Search (DFS)) possible in the graph from source to sink, we

[1]Gaurav Gupta, Sreeja Kamishetty and Praveen Paruchuri are with Machine Learning Lab, Kohli Center on Intelligent Systems, IIIT Hyderabad, Gachibowli, Hyderabad, 500032, India gaurav.gupta@research.iiit.ac.in, kamishetty.sreeja@research.iiit.ac.in, praveen.p@iiit.ac.in

identify the bottleneck value (i.e., edge with minimum capacity on the path) of this path in the graph.
- We then subtract the bottleneck value from the currently available capacity of each edge on the augmenting path to obtain the **residual graph** (which is updated every time a path is found) . We then iterate over the steps using the newly obtained residual graph till no path is possible from source to sink.

Algorithm 1 which we use from [16], [17] presents the steps of the Ford Fulkerson Algorithm. Table I provides description for terminology used in the algorithm. Note that the term pf does not have a residual graph version/definition, since pf does not change. As mentioned earlier, the limitation of using FFA for a realistic traffic network is that, we get the maximum capacity of the network by using the number of lanes in roads as the capacity matrix (since capacity of road directly corresponds to the number of lanes). Usage of other parameters for the capacity matrix may not result in maximum flow, although they may be optimizing on some other scale. This limits us to use number of lanes as the value in capacity matrix, but would be unable to choose or decide routes based on other parameters like speed limit, average time taken etc. Another issue is that the routes we obtain using FFA are from the usage of BFS/DFS which assume that all the routes have a priority of one i.e., cannot encode a priority value for routes.

TABLE I
NOTATION TABLE FOR FFA **AND PRIORITIZED ROUTING ASSISTANT**

Notation	**Description**
G(V,E)	Graph with V vertices and E edges
G_f*(V,E)*	Residual Graph with V vertices and E edges
c(U,V)	Capacity of the edge from U to V
c_f*(U,V)*	Capacity of the edge from U to V in the residual graph
f(U,V)	Flow of edge from U to V
s	Source node
t	Sink node
pf	**Priority function**

Algorithm 1: Ford Fulkerson Algorithm

<u>function FFA</u>;
Input : Given a Network $G = (V, E)$ with flow capacity c, a source node s , and a sink node t
Output: Compute a flow f from s to t of maximum value
$f(u, v) \leftarrow 0$ for all edges (u, v)
While there is a path p from s to t in G_f (computed using BFS or DFS), such that $c_f(u, v) > 0$ for all edges $(u, v) \in p$:
  Find $c_f(p)$ = $min\{c_f(u, v) : (u, v) \in p\}$
  For each edge $(u, v) \in p$:
    $f(u, v) \leftarrow f(u, v)$ + $c_f(p)$ (send flow along the path)
    $f(v, u) \leftarrow f(v, u)$ - $c_f(p)$ (the flow might be returned later)
  End For
End While

III. PRIORITIZED ROUTING ASSISTANT FOR FLOW OF TRAFFIC (PRAFT)

PRAFT works on a directed graph having a single source and a single sink similar to FFA (with intersections modeled as nodes and number of lanes in a road between two intersections as weight of the edge (road). More details below in section "Road Network".). PRAFT aims to maximize the flow (i.e., maximize the number of vehicles going from source to sink) while assigning different routes for vehicles based on their priority i.e., higher priority vehicles get assigned to higher quality routes. Note that quality of route is encoded using a priority function (pf in Algorithm 2). The priority function itself is based on underlying factors related to traffic flow e.g., length of road, estimated time to traverse a route (which is sum of expected times to traverse all the roads and intersections along that route) and other such factors. For a factor such as the expected time to reach destination, an ascending order over the expected time for different routes would be good i.e., lower expected time is better. However, for purposes of generality across factors we assume that our priority function will be built such that a descending order of priority values is good. To achieve this: (a) PRAFT maps the prioritized routing problem to the MCMFP and identifies the maximum priority maximum flow solution (b) PRAFT then uses the priority of vehicles to construct a dispatching strategy that routes higher priority vehicles along higher priority routes.

While both PRAFT and FFA work on a directed graph, PRAFT also needs a priority function which can be developed from the input description as follows: We use the length of the road/edge in a network and the speed limit of a road (since speed limit is a good representation of the average speed a road has) and compute (*distance/speed*) as our priority function where a lower value for this expression implies a higher priority. The following significant characteristics would make PRAFT useful in practice: (a) We explicitly consider priority of vehicles and routes which would be useful during emergency evacuation and possibly even for normal traffic. (b) The notion of priority we capture need not always be measurable by time. It can include fuel consumption, speed limits, type of road (highway, city road), length of route, scenic value of route, pollution related and many others. (c) The priority function we encode for each edge is an average value (can be exact value if needed e.g., length of road). For example, if priority happens to be time to traverse an edge i.e., if an edge needs say t_{ij} units of time for flow to traverse from node i to node j, it would mean that the average traversal time across all vehicles on that edge is t_{ij} and not individual times.

A. Algorithm for MCMFP

The **Minimum-cost flow problem (MCFP)** is a fundamental optimization problem in network flow theory which aims to find the lowest cost/cheapest possible way of sending a flow through a flow network. This problem was first solved by Edmonds and Karp [18], their algorithm is now commonly known as *Edmonds-Karp scaling technique*. Multiple different algorithmic approaches to improve the run time have since been made for MCMP [19] and [20]. The **Minimum-cost Maximum-flow problem (MCMFP)** is a variation of the MCMP to find the maximum flow, but have the lowest cost among all the maximum flow solutions. There are different algorithms such as the **Cycle Cancelling Algorithm** [21] and the **Hungarian Algorithm** [22] which help to solve MCMFP.

We now present details of the algorithm for MCMFP implemented for our purposes. MCMFP algorithm will have two different matrices for the same network: (a) Capacity Matrix (for number of lanes, i.e., width of the road) and (b) Priority function matrix (where priority is a user defined function as mentioned earlier). Since both the matrices are for the same graph their dimensions will be same. Note that the capacity matrix used for MCMFP is the same as for FFA, hence MCMFP will compute the same maximum flow as provided by FFA. Please refer Algorithm 2 for detailed steps. A key step in FFA is to identify the augmenting path in each iteration, which is obtained using BFS (or DFS). However, both BFS and DFS use an edge cost of one, which implies each step is weighed the same. To enable usage of different edge cost at each step which allows mapping of priority function to the edge costs (e.g., expected time to traverse a route), MCMFP algorithm uses the Dijkstra's Algorithm [23]. We make a modification here wherein we use the UCS (Uniform Cost Search) algorithm which is more known in the AI community and has less memory requirements among other small advantages (detailed set of differences are listed in [24]). Since UCS is guided by path cost (and hence our priority function) rather than path length, at each step of UCS, the next node n to be expanded is the one whose cost $g(n)$ is lowest, where $g(n)$ is the sum of the edge costs from the root to the node n.

Algorithm 2: Algorithm for Minimum Cost Maximum Flow Problem

function MCMFP;
Input : Given a network *G=(V, E)* with flow capacity c and a priority function *pf*, with source node s and sink node t.
Output: Compute the flow f from s to t to get the maximum flow and obtain paths in order of priority
$f(u,v) \leftarrow 0$ for all edges (u,v)
While there is a path p from s to t in G_f (computed using UCS), such that $c_f(u,v) > 0$ for all edges $(u,v) \in p$ and $\sum pf(u,v)$ for all edges $(u,v) \in p$ is highest across all possible paths (ensured by UCS):
  Find $c_f(p) = min\{c_f(u,v) : (u,v) \in p\}$
  For each edge $(u,v) \in p$:
    $f(u,v) \leftarrow f(u,v)$ + $c_f(p)$ (send flow along the path)
    $f(v,u) \leftarrow f(v,u)$ - $c_f(p)$ (the flow might be returned later)
  End For
End While

B. PRAFT Dispatching strategy

We encode two types of priorities: (a) Priority of edges modeled as an abstract priority function *pf* (in Algorithm 2). pf can be specified by user and can encode factors like time, fuel-consumption etc. as mentioned earlier. (b) Priority of vehicles which can depend on our requirements e.g., if higher priority vehicles receive faster routes, ambulances can be given highest priority, buses next and so on. *pf* can be the basis for modeling priority of vehicles. Dispatching strategy refers to the implementation of FFA or PRAFT solution in traffic, both of which will identify the same maximum flow (by design). We therefore dispatch vehicles in waves, where each wave has the maximum flow number of vehicles (unless lesser number of vehicles present to be dispatched). Two consecutive waves will be released with a gap of $flow\_wave\_time$ number of seconds to ensure smooth traffic flow. In FFA, paths are picked in increasing order of path length or depth (by BFS). Vehicles are then mapped randomly to the routes/paths that are picked. However in PRAFT, as the paths are generated they are picked in increasing order of path costs (by UCS) i.e., decreasing order of priority. We then map higher priority vehicles to higher priority paths.

IV. Illustrative Example

Example Setup: Figure 1 shows a road network (directed graph) with 4 vertices $\{0, 1, 2, 3\}$ and 4 edges (or roads) 0→1, 0→2, 1→2 and 2→3. There are two values associated with each edge/road. The first value marked on an edge represents the number of lanes in that edge i.e., capacity of the edge. The second marked value represents the priority of that edge.

Comparison of FFA and PRAFT: FFA uses BFS, hence picks paths in increasing order of length (or depth). In our example, path I (0→2→3) and then Path II (0→1→2→3) will be picked in that order. This implies maximum flow will be 4 units: path I filled with its maximum capacity of 2 and then 2 units on path II [Priority graph will not be used by FFA]. PRAFT uses both capacity and priority values to compute routes and picks paths in increasing order of costs: Path II (0→1→2→3) and then Path I (0→2→3). Maximum flow identified will still be 4 (3 units on Path II and 1 on Path I), since we identify flow using capacity graph which is same for both. However, when deciding routes based on edge priorities, we see that Path II has a cost of 4 (sum of edge costs 1, 1, 2 for edges 0→1, 1→2, 2→3) while the cost for Path I is 5 (sum of edge costs 3, 2 for edges 0→2, 2→3). Hence, Path II gets picked and filled by traffic first before Path I.

Dispatching strategy: As mentioned earlier, vehicles will be released in waves and each wave will have vehicles $\leq$ the maximum flow (4 in our example). In wave 1 if we have 4 vehicles $\{v1, v2, v3, v4\}$ generated with priority $\{4, 2, 3, 1\}$ [Higher number implies higher priority], FFA assigns vehicles to routes randomly. In PRAFT, we assign vehicles in the order of routes identified (Path 1 and then Path 2). So $v1$, $v2$ will be assigned to Path 1 and $v3$, $v4$ to Path

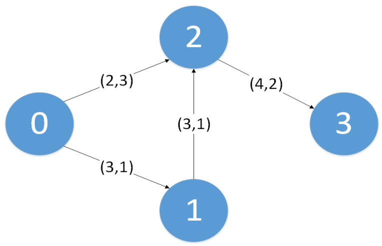

Fig. 1. Directed graph containing capacity/priority of network

2. PRAFT maps priority of vehicles to priority of paths in descending order: will assign $v1$, $v3$ and $v2$ to Path II and $v4$ to Path I. Here, PRAFT will result in either lower travel time or lower fuel consumption across all the vehicles depending on what we model. Let's say only 3 vehicles $\{v1, v2, v3\}$ are present in wave 2 with priorities $\{2, 3, 3\}$ [$v2$ and $v3$ are same vehicle type and assigned same priority]. FFA assigns $v1$, $v2$ to Path I and $v3$ to Path II. PRAFT assigns all 3 vehicles to Path II and can significantly save on time/fuel compared to Path I.

V. Experimental Setup

A. Road Network

For experiments, we use the Open Street Maps (OSM) to obtain an OSM file which can be exported to SUMO using NETCONVERT (a tool provided by SUMO). In particular, we first exported the New York, USA map into SUMO . Figure 2 shows two circles: inner circle represents the area which we have to evacuate (i.e., a radius of 250 meters from Empire State Building) and outer circle represents the minimum distance we have to send vehicles from the point of emergency (i.e 800 meters) to complete evacuation. We assume that no vehicle starts between 250 to 800 meters. The image is then converted into a network where roads intersected by the inner circle are starting points (sources) and intersected by the outer circle are end points (sinks). Multiple sources/sinks are converted into a single source/sink by connecting all of them to a single virtual source/sink node with no capacity limitations. The police agent would then provide routes for each vehicle, that enables maximum flow in the network. We let SUMO handle all the vehicle dynamics (e.g., speed, acceleration, interaction with other vehicles like overtaking etc.). Each wave of vehicles is released every ($flow\_wave\_time$) set to 5 seconds in our simulation, so we give enough time for vehicles to cover a safe distance before the next wave enters. Note that a route here specifies only the path to take from source to sink node but does not identify the specific lane to take, speed to travel and other dynamics that SUMO specifies.

B. SUMO Parameters

As described in the section "Road Network", we import the road network map from OSM to SUMO. For experimentation purposes, we retain the key road characteristics present in the map including type of road (like a highway, dirt road, inner city road and such) and its related parameters like speed limit, type of vehicles allowed, number of lanes and such with others. We assume traffic at the start of experimentation to be negligible and traffic lights are not being used in the simulation. The characteristics of each vehicle is shown in Table II and all the vehicles in our experiments follow these parameters unless specified otherwise. A total of 7600 vehicles were modeled in each of our experiments.

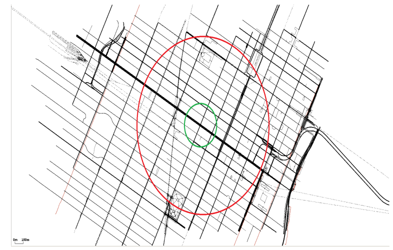

Fig. 2. Evacuate from inside the inner circle to outside the outer circle

TABLE II
VEHICLE CHARACTERISTIC AND ITS VALUE IN SIMULATION

Characteristic of vehicle	Value
Maximum speed	$90kph$
Acceleration	$5m/s^2$
Deceleration	$10m/s^2$
Minimum gap between vehicles	2.5 m
Length of vehicle	5 m

VI. Experiments

All experiments were performed on the New York city map.

A. Validity of Flow Rate provided by FFA

In our first experiment, we aim to validate the claim that using FFA gives us the best flow rate (i.e., number of vehicles that exit every ($flow\_wave\_time$ =) 5 seconds). The flow rate obtained using FFA is 19, hence we release 19 cars every 5 seconds and test against other flow rates (i.e., we use the same number of routes but decrease/increase the flow rate). The total number of routes here is 13 in the max flow solution (note that total number of routes from source to sink is higher since only a subset can concurrently support to achieve the maximum flow). Figure 3 shows the flow rate i.e., number of vehicles sent out per wave on x-axis and the total time taken for the simulation in seconds on y-axis. Please note that even though the flow rate is changed, the routes being used are the ones prescribed by FFA. For the different flow-rates mentioned on x-axis, we see a drastic increase in total time w.r.t. to the optimal flow (of 19 vehicles per wave). This is on expected lines, since the number of vehicles that should flow per route is determined by the bottle-neck of number of lanes in each route. This experiment holds true for PRAFT as well since as mentioned earlier it has the same maximum flow as FFA.

B. Priority based Routing: Average Statistics

We now examine the effect of modeling priorities on routing of vehicles. In particular, we modeled 4 priority

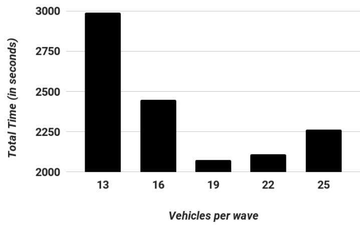

Fig. 3. Total time for evacuation based on flow rate

classes/categories for vehicles. We then find routes using PRAFT, with priority for the route modeled as estimated time for completion (i.e., distance/speed limit). Table III shows the priority type and the count of vehicles present in the simulation with that priority. When PRAFT is ran on the New York Map, it identifies 10 different routes from the start node to evacuate the vehicles. The algorithm then distributes the vehicles in the following fashion: $2-3-3-2$ i.e., first 2 routes with highest priority (i.e., lowest estimated travel time) get assigned for vehicles with highest priority (i.e., Type 1), next 3 routes with highest priority for Type 2 vehicles and so on.

In this experiment, we first computed the time needed to evacuate each individual vehicle in seconds using FFA and PRAFT. We then performed a paired-sample t-test on the evacuation time results of FFA and PRAFT. The test was performed on 7600 observations (i.e., vehicles simulated) for each algorithm and we obtained a two tailed p value $< .0001$, which shows that the difference between the two sets of observations is highly statistically significant. These results show that there are significant inherent differences between the outputs of FFA and PRAFT. The results also showed that: (a) The mean (μ) evacuation time for FFA is 42.95 seconds while for PRAFT is 34.09 seconds. This implies that it needs each vehicle about 26.3% higher time to reach from source to sink with the path prescribed by FFA as compared to PRAFT. (b) The standard deviation (σ) value for FFA is 13.51 while for PRAFT is 6.07. The mean and standard deviation values show that not only PRAFT prescribes better routes on an average, it also picks them in a fashion that decreases the variance between routes in terms of time needed to evacuate (since routes get picked in decreasing order of priority by PRAFT as opposed to having no particular pattern in picking of routes by FFA since it does not optimize on priority values).

TABLE III
COUNT OF VEHICLES FOR EACH TYPE OF PRIORITY

Priority Type	Type 0	Type 1	Type 2	Type 3
Count	1200	3600	2000	800

C. Priority based Routing: Individual Characteristics

As described in previous experiment, we modeled 4 different priorities for vehicles and solved using FFA and PRAFT. While the previous section focuses on average statistics, we focus here on individual vehicle behaviors for the same experiment. In that sense, no new experiment is run here, but we perform a different analysis for the same results. In particular, we present scatter plots in figures 4, 5 where x-axis represents the time at which a vehicle enters the simulation and y-axis represents the time it takes for the vehicle to exit the simulation. All the 7600 vehicles that were part of the simulation are represented in the scatter plot. We also have different colors for vehicles with different priorities. For example, we use red for Type 1 (highest priority) vehicles, blue for Type 2 and so on as indicated in the figures.

Figure 4 shows the simulation time on x-axis from 0 to 2000 seconds and time taken by each vehicle to exit the simulation in seconds on y-axis. A total of 7600 points are represented in the figure, one per vehicle. Given that 19 vehicles are released per wave (i.e., every 5 seconds), we have $(7600/19) * 5 = 2000$ seconds for all the vehicles to be released. The figure shows that PRAFT gives a scatter plot which is neatly distributed in terms of priorities. The reason here is that higher priority vehicles are explicitly assigned higher priority routes for PRAFT i.e., routes with shorter time to finish evacuation while the ones with lower priority are assigned routes which take longer to evacuate. Hence the reason we see a clear distribution where Type 1 vehicles need shortest time to the exit simulation, Type 2 next and so on.

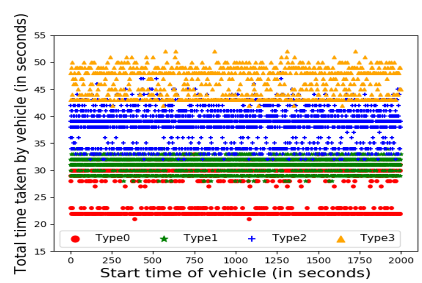

Fig. 4. Scatter plot for PRAFT

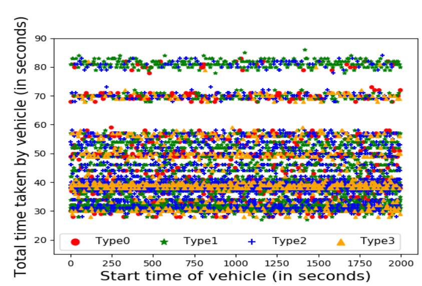

Fig. 5. Scatter plot for FFA with Priority Labels

Figure 5 shows the scatter plot using FFA. FFA does not explicitly consider the priority of vehicles while assigning routes, hence we see a plot where the vehicle types are not neatly clustered as in Figure 4. The scattering is because the algorithm does not differentiate between higher and lower priority vehicles. Hence any of them is as likely to be assigned a path that takes shorter or longer to evacuate as long as the overall flow is maximized. Please note that the number of vehicles for each priority remains the same across both the figures, as mentioned in Table III. To conclude, PRAFT not only assigns a faster route for high priority vehicles but also has a predictable trend in the evacuation time for vehicles of each priority category.

D. Diversity of Parameter Values for Vehicle Types

In all our previous experiments, we used the same set of values for parameters such as maximum speed, acceleration etc. (see section "SUMO Parameters") for all the vehicles. However in real-life, vehicles will not have the same parameters. In this experiment, we use a different set of parameters for the different vehicle types as presented in Table IV. For experimentation purposes, we assign higher priority to slower moving vehicles and compute the average time taken by each vehicle type using FFA and PRAFT shown in Table V. From the table we observe that the average evacuation time in seconds for an individual vehicle (i.e., average time needed to traverse from source to sink), is better when we use PRAFT compared to FFA. This is because FFA does not have a concept of priority for routes and vehicles get sent randomly on the different routes identified while PRAFT prescribes better routes for higher priority vehicles. PRAFT would therefore be pretty useful to traffic police and/or policy makers since they would have the option to redirect certain types of vehicles to specific routes by tuning the priorities.

TABLE IV
DIFFERENT PARAMETER VALUES FOR DIFFERENT VEHICLES

Vehicle Type	Bus	Truck	Car	Motorcycle
Acceleration (m/s^2)	1.5	3	5	6
Deceleration (m/s^2)	4	4	7.5	10
Max speed (km/hr)	85	130	180	200

TABLE V
AVERAGE EVACUATION TIME USING FFA AND PRAFT

Vehicle Type	Bus	Truck	Car	Motorcycle
PRAFT	38.9	31.13	29.95	39.07
FFA	52.33	47.2	44.02	43.9

VII. CONCLUSIONS

In this paper, we develop a traffic assistant named PRAFT that performs a priority based routing while maximizing the flow of traffic which was not handled in prior works. We performed a series of experiments using the well-known traffic simulator SUMO and could show that PRAFT indeed maps higher priority vehicles to better quality routes while ensuring maximum traffic flow. PRAFT can provide a way for policy makers to take routing decisions conditioned on the vehicle type. This work lays the basis to pursue a number of directions in future: (a) Identification of realistic priority functions and results obtained by them. (b) Encoding of individual preferences in priority function to tailor routes better e.g., a low priority car short on gas maybe sent on a high priority route or a car with an individual preference to travel North maybe accommodated if the information is known beforehand etc.

REFERENCES

[1] A. J. Pel, M. C. J. Bliemer, and S. P. Hoogendoorn, "A review on travel behaviour modelling in dynamic traffic simulation models for evacuations," *Transportation*, vol. 39, no. 1, pp. 97–123, 2012.

[2] E. Kwon and S. Pitt, "Evaluation of emergency evacuation strategies for downtown event traffic using a dynamic network model," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1922, pp. 149–155, 2005.

[3] A. Jotshi, Q. Gong, and R. Batta, "Dispatching and routing of emergency vehicles in disaster mitigation using data fusion," *Socio-Economic Planning Sciences*, vol. 43, no. 1, pp. 1–24, 2009.

[4] T. Telegraph, "Figures expose ambulance delays in life and death calls." http://www.telegraph.co.uk/news/health/news/10962298/Figures-expose-ambulance-delays-in-life-and-death-calls.html, 2015.

[5] A. L. Bazzan, "A distributed approach for coordination of traffic signal agents," *Autonomous Agents and Multi-Agent Systems*, vol. 10, no. 1, pp. 131–164, 2005.

[6] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility," in *The Third International Conference on Advances in System Simulation (SIMUL), Barcelona, Spain*, 2011.

[7] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, pp. 128–138, December 2012.

[8] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.

[9] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian journal of Mathematics*, vol. 8, no. 3, pp. 399–404, 1956.

[10] L. R. Ford Jr and D. R. Fulkerson, *Flows in networks*. Princeton university press, 2015.

[11] J. M. Drake and D. M. Lodge, "Global hot spots of biological invasions: evaluating options for ballast–water management," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 271, no. 1539, pp. 575–580, 2004.

[12] A. Schrijver, "On the history of the transportation and maximum flow problems," *Mathematical Programming*, vol. 91, no. 3, pp. 437–445, 2002.

[13] E. J. Moore, W. Kichainukon, and U. Phalavonk, "Maximum flow in road networks with speed-dependent capacities-application to bangkok traffic.," *Songklanakarin Journal of Science & Technology*, vol. 35, no. 4, 2013.

[14] T. J. Cova and J. P. Johnson, "A network flow model for lane-based evacuation routing," *Transportation research part A: Policy and Practice*, vol. 37, no. 7, pp. 579–604, 2003.

[15] G. Gupta and P. Paruchuri, "Effect of human behavior on traffic patterns during an emergency," in *19th IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2052–2058, 2016.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Section 26.2: The ford-fulkerson method," *Introduction to algorithms*, pp. 651–664, 2001.

[17] Wikipedia contributors, "FordâĂŞfulkerson algorithm — Wikipedia, the free encyclopedia," 2018. [Online; accessed 27-April-2018].

[18] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM (JACM)*, vol. 19, no. 2, pp. 248–264, 1972.

[19] R. K. Ahuja, J. B. Orlin, and R. E. Tarjan, "Improved time bounds for the maximum flow problem," *SIAM Journal on Computing*, vol. 18, no. 5, pp. 939–954, 1989.

[20] J. B. Orlin, "A faster strongly polynomial minimum cost flow algorithm," *Operations research*, vol. 41, no. 2, pp. 338–350, 1993.

[21] A. V. Goldberg and R. E. Tarjan, "Finding minimum-cost circulations by canceling negative cycles," *Journal of the ACM (JACM)*, vol. 36, no. 4, pp. 873–886, 1989.

[22] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics (NRL)*, vol. 2, no. 1-2, pp. 83–97, 1955.

[23] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[24] A. Felner, "Position paper: Dijkstra's algorithm versus uniform cost search or a case against dijkstra's algorithm," in *Fourth Annual Symposium on Combinatorial Search*, 2011.