Implementation and Simulation of Handover Techniques to Guarantee Service Continuity through Microservices at Edge

Luigi Bitonti and Mauro Tropea
Dimes Department, University of Calabria, Via P. Bucci 39/c, 87036 Rende (CS), Italy

Keywords: Automotive, Edge Computing, Collision Avoidance.

Abstract: Edge computing is a mesh network of micro data centers, capable of processing and storing critical data locally, and of transmitting all data received and / or processed to a central data center or to a cloud storage repository. With this paradigm, data, applications and computational services are brought from the cloud servers to the margins of the network. To divide their computational power, virtual hosts are distributed within a single physical host through virtualization. One of the virtualization techniques that is spreading is containerized programming. In this paper, the concepts just mentioned will be applied to a use case, to ensuring continuity of service to vehicles traveling on a fast-moving road and which are managed by Edge technology. Two handover algorithms supported by microservices at Edge are proposed and performance assessed.

1 INTRODUCTION

A cellular telecommunication network is a communication network that has the last link wireless and is distributed geographically in areas called *cells*, each served by at least one fixed-location transceiver called *Base Transceiver Station (BTS)*. The division of territory into cells allows to guarantee the service to a large number of users with a single BTS that operates the frequency reuse. When a mobile terminal is on the edge of a cell, it starts a control mechanism in order to check a signal with a greater power by a different BTS. If the mobile terminal finds a new BTS with a stronger power, it performs a procedure called handover (Salihin et al., 2018). There are also four basic techniques for handover: 1) Received signal strength (RSS); 2) RSS with threshold; 3) RSS with hysteresis; 4) RSS with hysteresis and threshold. Based on the handover metrics mentioned above, the decision about how and when to switch the interface to which network will be made.

Edge computing (Khan et al., 2019) has gained considerable popularity in academic and industrial sector in the last few years. It serves as a key enabler for many future technologies like 5G, Internet of Things (IoT) (Shah and Yaqoob, 2016), augmented reality and vehicle-to-vehicle communications by connecting cloud computing facilities and services to the end users. And, a particular attention has been addressed to security aspects of these technologies (Santamaria et al., 2019; Santamaria et al., 2018). The Edge computing paradigm provides low latency, mobility, and location awareness support to delay-sensitive applications. The cell permanence time and the mobile analysis for predictive services are important aspects in wireless networks and they are object of study by research communities, such as it possible to view in (De Rango et al., 2008), (Fazio et al., 2017), (Frnda et al., 2013).

In this work, we have considered as scenario a Vehicular Ad-Hoc Network (VANET) (Ghori et al., 2018) where the mobile device supports simultaneous connection to multiple access points. Some studies show the importance of prediction in handover mechanism for guaranteeing QoS such as (Fazio et al., 2016). During the handover process, there is no interruption of the service, given that the device remains connected to at least one access point. Very studied aspects for the ad-hoc network are routing issues as it is possible to view in (Zhou et al., 2006), (Fotino et al., 2007), (Socievole et al., 2011), and also for VANET is an important topic as it is possible to view in (De Rango et al., 2009), (Fazio et al., 2013). Moreover, in the considered scenario the support of edge computing (Pan and McElhannon, 2017; Khan et al., 2019) has been considered in order to permit the communication in the vehicular environment taking into account the latency, that can play an important role for avoiding vehicles accidents. In this work, as edge device has been chosen the cellular BTS that is placed at

Bitonti, L. and Tropea, M.
Implementation and Simulation of Handover Techniques to Guarantee Service Continuity through Microservices at Edge.
DOI: 10.5220/0009971002870293
In *Proceedings of the 10th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2020)*, pages 287-293
ISBN: 978-989-758-444-2

roadside and then it can assist the vehicles that moving in the road. The BTSs long the road are numerous and then, they can collaborate each other in order to guarantee service continuity to the mobile devices, avoiding any form of interruption. In this work, a solution has been proposed to guarantee the continuity of the service in an automotive environment managed with Edge Computing technology (Klas, 2017). This technology was chosen because of its low latency and high reliability properties in respect to Cloud architecture (Pan and McElhannon, 2017). Docker containers were used to develop the servers to make applications light and (Doan et al.,). In this way, if the servers do not need to use all their resources to satisfy the requests, they could also be used by other clients. The servers have been grouped in a Cluster, to be managed efficiently. Kubernetes was used to orchestrate the Cluster containers and to make the system robust (Bachiega et al., 2018). To ensure the continuity of the service, *two handover algorithms* have been implemented. The first algorithm is based on the *threshold RSS technique*, the second one adds *Cell Breathing Mechanism*. The proposed mechanisms have been implemented in Java, taking advantage of libraries and tools adapted to the context.

The rest of this paper is organized as follows: Section 2 presents the proposal of guaranteeing service continuity to vehicles on the road managed by Edge technology. In Section 3, a description of the simulator and the implementation details are given. In Section 4, we describe the numerical results. Finally, Section 5 concludes the paper.

2 PROPOSED SOLUTION

The roadside BTSs communicate with each others thanks to the transport infrastructure backbone. By taking advantage of 5G networks, latencies are reduced to about 15 ms (Erel-Özçevik and Canberk, 2019). In this work, a smart BTS equipped with a processing unit has been considered able to perform data elaboration and then, to make decisions analyzing obtained results on the basis of the proposal algorithm. In this way, the BTS is also able to evaluate if it has to propagate messages arriving from previous BTSs towards the next BTSs or to stop forwarding because it is not necessary. To apply this behavior the following considerations can be made: all vehicles registered with the BTS are tracked; if the last vehicle needs to be notified, then the message must be propagated to the adjacent BTS; vice versa, if the last vehicle is at a distance from the event such that the warning is not necessary, then it is useless to forward the message to the adjacent BTS, because surely the vehicles under its range will be far enough, and then it is not necessary the use of the edge.

To implement handover processes it was necessary to create two servers that work in parallel, able to simulate two different BTSs. This was applied considering three virtual machines able to form the Kubernetes cluster: one machine has been used as a master; other two as worker nodes. The virtual machines were hosted on a normal PC, as it is possible to view in Figure 1.

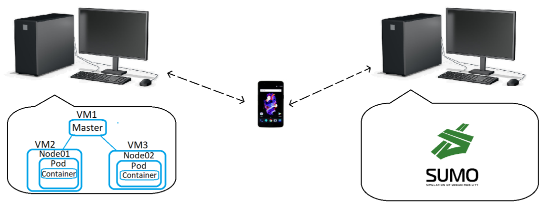

Figure 1: Architectural Scheme.

An urban mobility simulator, SUMO, was used to generate data from vehicles (Lim et al., 2017). SUMO is an open source software that allows to create various traffic scenarios. This simulator was used to have the most realistic data possible. It is also possible to interface the simulator with the Java programming language, thus making it easier to collect vehicle data at runtime. The simulator runs on an other PC different to the one hosting *containers*, as it is possible to view in Figure 1.

For the sake of simplicity, it is assumed that all cars travel in one direction. To facilitate the simulation and make it faster, the two servers are supposed to be at a distance of 100 meters from each other and their coverage range is 90 meters. In this way, it is possible to know what happens during handover since the vehicles remain in the area covered by both nodes for a sufficiently long time, see Figure 2.

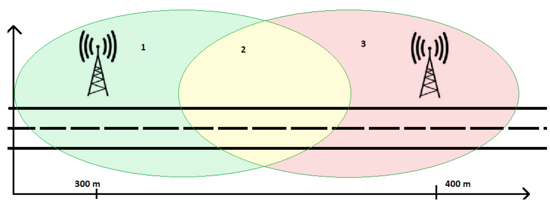

Figure 2: Considered Test Environment.

In Figure 2, it is possible to note three different zones: 1) zone covered by first server; 2) zone covered where a mobile device can be covered by both first and second server, here takes place the handover mechanism; 3) zone covered by only second server.

2.1 Docker and Kubernetes

Docker is an open source platform that allows to develop and to run applications (doc, 2019). With Docker it is possible to keep applications separate from the hosting infrastructures, to make software distribution possible quickly and easily using a structure called container. Kubernetes is a portable, extensible and open source platform useful for managing workloads and services developed with containers (kub, 2019). An important concept in Kubernetes is the cluster. It is a set of machines hosting containerized applications, where at least a machine is a *master node* (controlling and managing operations) and another one is a *worker node* (run the master instructions). The smallest unit that constitutes a Kubernetes application is represented by the Pod, that represents the processes running on the cluster.

2.2 Vehicle Traffic Simulator

Through the *TraaS* library it was possible to interface SUMO with Java (sum, 2019), (tra, 2019), (Artal-Villa et al., 2019). Eclipse was used as a development environment. For vehicle configuration and simulation, *XML* language files have been written. For each vehicle, the IDentifier (ID) number, speed and position are taken. These parameters are formatted according to a specific code, so as to be made legible by the receivers. When the vehicle is in an area covered by both BTSs, the vehicle sends the message to both, thus simulating the mechanism of macro-diversity. Macro-diversity is a mechanism introduced in UMTS networks through which it is possible to be connected simultaneously to more than one cell (Gesbert et al., 2010). In any case, although the vehicle is connected to both stations, only one is used to warn it with warning messages. The choice on which the two stations should have the control is made by the stations themselves and the algorithm, shown later, will explain it. When the vehicle no longer receives the signal from a station, it communicates this condition to the one whose signal it is currently receiving, in order to warn it of the new condition. In this way, a *soft handover* was carried out: the vehicle passes under control of a new station without remaining always connected to the network. The continuity of the service is thus guaranteed. The Constrained Application Protocol (CoAP) was used to communicate with the servers.

3 SOFTWARE IMPLEMENTATION

The servers were implemented in Java language and use the CoAP protocol (cal, 2019) to communicate with the vehicles or other servers. For the management of handover, two paradigms were used: a *RSS with threshold* technique and a *combination of RSS with threshold and cell breathing*. An explanation of the type of messages that can be generated by the system and a description of both algorithms will now be given. Messages can be classified into five different typologies:

- **Type 0:** it is the most used type of message. It is a periodic message that each vehicle sends to its reference server to update it on its speed and position;
- **Type 1:** this message is sent from one server to another upon it is detected a particular event;
- **Type 2:** it is sent from the vehicle to the server for notifying handover end. The vehicle switches from the old server to the new one;
- **Type 3:** it is a message that arises from the receipt of a type 2 message. It is used to notify to the old server that a vehicle is disassociated from it (the old server removes it from its database);
- **Type 4:** it is the only message used in the second handover implementation for communicating to a server that its coverage range is decreasing.

Thanks to the exchange of these messages the soft handover can be used and then a vehicle can pass from a coverage area served by a BTS to the new area served by another BTS able to manage the mobile device. In the following the two approaches, with a RSS threshold and with cell breathing mechanism are described and compared in order to show the behavior of the proposal algorithms.

3.1 BTS with RSS Threshold Mechanism

In this scenario, the server continually receives messages from vehicles that are under its coverage. To be initialized, the server must know the location/position of the neighbor server, its IP address, the port on which to send the data and the range of coverage. Taking into consideration Figure 2, it is assumed that if the vehicle is inside zone 2, it is in a location where it can receive a signal from the radio base station 1 whose power is above of the acceptable threshold. In zone 3, the signals from server 1 can no longer be processed. The mechanism adopted for performing

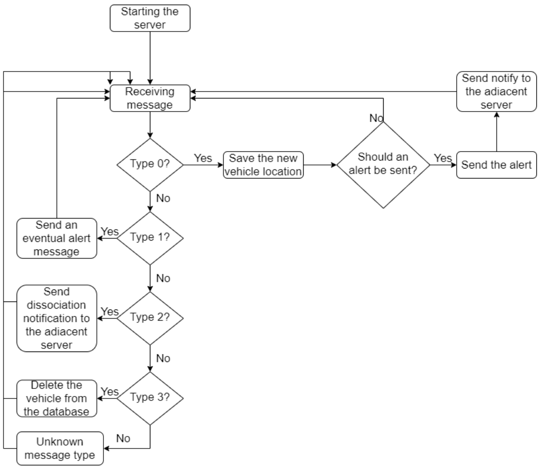

Figure 3: Flowchart of a server with RSS control mechanism.

handover is based on RSS technique, a mechanism well known in cellular networks (Tinkhede and Ingole, 2014). When a vehicle receives a *Type 0* message, the vehicle's position is saved. In this way the server is aware of the presence of this vehicle under its coverage. When a vehicle is in an alert condition, such as a sudden braking, the server will notify all the vehicles under its coverage and in the vicinity of the vehicle that caused the alert generation. In the case of server 2, it will send a notification only to vehicles in zone 3. Server 1, on the other hand, takes care of vehicles in zone 1 and zone 2. Server 2 also sends a *Type 1* message to server 1 to inform it that a vehicle under its coverage has made an emergency stop. When a *Type 1* message is received, the server must check whether the vehicles under its control are near the vehicle that has braked sharply. If so, it forwards a message to these vehicles suggesting them of braking. In this way, thanks to the collaboration between the two servers, it was possible to avoid potential accidents. Upon receiving a *Type 2* message, the server becomes aware that the vehicle is under its control, since it is in a position such that the signal strength of the server 1 is below a certain threshold. Therefore, it sends a *Type 3* message to the adjacent server, to inform of this new condition. The message contains the ID of the vehicle under handover process. When a server receives a *Type 3* message, it is informed that a vehicle is no longer under its management. It can therefore proceed with the removal of the data relating to it, not more necessary. Hereafter, the server will no longer exchange messages with this vehicle. The main issue of this implementation regards no load balancing functionality. In Figure 3, the flow diagram of this mechanism is depicted.

3.2 Server with Cell Breathing Technique

In this section an improvement on the servers functionality is presented in order to enhance the handover implementation in the system modifying the behavior of the BTSs in the intersected coverage area. The proposed improvement is based on the cell breathing concept (Thng et al., 2005), mechanism in use in UMTS networks. The idea is to perform a sort of load balancing into the BTS stations decreasing the number of vehicles to manage in a BTS with the consequent increasing into the cell managed by the other BTS. In the following, some details about the mechanism implementation in the considered system will be explained. In order to consider the new mechanism, three new data structures have been introduced. The first, called *ZonaComune vehicles*, keeps track of the number of vehicles present in zone 2, i.e. in the area where potentially both servers could manage them. The second one, *blackList*, indicates those vehicles which, although still in the coverage area, they no longer need to be managed, as they have been assigned to the next server. The third one, *VehiclesToControllare*, indicates the vehicles that have been assigned to the new server. When a server receives a message from a vehicle, it verifies that its ID is not one to be ignored. If so, the message is discarded without further analysis, since the vehicle is under the management of another server. Otherwise, it analyzes the position of the vehicle and it controls if the vehicle can potentially be transferred to another server by comparing its position with the position of the adjacent server and its range of coverage. This is, the server becomes aware of the area where the vehicle is located. If the vehicle is in zone 2, it is added to the data structure called *VehiclesZoneComune*. After having updated the list, the server checks if it has exceeded the maximum number of connections that it can accept regarding the mobile devices belonging to zone 2. If the limit has been exceeded, it virtually decreases its range of action. This means that it gives control of the first vehicle of the queue, effectively relieving its computational load. To ensure that the vehicle is supervised by the new server, it is necessary to inform this server on the new condition. The information is sent using the *Type 4* message. When a server receives a *Type 4* message, it takes the vehicle ID contained in that message and it enters this ID in the appropriate data structure. In this way, the control of the vehicle goes from one server to the other. So, the handover was carried out using the Cell Breathing paradigm. In Figure 4, the flow diagram of the server that acts with cell breathing paradigm is shown.

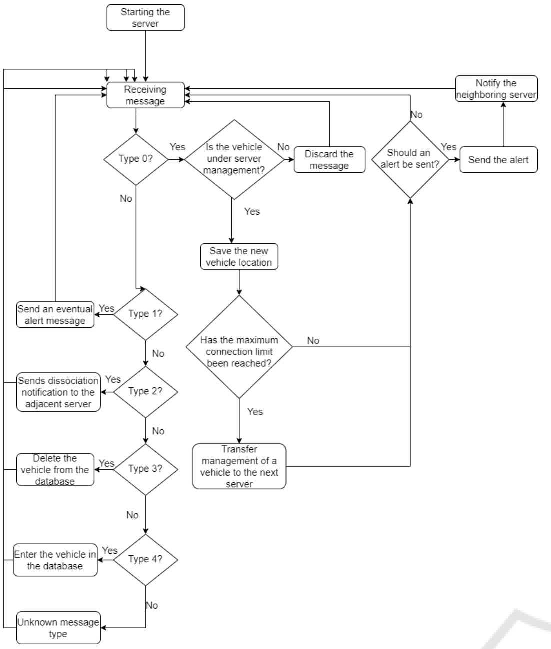

Figure 4: Flowchart of a server with Cell Breathing technique.

4 PERFORMANCE EVALUATION

In this section, the performance evaluation carried out through implemented simulator are shown. Different simulation are performed varying some input parameters. The number of vehicles considered in the simulative campaigns was varied from 1 to 6

A first result is referred to the number of packets managed by two servers that communicate with the vehicles on the road. In Figure 5, the number of packets managed without cell breathing paradigm is shown, while in Figure 6, the number of packets in the use case of cell breathing approach is depicted. In the first case, the management of vehicles and, therefore of the packets to be sent is entirely covered by the first server. In the second case, instead, despite the vehicle positions are the same, a better load distribution can be appreciated. This is because some of vehicles, although still in the range of the first server, are forced to perform handover, so moving under the management of the second server.

Another interesting analysis can be made on the time that elapses between the detection of the danger event and the communication of this event to the last vehicle of the set. The experiments were conducted using both handover techniques. Considering only the first mechanism, the vehicles are all under the control of server 1. Ten different simulations are performed in order to have an average value. The

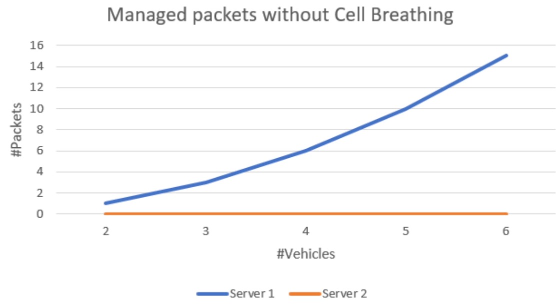

Figure 5: Packages managed without cell breathing technique.

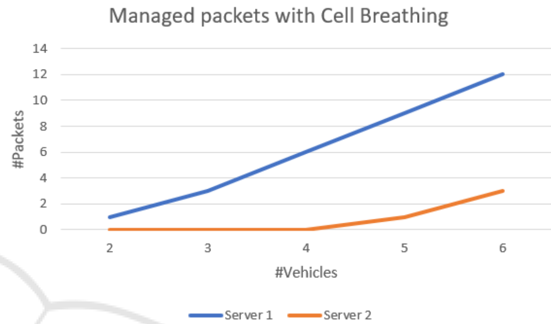

Figure 6: Packages managed with cell breathing technique.

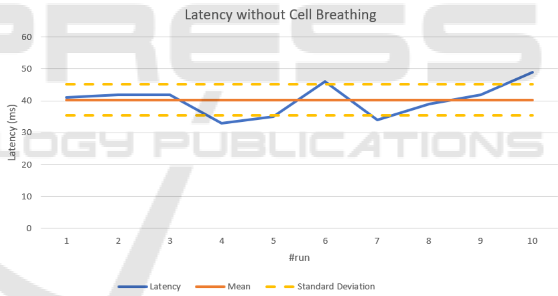

Figure 7: Latency without cell breathing technique.

recorded roundtrip time is around 40 milliseconds, see Figure 7, with a standard deviation of around 5 milliseconds. The time that elapses between the transmission of the message by the vehicle and the processing of the same by the server is about 20 milliseconds. Moreover, we have observed that the number of involved vehicles does not contribute to this latency because the experiments show that cars receive the danger signal almost simultaneously, with a time difference of less than one millisecond. Considering instead the second approach, half of the vehicles present in the area are under the control of server 2; the other half are under the control of server 1. In this case the recorded roundtrip time is around 60 milliseconds, with a standard deviation of around 5 milliseconds because another hop is added in the messages forwarding, see Figure 8.

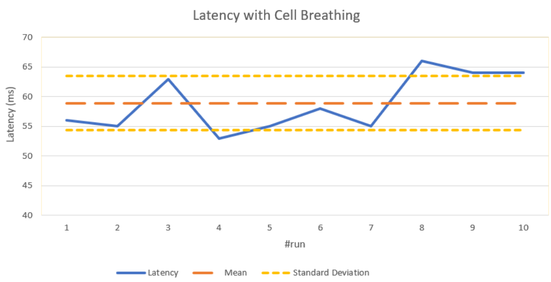

Figure 8: Latency with cell breathing technique.

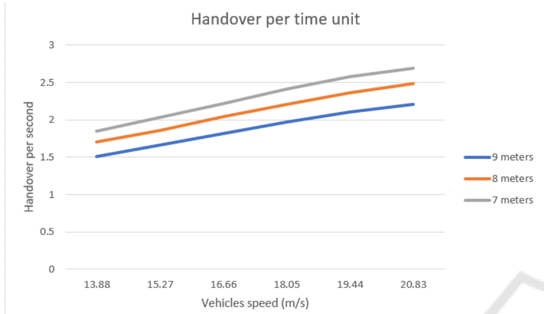

Figure 9: Handover per time unit.

Finally, the last analysis concerns the number of handovers. The ratio between the handover number and the employed time has been calculated. In this way the handover rate per seconds has been obtained. This value is influenced by the vehicles speed and their distance from the vehicle ahead. Proceeding at higher speeds, the BTS must manage a greater number of handovers, because the speed with which the mobile device changes position in the area is greater, see Figure 9. Considering same speeds, the number of handovers per time unit increases, as the distance between two vehicles decreases.

5 CONCLUSIONS

In this paper, a solution to guarantee the continuity of the service in an environment managed with Edge Computing technology has been proposed. Docker containers have been used to make applications light and portable. Kubernetes was used to orchestrate the Cluster containers and make the system robust. The results obtained show that, based on the algorithm used, there may be improvements in the management of the mobile devices. The use of Cell Breathing favors the balancing of the computational load, which becomes a crucial factor if compared to realistic environments, in which the number of vehicles to manage is quite high. Conversely, the latency recorded by some vehicles increases, which leads to delays in the delivery of messages.

REFERENCES

(2019). Docker: Empowering app development for developers. https://www.docker.com/.

(2019). Eclipse californium (cf) coap framework. https://www.eclipse.org/californium/.

(2019). Kubernetes: Production-grade container orchestration. https://kubernetes.io/.

(2019). Sumo - simulation of urban mobility. http://sumo.sourceforge.net/.

(2019). Traci/traas - sumo documentation. https://sumo.dlr.de/docs/TraCI/TraaS.html.

Artal-Villa, L., Hussein, A., and Olaverri-Monreal, C. (2019). Extension of the 3dcoautosim to simulate vehicle and pedestrian interaction based on sumo and unity 3d. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 885–890. IEEE.

Bachiega, N. G., Souza, P. S., Bruschi, S. M., and de Souza, S. d. R. (2018). Container-based performance evaluation: A survey and challenges. In *2018 IEEE International Conference on Cloud Engineering (IC2E)*, pages 398–403. IEEE.

De Rango, F., Fazio, P., and Marano, S. (2008). Utility-based predictive services for adaptive wireless networks with mobile hosts. *IEEE Transactions on Vehicular Technology*, 58(3):1415–1428.

De Rango, F., Veltri, F., Fazio, P., and Marano, S. (2009). Two-level trajectory-based routing protocol for vehicular ad hoc networks in freeway and manhattan environments. *Journal of Networks*, 4(9):866–880.

Doan, T. V., Nguyen, G. T., Salah, H., Pandi, S., Jarschel, M., Pries, R., and Fitzek, F. H. Containers vs virtual machines: Choosing the right virtualization technology for mobile edge cloud. In *2019 IEEE 2nd 5G World Forum (5GWF)*, pages 46–52. IEEE.

Erel-Özçevik, M. and Canberk, B. (2019). Road to 5g reduced-latency: A software defined handover model for embb services. *IEEE Transactions on Vehicular Technology*, 68(8):8133–8144.

Fazio, P., De Rango, F., Tropea, M., and Voznak, M. (2017). Cell permanence time and mobility analysis in infrastructure networks: Analytical/statistical approaches and their applications. *Computers & Electrical Engineering*, 64:506–523.

Fazio, P., Sottile, C., Santamaria, A. F., and Tropea, M. (2013). Vehicular networking enhancement and multi-channel routing optimization, based on multi-objective metric and minimum spanning tree. *Advances in Electrical and Electronic Engineering*, 11(5):349–356.

Fazio, P., Tropea, M., De Rango, F., and Voznak, M. (2016). Pattern prediction and passive bandwidth management for hand-over optimization in qos cellular networks with vehicular mobility. *IEEE Transactions on Mobile Computing*, 15(11):2809–2824.

Fotino, M., Gozzi, A., Cano, J.-C., Calafate, C., De Rango, F., Manzoni, P., and Marano, S. (2007). Evaluating energy consumption of proactive and reactive routing protocols in a manet. In *IFIP Conference on Wireless Sensor and Actor Networks*, pages 119–130. Springer.

Frnda, J., Voznak, M., Rozhon, J., and Mehic, M. (2013). Prediction model of qos for triple play services. In *2013 21st Telecommunications Forum Telfor (TELFOR)*, pages 733–736. IEEE.

Gesbert, D., Hanly, S., Huang, H., Shitz, S. S., Simeone, O., and Yu, W. (2010). Multi-cell mimo cooperative networks: A new look at interference. *IEEE journal on selected areas in communications*, 28(9):1380–1408.

Ghori, M. R., Zamli, K. Z., Quosthoni, N., Hisyam, M., and Montaser, M. (2018). Vehicular ad-hoc network (vanet): Review. In *2018 IEEE International Conference on Innovative Research and Development (ICIRD)*, pages 1–6.

Khan, W. Z., Ahmed, E., Hakak, S., Yaqoob, I., and Ahmed, A. (2019). Edge computing: A survey. *Future Generation Computer Systems*, 97:219–235.

Klas, G. (2017). Edge computing and the role of cellular networks. *Computer*, 50(10):40–49.

Lim, K. G., Lee, C. H., Chin, R. K. Y., Yeo, K. B., and Teo, K. T. K. (2017). Sumo enhancement for vehicular ad hoc network (vanet) simulation. In *2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, pages 86–91. IEEE.

Pan, J. and McElhannon, J. (2017). Future edge cloud and edge computing for internet of things applications. *IEEE Internet of Things Journal*, 5(1):439–449.

Salihin, S. S., Nissirat, L. A., Noor, R. M., and Ahmedy, I. (2018). Handover schemes for vehicular ad-hoc networks over long term evolution advanced: A survey. In *2018 International Conference on Computational Approach in Smart Systems Design and Applications (ICASSDA)*, pages 1–7. IEEE.

Santamaria, A. F., Raimondo, P., Tropea, M., De Rango, F., and Aiello, C. (2019). An iot surveillance system based on a decentralised architecture. *Sensors*, 19(6):1469.

Santamaria, A. F., Tropea, M., Fazio, P., and De Rango, F. (2018). Managing emergency situations in vanet through heterogeneous technologies cooperation. *Sensors*, 18(5):1461.

Shah, S. H. and Yaqoob, I. (2016). A survey: Internet of things (iot) technologies, applications and challenges. In *2016 IEEE Smart Energy Grid Engineering (SEGE)*, pages 381–385. IEEE.

Socievole, A., De Rango, F., and Coscarella, C. (2011). Routing approaches and performance evaluation in delay tolerant networks. In *2011 Wireless Telecommunications Symposium (WTS)*, pages 1–6. IEEE.

Thng, K. L., Yeo, B. S., and Chew, Y. H. (2005). Performance study on the effects of cell-breathing in wcdma. In *2005 2nd International Symposium on Wireless Communication Systems*, pages 44–49. IEEE.

Tinkhede, P. and Ingole, P. (2014). Survey of handover decision for next generation. In *International Conference on Information Communication and Embedded Systems (ICICES2014)*, pages 1–5. IEEE.

Zhou, B., Lee, Y.-Z., Gerla, M., and De Rango, F. (2006). Geo-lanmar: a scalable routing protocol for ad hoc networks with group motion. *Wireless Communications and Mobile Computing*, 6(7):989–1002.