A Matlab-based Scheduling Simulator for Deterministic Real-time Communications in ITS

Sikandar Zulqarnain Khan, Daniel Dinis, Renato Lopes, Paulo Pedreiras, Joaquim Ferreira

Instituto de Telecomunicações
Universidade de Aveiro, Portugal

Abstract—**This work presents a MATLAB-based discrete event simulator designed for the purpose of analysing the performance of VFTT-protocol based scheduling algorithms. This simulator is capable of performing complex tests in order to ascertain the applicability of V-FTT protocol in an infrastructure-based wireless vehicular network. The simulator provides its outputs in the form of text and graphs showing the efficiency of different output variables in an infrastructure based Intelligent Transportation Systems (ITS). A simulation example illustrates the use of the software presented.**

Keywords— **VFTT, ITS, STDMA, SZ, RSU, OBU**

I. Introduction

Due to the increasing processing power of computers in the last few decades, mathematical packages such as MATLAB/SIMULINK are considered good options for performing simulation based analysis. MATLAB being a strong platform for solving complex mathematical operations accompanied by powerful features of Simulink provide a user-friendly environment for modelling real world scenarios. The GUI interface of MATLAB/SIMULINK allows for the fast configuration of inputs and outputs modelling various cases of same scenarios.

The work presented here describes the development of a Matlab-based software tool designed specifically for analysing the performance of scheduling algorithms based on V-FTT protocol. This simulation tool contains models for an infrastructure-based ITS; allowing for changes in its inputs; making it is possible to compare different road case scenarios. Some simulation results are also presented to illustrate the operation of the proposed software environment.

Before discussing the details of this GUI simulator, a brief overview of the V-FTT protocol and bandjacking technique is given in Section A. Section B discusses the justification for a need to develop this simulator. Section C discusses the user's interface and the inputs of the simulator. The outputs generated are discussed in Section D. A V-FTT scheduling algorithm is also briefly discussed before concluding the work in the end.

A. *Vehicular-Flexible Time Triggered Protocol*

Recently, a proposal for deterministic medium access control (MAC) for vehicular environment called the "Vehicular Flexible Time-Triggered (V-FTT) Protocol" was presented in [1]. The Vehicular Flexible Time-Triggered Protocol (V-FTT) is a multi-master multi-slave protocol that embodies timeliness properties and admission control mechanism for deterministic medium control. The operation of the V-FTT protocol is supported by a bandjacking technique [2] which is discussed briefly in section B. The V-FTT protocol adopts spatial time division multiple access (STDMA) approach for scheduling the transmission of on-board units (OBUs) such that the road-side units (RSUs) act as masters and OBUs act as slaves [3].

The V-FTT protocol divides the time into consecutive elementary cycles (ECs), each one composed of a collision-free phase and contention-based phase. The Collision-free phase is composed of an Infrastructure Window (IW) and Synchronous OBU Window (SOW) whereas contention-based phase is composed of Free Period (FP), as shown in Fig.1.

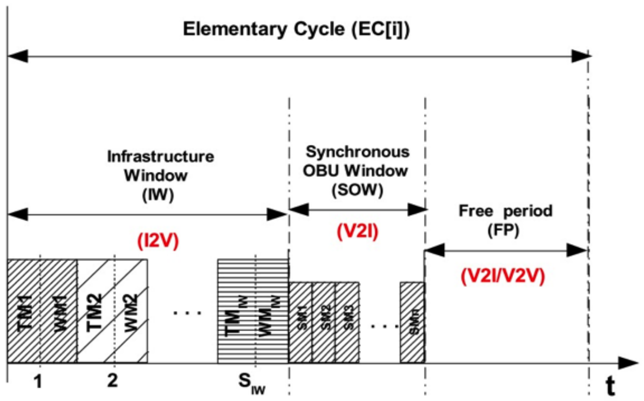


Fig.1: Elementary Cycle of Vehicular Flexible Time-Triggered protocol.

The V-FTT protocol inherits most of its concepts from the original Flexible Time-Triggered protocol definition [4]; while adding some new features to cope with the safety related communications in wireless vehicular networks. In particular it adopts redundant scheduling for OBUs transmissions to increase reliability and to cope with the variations of the propagation patterns of the radio links, caused by atmospheric and traffic conditions. According to the proposed redundant scheduling scheme, a single OBU is scheduled by a configurable number of RSUs for the same transmission slot [3]. As RSUs cooperate to schedule OBUs safety communications, they must be able to coordinate their own transmissions, avoiding any possible mutual interference. To support RSU coordination, it is assumed that they are fully interconnected by a backhauling network. It is also assumed that RSUs are able to receive messages from vehicles travelling in both directions and vehicles can receive messages from various adjacent RSUs [3].

Each RSU transmits its Trigger Message (TM) in its particular transmission slot (in the IW); to schedule the OBUs transmission slots (in SOW), using just one message. This scheme is known as master multi-slave; as a single master (RSU) message triggers the transmission of a number of slave (OBU) message as opposed to the traditional master-slave in which each master message triggers just one slave reply. As a configurable number of RSUs cooperate to redundantly schedule the transmissions of the same OBU, it can be said that V-FTT adopts a multi-master multi-slave spatial TDMA. In this RSU coordination proposal, RSUs transmit the OBUs scheduling in a reserved window called the Infrastructure Window. Within this window time slots are reserved for each RSU. As RSUs are synchronized, they are able to respect the time slot boundaries.

The infrastructure window is followed by the synchronous OBU window where OBUs have the opportunity to transmit information to RSUs (V2I communication). Each OBU has a fixed size slot to transmit vehicle's Information (i-e. speed, acceleration, heading, etc.) and/or a safety event. The Synchronous OBU Window (SOW) duration is variable as it depends on the number of vehicles in the region of an RSU. The elementary cycle ends in a free period (FP). During free period, the OBUs transmit non-safety messages whereas the non-VFTT enabled OBUs can transmit safety messages.

In V-FTT protocol, the roadside units are responsible for two main functions: (i) to schedule the transmission instants of the OBUs in what concerns the safety frames they have to broadcast (ii) to receive information from the OBUs, edit that information and publish the edited safety information in the adequate places and instants (might be a broadcast or might be a communication to selected vehicles).

From the communications point of view, the OBUs must listen to the RSU transmissions (at least one RSU should be heard), retrieve the safety information and dispatch this information forward.

Constant size Cooperative Awareness Messages (CAMs) are transmitted during the Synchronous OBU Window (SOW). CAM messages are broadcast messages that include several possible data elements (e.g., Crash Status, Dimension, Heading, Latitude, Longitude, Elevation, Longitudinal Acceleration and Speed). CAM messages are transmitted periodically and have strict timing requirements. They are generated by the CAM Management and passed to lower layers according to some set of rules which are checked every 100ms [5]. A CAM message is dropped whenever the channel access request does not result in actual channel access before the next message is generated. There is temporary reduction in the performance efficiency of the application if a periodic message misses its time limit.

Non-registered OBUs also receives safety information from RSUs. However, they are not able to transmit information according to the proposed protocol, although they can still contend for transmission during the free period, but without any guarantees.

The information broadcasted by the RSUs must be trustworthy, that is why they must validate the OBUs' events before broadcasting them. This validation must obviously be performed in bounded time so that the results could be transmitted to the OBUs in real-time [5].

Road segments covered by RSUs running the V-FTT protocol are called Safety Zones (SZ). Whenever a vehicle enters a safety zone, it registers itself with the infrastructure and it is assigned a temporary identifier. This temporary identifier is used by the RSUs to schedule its CAM transmissions in the SOW. The responsibility of scheduling vehicles moving along a road equipped with a roadside infrastructure is passed from RSU to RSU in a cooperative and distributed way. This handover process is also dependable and timely.

The deployment of safety wireless vehicular communications in the scope of ITS applications supported by roadside backhauling networks requires an end-to-end deterministic behaviour. For example, a vehicle involved in an accident should be granted timely access to the wireless medium to transmit a safety message, which once validated by the roadside infrastructure, should trigger the timely transmission of warning messages to other vehicles approaching the accident site. Therefore, it is important to have a proper scheduling of the communication channel to allow critical information to be transmitted with minimum latency. Moreover, the vehicles close to the accident or driving in its direction could be within the coverage area of different RSUs; each enforcing a real-time MAC protocol for the vehicles in their coverage area. In this scenario the responsibility of scheduling the vehicle transmissions is passed to nearest RSU; thus requiring deterministic handover to extend the local (RSU) real-time guarantees to the whole roadside infrastructure [5].

The requirement of a real-time MAC protocol to allow a privileged access to the medium (channel); even in congested environments; motivated the development of the bandjacking technique [2][6]. After realizing that bandjacking is an effective technique to ensure medium access in open environments crowded with contention-based communications, it was devised that it could bring the features offered by the V-FTT paradigm to the wireless domain. So in this way this mechanism is capable of exploiting a contention-based medium by providing a real-time deterministic communications in vehicular environments. In a bandjacking technique; the critical data transmission is preceded by a fixed length black-burst transmission [7]; that ultimately jams all the ongoing communications. If the jamming signal is long enough, all stations will eventually find the medium busy and postpone (back-off) their transmissions to a later time. While on the other hand the critical station will transmit its data packet immediately after the black-burst sequence and hence will gain prioritized access to the medium compared to non-critical stations respecting the IFS or performing Clear Channel Assessments (CCA) before initiating their transmission. Further details of the bandjacking technique and black burst sequence can be found in [6]] and [7].

B. The need for development of new scheduling Simulator

A number of software packages like SUMO [8], ns-3 [9] and iTETRIS [10] are available as open sources and could have been used for analysing the performance of V-FTT protocol based scheduling algorithm. However ,developing a scheduling algorithm for an infrastructure based ITS running the V-FTT protocol could have been much time consuming and arduous. That is why we opted for the development of a new simple-to-use software package; designed specifically for our peculiar infrastructure based ITS; to simulate the traffic patterns in the scope of V-FTT protocol. The main goal of our simulator is to integrate simple traffic scenarios with V-FTT based scheduling; without explicitly simulating the radio channels as these software packages would require. It should be noted that with V-FTT based redundant scheduling; in which each OBU is scheduled by 2 or 3 RSUs; the probability of an OBU not to receive a trigger message is very low. Radio channel level simulation of the V-FTT protocol will be important to assess the effectiveness of the bandjacking mechanism. We plan to integrate SUMO at a later stage so our scheduling simulator could be fed with more realistic traffic patterns. The simulator integrates wireless communications and road traffic platforms in an environment that could easily be tailored to scenarios allowing performance analysis of V-FTT protocol. This simulator is discussed in detail in the following section.

C. The user interface and inputs of the Vehicular-Flexible Time Triggered Matlab-based Simulator

The user interface of the V-FTT protocol based Matlab simulator is shown in Fig.2 with the default inputs as required for simulation. The user is required to choose certain parameters from three input sections i-e inputs related to the “Motorway (SZ)”, the “V-FTT protocol” and the “Simulator settings”. From “Motorway inputs” section, a user can choose variables such as lane width, vehicle length, vehicles spacing and the RSU coverage range (all units in meters); depending on the specific scenario that a user wishes to feed into the simulator. The “V-FTT input” section enables a user to select the V-FTT settings e-g the bandjacking time, size of the elementary cycle (EC) and modulation type. The “Simulator settings” section allows a user to choose whether the results produced be in graphical format or in the form of a textual file called “Log File”. The user also has the option of running this simulator in a non-random fashion with a list of already knows inputs. In that case the simulator picks up the data from an input file called “event_list.txt”; which is also discussed later in this section.

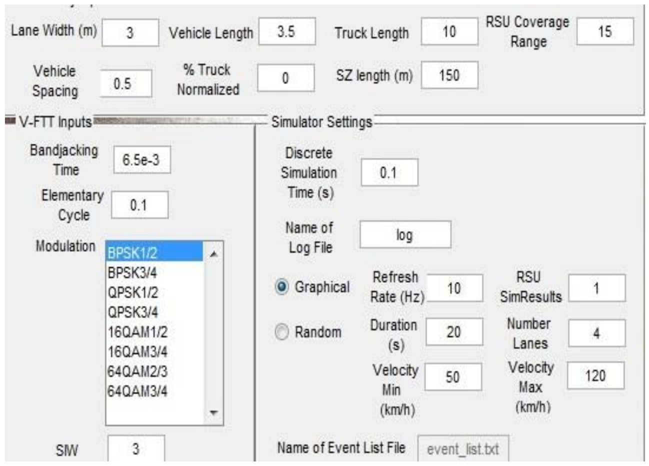


Fig.2: User Interface of the Vehicular Flexible Time-Triggered Simulator (with default Inputs).

All the input parameters related to the V-FTT protocol, the safety zone and the simulation settings are used to initialize the auxiliary structures and statistical counters defined inside the simulator. Based on the values of these input parameters and the correlation existing among them; as defined by the mathematical expressions inside the simulator; the outputs are generated. These outputs can either be in form of a graph or a text file as opted by the user before the start of the simulation. The flowchart in Fig.3 shows the flow of events taking place inside simulator after the simulation starts.

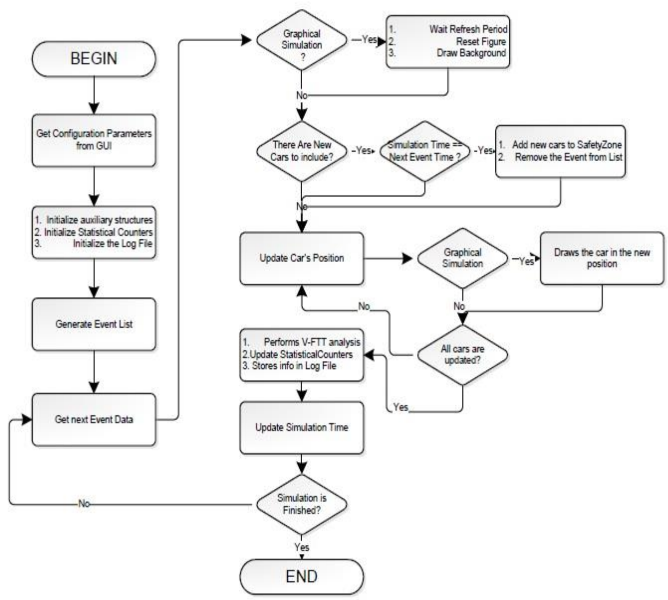


Fig.3: Flowchart of events of Matlab-based V-FTT Simulator.

Fig.3 shows that when simulation starts; all the auxiliary variables and statistical counters are initialised according to the input (configurable) parameters. Based on the values of these input parameters an event list is generated which specifies the time of entry of each OBU into the Safety Zone. The simulator then checks whether the user has opted for an output in a graphical format or in the form of a file. In former case, the simulator draws the background with all the RSUs installed on road sides, the number of lanes of the motorway and the number of OBUs entering at discrete times. If the graphical mode is not chosen; the simulator goes directly to perform the simulation without drawing the whole scenario in GUI format. The simulator then starts picking up OBUs from the event list and adding them up to the safety zone according to their entry times. While adding an OBU into the SZ; it is removed from the event list at the same time as it has already been served. With the updating of the simulation clock, the positions of cars in the SZ are updated and the V-FTT analysis performed. The results generated are stored in temporary variables to produce outputs in the form of textual file "logfile.txt" and graphs. The simulation time is constantly checked for its finish time before which the simulation continues to generate results at discrete times.

D. *Output details of the V-FTT Matlab-based Simulator*

After briefly discussing the user interface, architecture and input parameters of the V-FTT protocol simulator, this section discusses the outputs generated by the simulator. Fig.4 shows the graphical output of the simulator for the default input parameters as shown in Fig.2. The SZ is taken to be 150m with a 3m width for each lane as default values. The bandjacking value is 6.5 ms, the EC duration is equal to 100 ms and the modulation type is BPSK1/2.

When the simulation starts a screen as shown in Fig.4 is shown. The screen shows the graphical view of the traffic situation as entered by the user through the input parameters.

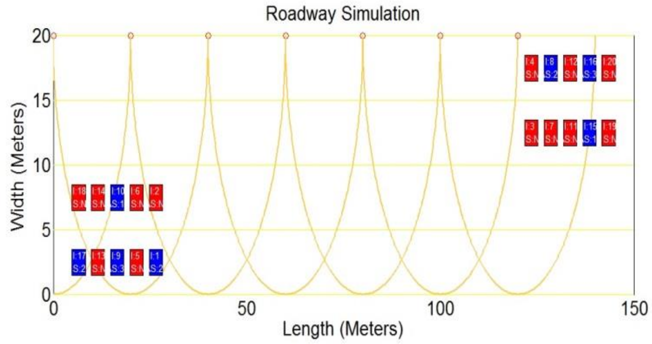


Fig.4: Graphical Output of Vehicular Flexible Time-Triggered protocol

The parallel (yellow) lines represent the lanes of the motorway whereas the parabolic (yellow) circles denote the geographic regions covered by RSUs (SZ). The motorway (by default) is a two-way four lines road where nodes travel in two directions. The moving squares represent OBUs or nodes going from RSU to RSU across the SZ in both directions. While travelling across the safety zone, these mobile nodes can be seen changing their colours. The blue coloured nodes are the ones that have been allocated slots in the SOW of the current EC; whereas the red coloured nodes are the OBUs that have not been allocated slots in the SOW of the current EC. As nodes move on, they change their colour from red to blue as they get slots in the next EC. The moving nodes also give information about the temporary IDs and the slot number allotted to it by each RSU. As the simulation continues, all these information are updated whenever a significant event occurs.

Fig.5, Fig.6 and Fig.7 shows the result of simulations for different SIW values where the rest of the inputs parameters are taken to be the same. SIW is the number of slots to be used in the RSU Infrastructure Window (one slot per RSU) corresponding to the maximum number of simultaneous RSU transmissions that an OBU can listen to.

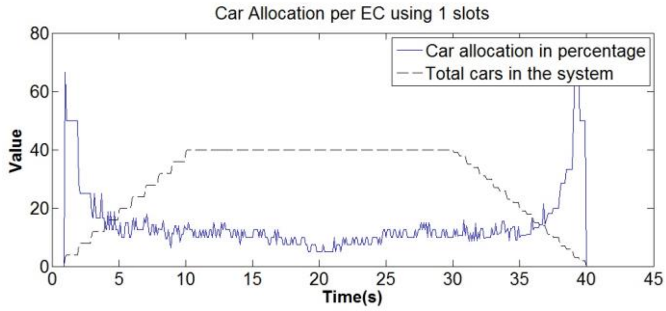


Fig.5: Percentage of allocated slots with SIW=1.

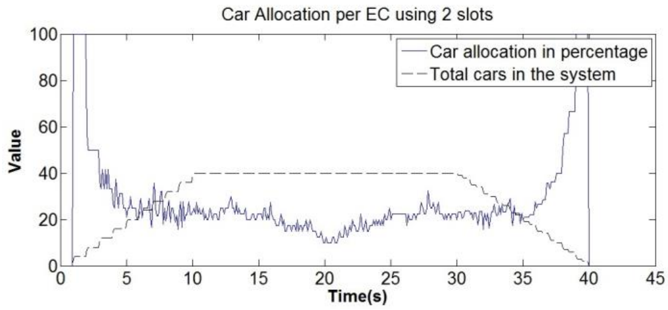


Fig.6: Percentage of allocated slots with SIW=2.

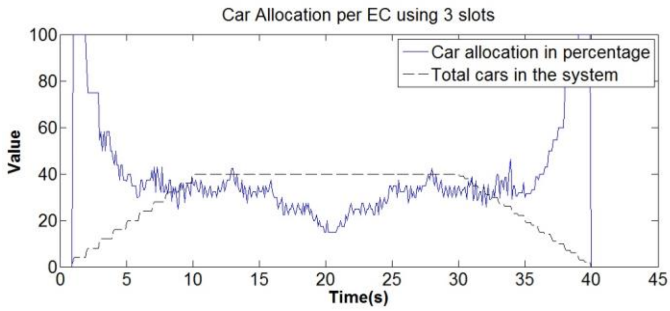
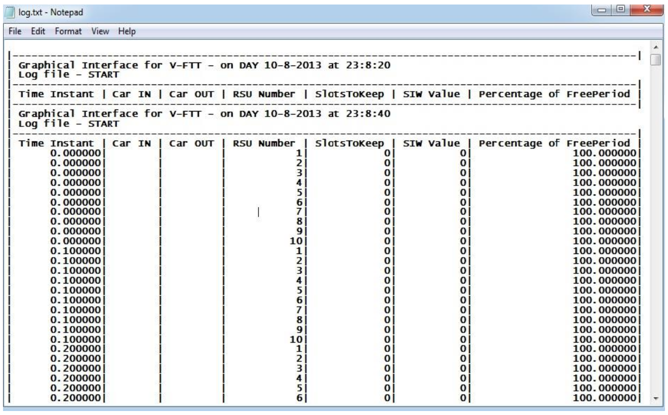


Fig.7: Percentage of allocated slots with SIW=3.

The graphs in Fig. 5-7 shows that when congestion is low (which happens during the start of simulation) the efficiency of slot allocation is high as almost all the nodes (OBUs) are allotted slots by the V-FTT scheduler. However, as the number of nodes in the SZ increases, the efficiency of slot allocation falls accordingly. Similarly the efficiency is high for a high SIW whereas the efficiency is low for a low SIW as depicted in the graphs above.

The general format of the output file “Log file”, generated at the end of each simulation is shown in Fig.6. This file contains several columns each representing a different set of information. The first column shows discrete time instants at which simulation is performed. Column2 and Column3 show instants at which OBUs enter/leave the safety zone. Column4 shows the RSU number where as Column5 is the maximum number of slots is the SOW that each RSU can allocate. Last column shows the percentage of Free Period (FP) which was discussed earlier.



```
log.txt - Notepad
File Edit Format View Help
|-----------------------------------------------------------------------------------------------|
| Graphical Interface for V-FTT - on DAY 10-8-2013 at 23:8:20
| Log file - START
|-----------------------------------------------------------------------------------------------|
| Time Instant | Car IN | Car OUT | RSU Number | SlotsToKeep | SIW Value | Percentage of FreePeriod |
|-----------------------------------------------------------------------------------------------|
| Graphical Interface for V-FTT - on DAY 10-8-2013 at 23:8:40
| Log file - START
|-----------------------------------------------------------------------------------------------|
| Time Instant | Car IN | Car OUT | RSU Number | SlotsToKeep | SIW Value | Percentage of FreePeriod |
|     0.000000|       |         |           1|           0|         0|              100.000000|
|     0.000000|       |         |           2|           0|         0|              100.000000|
|     0.000000|       |         |           3|           0|         0|              100.000000|
|     0.000000|       |         |           4|           0|         0|              100.000000|
|     0.000000|       |         |           5|           0|         0|              100.000000|
|     0.000000|       |         |           6|           0|         0|              100.000000|
|     0.000000|       |         |           7|           0|         0|              100.000000|
|     0.000000|       |         |           8|           0|         0|              100.000000|
|     0.000000|       |         |           9|           0|         0|              100.000000|
|     0.000000|       |         |          10|           0|         0|              100.000000|
|     0.100000|       |         |           1|           0|         0|              100.000000|
|     0.100000|       |         |           2|           0|         0|              100.000000|
|     0.100000|       |         |           3|           0|         0|              100.000000|
|     0.100000|       |         |           4|           0|         0|              100.000000|
|     0.100000|       |         |           5|           0|         0|              100.000000|
|     0.100000|       |         |           6|           0|         0|              100.000000|
|     0.100000|       |         |           7|           0|         0|              100.000000|
|     0.100000|       |         |           8|           0|         0|              100.000000|
|     0.100000|       |         |           9|           0|         0|              100.000000|
|     0.100000|       |         |          10|           0|         0|              100.000000|
|     0.200000|       |         |           1|           0|         0|              100.000000|
|     0.200000|       |         |           2|           0|         0|              100.000000|
|     0.200000|       |         |           3|           0|         0|              100.000000|
|     0.200000|       |         |           4|           0|         0|              100.000000|
|     0.200000|       |         |           5|           0|         0|              100.000000|
|     0.200000|       |         |           6|           0|         0|              100.000000|
```

Fig.6: Format of the output file as generated by the simulator.

Before concluding the paper a brief overview of the VFTT-scheduling algorithm is given in section II.

II. SCHEDULING ALGORITHM AND SIMULATION RESULTS

The V-FTT based scheduling algorithm, schedule the transmissions of synchronous OBUs messages (CAMs) in a road traffic scenario similar to the one presented in [3]. The V-FTT protocol guarantees deterministic real-time communications, as the scheduling algorithm based on it, provides a conflict-free transmission of OBUs safety messages in a wireless vehicular network. A simpler protocol interference model is also adopted to account for the vehicles that are within the coverage range of more than one RSU. In such interference-based scenario the responsibility of scheduling an OBU’s transmissions is passed to the nearest RSU; requiring a deterministic handover from one RSU to its adjacent next [3]. A simpler node assignment scheme is followed to derive a conflict-free schedule; for all the nodes within the geographic range of each RSU. A slot-reuse mechanism is also exploited to allow for more OBUs to be served; ultimately resulting in an increase in the efficiency of scheduling algorithm [3].

The output of this algorithm is a matrix (Sched matrix) that contains the set of message IDs corresponding to CAM messages; that shall be scheduled by each RSU in the EC to follow. The algorithm executes concurrently in all RSUs; scheduling the transmissions of OBUs that are within the range of that particular RSU at that particular instant. The scheduling algorithm executes in the fashion as shown in Fig.7. Further details of the algorithm are beyond the scope of this paper and can be found at [3].

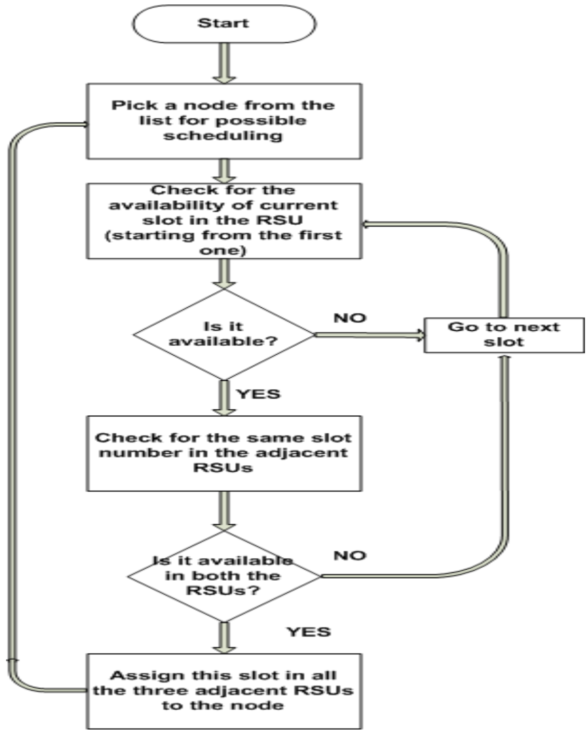


Fig.7: Flow of the VFTT based scheduling algorithm.

III. CONCLUSIONS

In this paper we presented a MATLAB-based discrete event simulator; designed specifically for analysing the performance of a V-FTT protocol based scheduling algorithm. The algorithm schedules the transmissions of OBUs’ CAMs; travelling in an infrastructure based wireless vehicular network; thus guarantying a deterministic real-time communication. A brief overview of the V-FTT protocol and bandjacking technique was discussed before discussing the architecture and I/Os of the simulator. A simulation example was also illustrated based on the default inputs of this simulator. The scheduling algorithm was also presented very briefly. The outputs of this simulator; both in textual file format and graphical format were also discussed.

IV. ACKNOWLEDGEMENT

The work presented here is supported by European Union seventh Framework Program (FP7) under the project titled “Intelligent Cooperative Sensing for Improved Traffic Efficiency (ICSI)”.

REFERENCES

[1] I. F. 317671, “Design and performance evaluation of a real-time MAC for IEEE 802.11p - D3.1.1: Fundamental Design Decisions of the Deterministic MAC Protocol," http://www.ict-icsi.eu/deliverables.html, online, as in June 20 2014, 2013.

[2] P. Bartolomeu and J. A. Fonseca, "Channel Capture in Noisy Wireless Contention-Based Communication Environments," in Proc. of the 8th IEEE Int. Workshop on Factory Communication Systems, France, May 2010.

[3] Sikandar Zulqarnain Khan, Paulo Pedreiras and Joaquim Ferreira, "Improved Real-Time Communication Infrastructure for ITS", in INForum Sept; 2014.

[4] L. Almeida, P. Pedreiras, and J. Fonseca, "The ftt-can protocol: why and how," Industrial Electronics, IEEE Transactions on, vol. 49, no. 6, pp. 1189-1201, Dec 2002.

[5] ETSI, "Final draft ETSI ES 202 663 V1.1.0 (2009-11), ETSI Standard, Intelligent Transport Systems (ITS); European profile standard for the physical and medium access control layer of Intelligent Transport Systems operating in the 5 GHz frequency band," November 2011.

[6] P. Bartolomeu, "Dependable Wireless Real-Time Communications for Open Environments", PhD Thesis, University of Aveiro, 2014.

[7] J. Sobrinho and A. Krishnakumar, "EQuB-Ethernet Quality of Service Using BlackBursts," Local Computer Networks, 1998. LCN '98. Proceedings, 23rd Annual Conference on, pp. 286–296, Oct 1998.

[8] http://sumo-sim.org/

[9] http://www.nsnam.org/

[10] http://www.ict-itetris.eu/