IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 9, 2020, accepted July 22, 2020, date of publication July 28, 2020, date of current version August 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3012573

Distributed Mutual Exclusion Algorithms for Intersection Traffic Problems

HARISU ABDULLAHI SHEHU[1], (Graduate Student Member, IEEE), MD. HAIDAR SHARIF[2], AND RABIE A. RAMADAN[2,3]
[1]School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6012, New Zealand
[2]College of Computer Science and Engineering, University of Hail, Hail 2440, Saudi Arabia
[3]Computer Engineering Department, Cairo University, Cairo 12613, Egypt

Corresponding author: Md. Haidar Sharif (md.sharif@uoh.edu.sa)

ABSTRACT Conurbations around the globe are encountering the challenges of traffic congestion. Miscellaneous smart systems have been developed to help control and improve traffic flow in a cost-effective and measurable way. Yet, the existing systems solve the traffic congestion problem with cost-intensive traffic lights and mostly handle emergency cases poorly. In this article, to minimize the traffic congestion problems for any kind of vehicles without using traffic lights, we propose three deadlock-free algorithms namely: (i) Mutual exclusion algorithm based on single instruction (MEASIR), (ii) Mutual exclusion algorithm based on priority (MEAPRI), and (iii) Mutual exclusion algorithm based on multi-agent systems (MEAMAS). Communication inside a group is accomplished via a queue structure, while an external element (e.g., a router) is used for internal communications. Besides the depiction of experimental and simulation results, a complete statistical analysis has been performed to compare the performance of MEASIR, MEAPRI, and MEAMAS with their alternatives. Our proposed deadlock-free algorithms are not only efficient but also functional with a computational cost of $O(n)$ to enter the critical section, where n represents the number of all vehicles in a particular intersection.

INDEX TERMS Algorithm, critical section, mutual exclusion, multi-agent system, intersection, traffic congestion, vehicle.

I. INTRODUCTION

Cities and towns around the globe are coming upon the challenges of traffic congestion and increasing emissions. Nowadays, like home automation [1]–[3], traffic automation became one of the prominent factors for a smart city. Various intelligent transportation systems have been developed due to the increase in traffic congestion which brings about a question of whether there will be possible ways to improve the efficiency of how vehicles should beat the traffic [4]–[7]. Traffic light scheduling is the standard approach to solve the critical section problem at an intersection. In this approach, vehicles proceed in a stop-and-go style according to the occurrence of green light. Nevertheless, recent efforts on traffic light control focus on adaptive and smart traffic light scheduling. The key approaches include computational intelligent algorithm [8], evolutionary computation algorithm [9], traffic volume analysis [10], fuzzy logic [11], [12], neural network [13]–[15], and machine learning [16]. Determination of the optimal time for green light is a perplexing task, as traffic control systems are very complex nonlinear-stochastic systems [8], [17]. For security and safety, the traffic data processing should be as fast as possible using either a camera system or a laser scanner system [18], [19]. Moreover, computational complexity [20], [21] of smart algorithms is not suitable to apply real-time traffic light control. There exist several trajectory maneuver based algorithms (e.g., [22]–[24]), which are different from traffic light control. An intersection controller is deployed to optimally manipulate the trajectories of vehicles based on the conditions of nearby vehicles. Such a system can avoid potential overlaps. Both vehicles and intersection controllers communicate via wireless links.

An algorithm would be developed to only solve the problem related to traffic congestion by reducing the waiting time of vehicles at the intersection. But such an algorithm considers all vehicles on the lane to be the same by representing objects (vehicles) in a homogeneous way. Nevertheless, for emergency vehicles, reducing the amount of waiting time alone is not enough as in most cases.

The associate editor coordinating the review of this manuscript and approving it for publication was Sara Dadras.

The vehicles are either running to save people's lives in the case of an ambulance, or going after the bad guys (e.g., thieves, bandits, and etc.) in the case of a police car, or protecting the public in emergency situations (e.g., fire outbreak, earthquake, flooding, and etc.) in the case of a fire truck and many more. Overlooking any of these situations might result in a huge loss of people's lives and properties. Therefore, it is important for emergency vehicles to pass the intersection at the lowest possible time. However, in cases of emergency (e.g., accidents, fire, volcanic eruption, earthquake, mudslides, floods, tsunami, heatwave, hurricanes, law and order situations, and etc.), the emergency services in urban settings are frequently delayed at traffic intersections due to traffic delays. Consequently, the effectiveness of emergency service in many ways depends on response time. For instance, the response time of an ambulance can be defined as the time between reporting of an incident until when the ambulance has successfully been able to transport an affected individual to a hospital. Several studies have consistently asserted and proved the direct correlation between response time and degree of impact [25], [26]. According to the golden hour theory [27], [28], the initial 60 minutes after a traumatic injury are vital to chances of survival. For example, based on this theory if a patient of a road accident is rushed to a hospital within 60 minutes, the survival chances of the patient can increase significantly. In developing and underdeveloped countries, cost-intensive smart traffic lights are often not implemented, but rather follow a simple timer or green wave algorithm [29]. Henceforth, problems of traffic delays at traffic intersections for emergency services have not been resolved yet. Many intelligent algorithms have been proposed to control the traffic flow of vehicles at traffic intersections. For example, much research has been conducted to solve the traffic congestion problem at an intersection using vehicular ad hoc network (VANET) (e.g., [30]–[33]), where each vehicle communicates with other vehicles before proceeding to pass the intersection. Nevertheless, apart from the high cost, this may have, the method also has high message complexity.

In this article, to minimize the aforementioned existing problems at traffic intersections for any kind of vehicles flow, we propose three different mutual exclusion algorithms namely: (i) Mutual exclusion algorithm based on single instruction (MEASIR), (ii) Mutual exclusion algorithm based on priority (MEAPRI), and (iii) Mutual exclusion algorithm based on multi-agent systems (MEAMAS). In contrast to applying VANET, our current development uses ad hoc networks to enable vehicles to utilize a wireless network. The network is used to solve traffic congestion problems at a single-intersection. Our work does not allow interaction between vehicles; instead, all vehicles are with homogeneous features and communicate with a local controller (router). Our current trend could be deployed to solve the traffic congestion problem in smart cities. Mutual exclusion algorithms have been used to solve many real-world problems in distributed systems. Raymond's lock-based algorithm [34] for mutual exclusion allows each node to have only one parent and maintains a FIFO (first-in-first-out) queue of request is applied to tree on distributed resources. However, it is unknown if this same idea will function to solve traffic congestion problem. Since Raymond algorithm is guaranteed to be $O(\log n)$ per critical section (CS) entry, it is anticipated that the same idea will perform with a less complexity when applied to traffic congestion problem. We have utilized the same idea by making small modification and applying it to solve traffic congestion problem at intersection by assuming the node to be a lane. We have considered the rule of the mutual exclusion algorithm, which states that only one vehicle is allowed to be at the CS at any given time. But it is not true in most real road intersections. Both MEASIR and MEAPRI strictly utilize this mutual exclusion rule in which simultaneous access to shared resources is not allowed. Hence, both MEASIR and MEAPRI are starvation and deadlock-free. However, the existing problem incurred by the rule of mutual exclusion algorithm has been resolved completely in our proposed multi-agent systems (MAS) based MEAMAS, where more than one vehicle can pass the intersection at any given time. Initial communication inside group, between each vehicle and a local controller (router) is carried out using the upstream in the sense that each vehicle sends a request to the local controller as it arrives at the intersection. Nevertheless, the procedure at which the local controller grants token to vehicles is carried out in the downstream. Multiple comparisons without statistical tests and with statistical tests considering execution times and message costs illustrate effectiveness and efficiency of the Java-based implementation of MEASIR, MEAPRI, and MEAMAS over their alternative algorithms of CENDI [30], EMEV [35], DTLS [36], and EVSP [37]. The computational cost of each of our three algorithms is $O(n)$, whereas Raymond algorithm possesses $O(\log n)$. As compared to Raymond algorithm, our algorithms own higher computational complexity but lower than VANET based algorithms (e.g., Wu *et al.* [30]). As we are using the identical idea of Raymond algorithm, the computational complexity of our algorithms tends to be lower than their alternative algorithms. This could be one of the main reasons why our three comprehensively developed algorithms can achieve better performance for solving traffic congestion problem at the intersection.

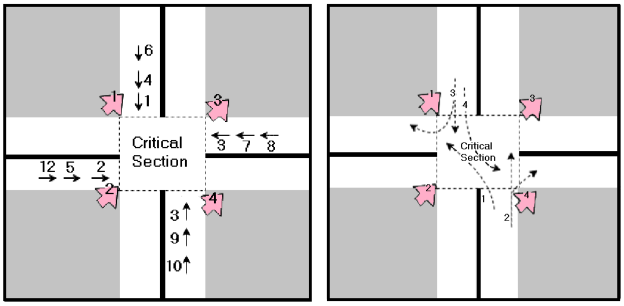
The rest of the paper is organized as: Section II briefs our proposed algorithms; Section III discusses related works; Section IV presents our system models; Section V illustrates our proposed algorithms; Section VI reports both experimental and simulation results followed by a detailed discussion including statistical tests as well as limitations of our algorithms that could be addressed in further study and finally, Section VII concludes the paper.

II. BRIEF IDEA OF OUR ALGORITHMS

Similar to Gradinariu and Tixeuil [38], MEASIR allows only a single instruction to be executed at a time; while

MEAPRI uses a priority-based technique e.g., Housni and Trehel [39]. MEAMAS applies the idea of multi-agent systems (MAS) [40]. It allows vehicles at the intersection to pass in group or move in platoon.

In this work, any vehicle that arrives at the intersection will be associated with a particular local controller, which in this case is a router at the intersection that will aid in letting the vehicle pass the intersection. Each router serves the vehicles on its lane by providing them with a token so as to be able to pass the intersection. One router is placed at the end of each intersection to accomplish a common task [41]–[44]. Therefore, four routers are required.



(a) With order of vehicles arrival (b) With multi-agent systems

FIGURE 1. **Two examples of an intersection. Vehicles and routers are represented by small and big arrows, respectively.**

All algorithms have both external and internal operations. In the external operation, a vehicle sends a request to its associated router as it arrives at the intersection, and then it will be put on the queue of vehicles that are on the same lane with it. In the internal operation, a router broadcasts a token request to other routers upon receiving a token request from a vehicle on its lane and the router is not in hold of the token. The router sends the token to grant a passing-permission to a vehicle on top of its queue if the router holds the token or upon getting the token from the other router. A common characteristic of MEASIR, MEAPRI, and MEAMAS is that they can provide a continuous green signal of traffic lights to the incoming emergency-service vehicles. In our proposed deadlock-free algorithms, normally vehicles do not communicate with each other (e.g., Hartenstein and Laberteaux [45] and Elhadef [46]) or via wireless communication (e.g., Wu *et al.* [30]), unless if a vehicle is a fire truck or an ambulance or a police car or a similar kind of emergency-service vehicle. Only emergency-service vehicles are granted token and allowed to pass the intersection as soon as they arrive at the intersection due to emergency cases. All other vehicles follow the standard procedure in passing the intersection. When a vehicle arrives at the intersection, it sends a request to its associated router for granting a privilege to pass the intersection. It will only be able to pass the intersection by the time it has been granted a token by its router. Only one vehicle is allowed to be at the CS at a time. Two examples of an intersection are given in Fig. 1(a) and Fig. 1(b) by considering vehicle arrival and MAS, respectively.

Both MEASIR and MEAPRI assume that there is no message lost, whereas MEAMAS uses the idea of MAS to allow vehicles to pass in a group. But the assumption of MEASIR and MEAPRI has been resolved in MEAMAS. Because MEAMAS ensures that all vehicles receive the token. All vehicles send *ack* (acknowledgment) to the router they receive the token from. In the case of a message lost, the router sends the token again to a vehicle after a fixed time and *ack* has not been received. A message is assumed to be lost when the counter finishes counting and no *ack* message has been received.

III. STATE-OF-THE-ARTS

Existing intelligent algorithms for solving problems at traffic intersections would be roughly categorized into two groups: (i) Emergency cases exclusive algorithms; and (ii) Emergency cases inclusive algorithms.

A. EMERGENCY CASES EXCLUSIVE ALGORITHMS

Emergency cases exclusive algorithms do not explicitly take into account the emergency circumstances. For example, Kumar and Umesh [47] presented an improved version of a distributed mutual exclusion algorithm using the concept of Maekawa's algorithm [48]. A coordinate system is widely used in computer vision [49]. Cheng and Yang [50] proposed a distributed and coordinated traffic signal control system based on multi-agent. Their simulation results claimed that their system reduced 37.8% total stop delay of cars than the fixed method and reduced 17.8% total stop delay of cars than the actuated method under the same condition. Mu *et al.* [51] proposed a distributed control system which comprised of some local fuzzy controller and a special case controller to solve traffic problem at an urban traffic intersection. Elhadef [46] improved the intelligent VANET based intersection control algorithm by making it more adaptable to realistic traffic scenarios and traffic bottlenecks. It is claimed that their algorithm satisfied the property of safety, liveness, and fairness of vehicle mutual exclusion for intersections problem. Park *et al.* [52] proposed a token-based group mutual algorithm for intersection traffic control for autonomous vehicles. Their algorithm decreased message complexity and performed better in system throughput than an existing traffic signal system.

Li *et al.* [53] addressed an approach to set the timing of traffic signals using reinforcement learning. Simulation results showed that the average delay was reduced to 14% when deep reinforcement learning method was used instead of ordinary reinforcement learning method. It was also found that a vehicle might spend about 13 seconds to pass the intersection even in morning peak hours. It was also assumed that no red-clearance happens and one phase occurs immediately after another which might increase the robustness of the system. However, Li *et al.* [53] set the minimum green time for each lane as 15 seconds. Assuming a situation where there are no vehicles on a particular lane and the traffic light is set to green when other vehicles are waiting on other lane. This will only increase the waiting time, causing other vehicles to wait more at the intersection without utilizing the empty space. Furthermore, Li *et al.* [53] assumed that there are two phases (north-to-south and south-to-north) for the vehicles to pass the intersection, which imposes a restriction on the direction in which vehicles should flow. This is not the case in real-world situation as a vehicle moving from one lane might go to either three sides of the intersection.

Qi *et al.* [54] presented an approach to real-time traffic emergency systems for intersections facing accidents using Petri nets. Different kinds of facilities including sensors to gather real-time traffic, cameras to detect accidents, loop detectors to count the number of vehicles, warning facility for sending accidental warning signals for vehicles, and traffic lights were installed. Petri net-based model was designed to ensure the correctness of the control system. Qi *et al.* [54] only provided a theoretical explanation and made assumption that the control logic validity can be verified without a simulation method. In addition, the approach may not be cost-effective in sense that it solves the problem by deploying too many facilities on the intersection. The approach emphasized more on emergency traffic light strategies for an accidental intersection rather than traffic light behaviors to support the smooth movement of emergency-service vehicles.

Huang *et al.* [55] proposed a synchronized timed Petri nets to regulate a traffic light control system to handle traffic congestion problem. A regulatory traffic light controller was designed to select a suitable traffic light phase in case of heavy traffic congestion. Five signal lights (namely the red light, yellow light, left turn arrow on the green, straight arrow on the green, and the right turn arrow on green) were used to make two-phase, six-phase, and eight-phase transitions. The liveness and reversibility of their model were proven through the reachability graphs analysis. Congestion was avoided using conditions and events of their model by assigning more green light time to a fixed phase that had more traffic. This may reduce traffic congestion, but it might lead to an excessive delay for other vehicles thereby violating fairness among vehicles at the intersection.

Wu *et al.* [30] focused on both centralized and distributed algorithms (CENDI) to solve the traffic congestion problem at intersection. In their centralized algorithm, a control center node was deployed at the intersection area. The control node communicated with vehicles. It controlled vehicles to pass the intersection successfully. Conversely, their distributed algorithm used VANET to enable vehicles to communicate with other vehicles. Basically, CENDI [30] made an assumption that the wireless channel should work in the FIFO order. The preemption of CENDI [30] allowed a vehicle with low priority to pass the intersection in certain situations, which were not considered an emergency case. In CENDI [30], a vehicle upon entering the intersection sent a message to all other vehicles (n) on the lane and waited for a reply from all other vehicles on the lane before processing to pass the critical section. This scenario made the message cost of $n \times n = n^2$ for CENDI [30]. In other words, the message complexity of CENDI [30] is as high as $O(n^2)$, where n represents the number of vehicles at a particular intersection.

Authors of [30], [53]–[55] demonstrated different views to solve traffic congestion problems at the intersection. But there was no evidence as to whether deadlock occurred or not with the no-red clearance of the algorithm proposed by Li *et al.* [53], and in case of failure in sending warning signals in Qi *et al.*'s work [54]. Apart from the high message cost of CENDI [30], allowing low priority vehicles to pass over high priority vehicles in Wu *et al.*'s work [30] and assigning more green light to a particular lane in Huang *et al.*'s work [55] failed to fully address fairness issues among vehicles at the intersection. In addition, approaches of [30], [53]–[55] made no attempt to consider what would happen in case of an emergency. This is anticipated to make these models less robust towards ensuring that emergency vehicles do not wait very long at the intersection.

B. EMERGENCY CASES INCLUSIVE ALGORITHMS

Emergency cases inclusive algorithms explicitly discuss emergency cases. For example, Tchuitcheu *et al.* [56] suggested a collision-free distributed algorithm for controlling traffic signals using the information provided by smart cameras. Their algorithm used smart cameras for real-time monitoring and assessment. Those cameras not only understood the traffic flow but also tracked special vehicles and prioritized emergency cases successfully. However, an example input showed that their algorithm was tested on three-lane rather than four-lane of an intersection. As such, questions remain unanswered as to whether their algorithm would achieve the same performance when tested four lanes of an intersection. In addition, if a camera will fail to fully capture the whole image of a vehicle, then the performance of their algorithm is not clear. Singh *et al.* [35] proposed an algorithm for the effective movement of emergency vehicles (EMEV) from the traffic control signal. The algorithm of EMEV [35] solved the problem by setting the traffic light of the lane in which an emergency vehicle was detected to green. It allowed not only the emergency vehicle but also all vehicles on the lane to pass the intersection while other vehicles were waiting at the intersection. This condition violated fairness among vehicles. This might also lead to starvation for other vehicles on other lanes. Krishna *et al.* [36] proposed a dynamic traffic light system (DTLS) for unhindered passing of high priority vehicles. The algorithm of DTLS [36] assumed that it was a rare case to have more than one emergency vehicle at an intersection and delayed all other vehicles for 2 seconds when another emergency vehicle was already at the intersection. This condition increased the overall algorithmic turnaround time. Furthermore, the effectiveness of DTLS [36] was assumed. A sensor was used to detect and set the traffic light of a fixed lane to green if there existed any emergency vehicle on the lane rather than testing it at an intersection. As such questions remain unanswered, it is hard to say whether DTLS [36] will work at an intersection. Anil *et al.* [37] addressed an emergency vehicle signal pre-emption (EVSP) system for heterogeneous traffic conditions. Based on the circumstances, the delay should be less for the queue with minimum length. However, the algorithm of EVSP [37] used lesser green light to a queue with minimum length. This might be good in a way not to cause traffic flow in other lanes. But it violated fairness among vehicles at the intersection, causing vehicles in lesser queue length waited more at the intersection. Other way round, if the number of vehicles in the lanes will increase, the execution times of EMEV [35], EVSP [37] and DTLS [36] will increase linearly (e.g., EMEV [35], EVSP [37]) or polynomially (e.g., DTLS [36]). In other words, precision and accuracy [57] degradation occur in these algorithms with the augmentation of the number of vehicles at the intersection.

The aforementioned discussion suggests that there exists further space for developing smarter algorithms to solve traffic congestion problems at the intersection.

C. CONTRIBUTIONS OF MEASIR, MEAPRI, & MEAMAS

In this article, we have aimed to address both emergency and non-emergency cases by proposing three different distributed mutual exclusion algorithms named MEASIR, MEAPRI, and MEAMAS. These algorithms ensure robustness toward the aforementioned issues, allowing vehicles to pass the intersection in an effective and efficient manner.

Our initial idea came from the non-emergency case handling algorithm of CENDI [30]. Thus, our current development would be considered as the incremental improvement of CENDI [30]. This improvement includes: (i) MEASIR, MEAPRI, and MEAMAS overcome the fairness violation among vehicles of similar kinds in CENDI [30]; (ii) MEASIR, MEAPRI, and MEAMAS are starvation and deadlock-free; (iii) MEASIR, MEAPRI, and MEAMAS handle both emergency and non-emergency cases; (iv) The computational complexity is reduced from $O(n^2)$ of CENDI [30] to $O(n)$ of either MEASIR or MEAPRI or MEAMAS; (v) Our most advanced algorithm MEAMAS overcomes completely the existing problem of the rule of mutual exclusion algorithm and allows more than one vehicle (i.e., a group of vehicles) to pass the intersection at any given time.

In addition, CENDI [30] assumes that all vehicles to be utilizing the VANET, which is not cost-effective. This can be considered as a disadvantage of CENDI [30], but the mind-taken disadvantages of CENDI [30] should be its message cost and violating fairness among vehicles. CENDI [30] allows preemption where a vehicle arriving at an intersection might be allowed to proceed to the CS without waiting for other existing and waiting vehicles. Henceforth, it allows vehicles with lower priority to pass the intersection before vehicles with higher priority in certain situations. This case both violates fairness among vehicles at CS and leads to a conflict in real-life traffic problems. If a real-time emergency-service vehicle will have to send messages to all existing and waiting vehicles at CS as it arrives at the CS, it will wait for *ack* from all of those vehicles at CS. But the emergency-service vehicle will not know the duration of response time from those vehicles. Within its waiting time at CS, many lives and/or properties would be lost. The situation can be worsened, if its sent message will be lost anyway. While this might not be an issue for other vehicles (not in a hurry), it is a great deal for emergency-service vehicles as well as people going to work, students going to school, and people trying to go for other essential needs, and so on. However, all of these problems have been addressed in our proposed algorithms.

IV. SYSTEM MODELS

A. COMMUNICATION

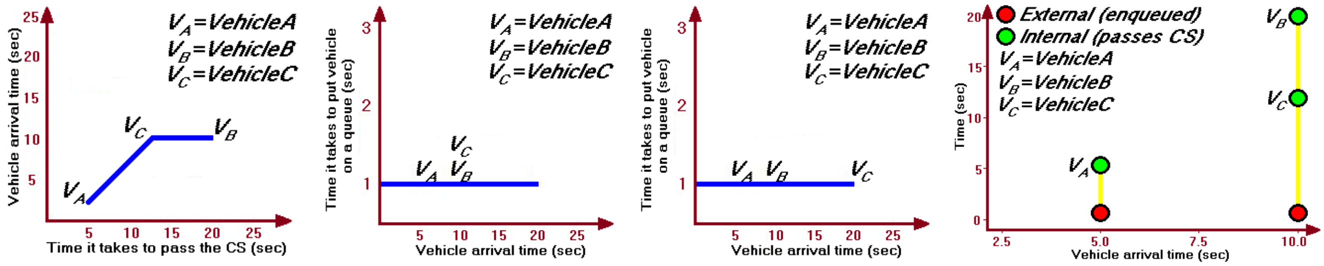
Assume that there is no loss, duplication, or modification of messages. In the former case, only a single message can be sent at a time in the internal communication, while in the later case several messages can be sent simultaneously. If several messages are sent simultaneously in the internal communication, the router with higher priority is given the token. Even if a token is being requested by a router with higher priority while a vehicle is at the critical section, the requested router has to wait first for that particular vehicle to pass the CS before the token can be granted to it.

1) INTERNAL OPERATION

In the internal operation, four routers are placed. Each router is associated with one intersection. A router broadcasts a token request to other routers upon receiving a token request from a vehicle on its lane. The router is not in hold of the token. The router sends the token to grant a passing-permission to a vehicle on top of its queue upon receiving the token from the router that is in hold of the token. The time to take for a vehicle for passing the intersection increases linearly with respect to the vehicle's arrival time as shown in Fig. 2(a). This is not always true in the case that when two different vehicles on different lanes arrive at the intersection simultaneously. The time that it will take for one of the vehicles to pass the intersection is obviously going to be greater than the time it will take for the other vehicle to pass. For example, vehicle A that arrives in 2 seconds is able to pass the intersection when the time is 5 seconds; whereas vehicle B and vehicle C that both arrive at the intersection in 10 seconds from different lanes but pass at different times. Vehicle C passes when the time is 12 seconds, while vehicle B passes when the time is 20 seconds. This is due to the fact that vehicle C arrives at the intersection when its router possesses the token or its router has a higher priority than that of vehicle B.

2) EXTERNAL OPERATION

In the external operation, the vehicle sends a request to its associated router as it arrives at the intersection.



(a) Vehicles pass asynchronously. (b) Vehicles arrive simultaneously. (c) Vehicles arrive asynchronously. (d) Modelling using line segments.

FIGURE 2. Possible occurring events in (a) internal operation as well as (b) and (c) external operations. Relationship between the time in internal operation of (a) and external operation has been shown in (d) using line segment plotting. The enqueued time in the external operation is constant.

Upon receiving a request from a vehicle, the associated router puts the vehicle to its associated queue. The vehicle is enqueued to a queue as it arrived at the intersection. For this reason, we can say that the time it takes to put a vehicle on a queue is constant with respect to its arrival time as shown in Fig. 2(b) and Fig. 2(c). In Fig. 2(b), it takes one second to put vehicle A that arrives at the intersection after 5 seconds as well as vehicle B and vehicle C that arrive simultaneously at the intersection after 10 seconds. In Fig. 2(c), it takes one second to put vehicle A, vehicle B, and vehicle C on to the queue, even though they arrive at the intersection after 5 seconds, 10 seconds, and 20 seconds, respectively.

3) LINE SEGMENTS REPRESENTATION

Fig. 2(d) models how the time in internal and external operations relates to each other. The red circle shows how long it will take for a vehicle to be enqueued as it arrives at the intersection. The green circle indicates the duration it will take the vehicle to pass the CS and the orange line makes the relationship between the internal and external operation with respect to time. From Fig. 2(a), it takes the vehicle A just one second to be enqueued and a total of 5 seconds to pass the intersection. Both vehicles B and C arrive at the intersection simultaneously. It takes only one second for them to be enqueued. But the vehicle C exits the intersection in 12 seconds, whereas the vehicle B exits the intersection in 20 seconds. It is noticeable that the time to take all vehicles to be enqueued in the external operation is constant (just 1 second) across all vehicles. But the time to take each vehicle to pass the intersection may differ depending on what time and from which lane a vehicle arrives at the intersection. Nevertheless, this does not violate the fairness among vehicles on the same lane, as all vehicles arriving at the same lane pass the intersection using the FIFO order.

B. CRITICAL SECTION (CS)

A vehicle can request for a token to be able to enter into the critical section just once from its router. If the vehicle passes the critical section, it should send the token back to its associated router.

C. NOTATION AND MESSAGE TYPE

This is also the same as the centralized algorithm [30], whereby a vehicle undergoes three phases to pass the intersection. The three phases are mentioned below.

- IDLE ⇒ A vehicle is said to be in the idle state, if it is out of the queue area.
- WAITING ⇒ A vehicle is said to be in the waiting state, if it waits for a permission to enter into the critical section.
- PASSING ⇒ A vehicle is said to be in the passing state, if it passes the intersection after getting a permission to pass by its associated router.

D. ADDITIONAL NOTATIONS

- *cnt* ⇒ It is a counter to record the number of vehicles that pass the intersection from a particular lane at a time.
- *mp* ⇒ Minimum number of vehicle that should pass a particular lane at a time.
- *e* ⇒ Emergency parameter that allows a vehicle to be granted a token immediately as it arrives at the intersection without waiting if $e = 1$. Otherwise, it follows normal procedure if $e = 0$.
- *ack* ⇒ Acknowledgment message between communicating processes.

The emergency parameter is always set to 0 unless if a vehicle is an ambulance or a fire truck or a police car which should pass the intersection immediately due to emergency.

V. OUR PROPOSED ALGORITHMS

In this section, we have provided the detailed of MEASIR, MEAPRI, and MEAMAS. Except in emergency situations, any vehicle at the intersection only gets token from the routers on its lane. In the case of an emergency, any router that is in hold of the token sends the token to an emergency vehicle. If none of the routers are in hold of the token, all routers search for the token in their respective lanes, and then the router that acquires the token sends the token to the emergency vehicle.

A. DEFINITION OF NOTATIONS

This subsection provides insight of notations used in MEASIR, MEAPRI, and MEAMAS.

- *R* represents a router.

- i, j, k, and l represent the identity (id) of the four ends of the intersection, where each id indicates an end of a particular intersection.
- $R_{i,j,k,l}$ represents routers in four ends of the intersection, where R_i indicates a router at intersection i, R_j points to a router at intersection j, etc.
- $R_{i,j,k,or\, l}$ provides a scenario, where only one of R_i, R_j, R_k, or R_l is involved.
- $R_{i,j,k,l}$ provides a scenario, where all routers R_i, R_j, R_k, and R_l are involved.
- $R_{j,k,or\, l}$ provides a scenario, where only one of R_j, R_k, or R_l is involved.
- $R_{j,k,l}$ provides a scenario, where all routers R_j, R_k, and R_l are involved.
- The vehicle i represents a vehicle that arrives at the intersection from lane i.
- The vehicle j represents a vehicle on top of R_i's queue.
- $R_{i,j}$ represents a scenario, where a router on lane i sends token to a vehicle j on top of its queue. All routers send token to vehicles together with the router's id so that the vehicle will know to which router it will send back the token after it exits the intersection.

B. FLOWCHARTS OF MEASIR AND MEAPRI

Basically, the flowchart in Fig. 3 is directly related to MEASIR. It highlights how a vehicle on arriving at the intersection will proceed to pass the intersection using. Initially, upon entering the intersection, if the arrival vehicle belongs to an emergency-service vehicle, it broadcasts a request to all routers. If any of the routers has the token, it sends the token to that emergency-service vehicle immediately. Otherwise, each router will search for the token on its lane. If any router will get the token, it will send the token to the requested emergency-service vehicle. Conversely, if the vehicle is not related to an emergency-service vehicle, the vehicle will be enqueued to the bottom of the queue of its own router. If the router will have the token at the time of the request, then it will send the token to the vehicle on top of its queue. If the router will not have the token at the time of the request, the router will acquire the token by sending a request to other routers. On receiving the token from the router, the vehicle proceeds to the CS and sends back the token to the router. When a router sends a token to a requested vehicle, it sends the token together with its identity (id) to the vehicle. Using id, all vehicles know which routers to send the token back after exiting the intersection. Identical procedures in Fig. 3 can be adopted for MEAPRI, except for the priority-based token management in MEAPRI. Even if all routers requested for the token simultaneously, the router with the highest priority will be granted the token. Priority information of all routers will be updated simultaneously to ensure all routers get the token to serve vehicles on their lanes.

C. MEASIR BASED ON SINGLE REQUEST

The implementation detailed of MEASIR is illustrated in Algorithm 1.

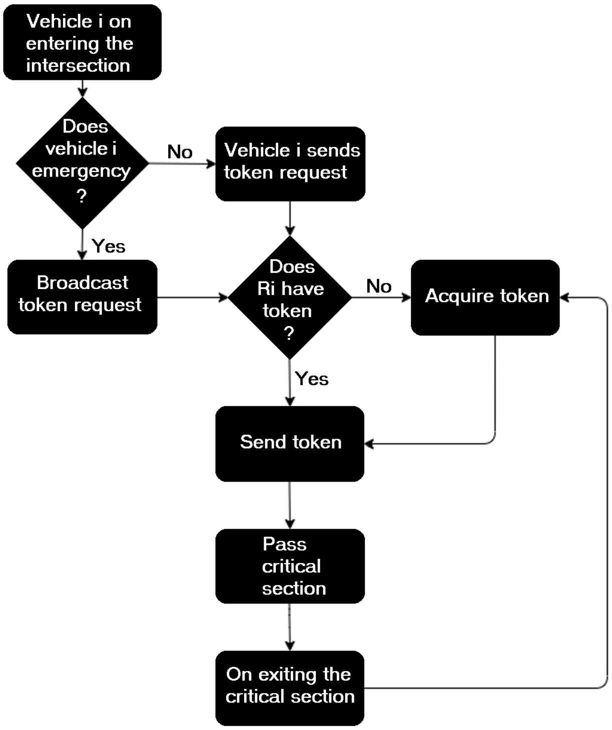


FIGURE 3. Generic flowchart of MEASIR and MEAPRI.

1) SPECIFICATION

Communications occur one after another i.e., only a single router can send a request to other routers at a time.

2) OPERATION OF REQUEST

Unless if the vehicle is an ambulance or a fire truck or a police car, whereby the vehicle i sends token request by setting $e = 1$. The rest vehicles send token request by setting $e = 0$. A vehicle that sends a request by setting $e = 1$ usually broadcasts the request to all the routers. When the routers receive a broadcast request from vehicle i, the following items happen.

- If any of the routers ($R_{i,j,k,or\, l}$) has token, $R_{i,j,k,or\, l}$ with the token sends it to vehicle i.
- If none of the routers $R_{i,j,k,l}$ has token, all routers $R_{i,j,k,l}$ try to acquire token from its lane from a vehicle that holds it (if any).
- The router with the token in $R_{i,j,k,l}$ sends the token to vehicle i.

On the other hand, when any other vehicle that sends a request by setting $e = 0$ enters the queue area. It switches from IDLE to WAITING and sends a token request to its router. When a router R_i receives the request by the vehicle, the following items happen.

- R_i adds i to its queue.
- If R_i has the token, then R_i sends the token to vehicle j, where vehicle j represents the vehicle on top of R_i's

Algorithm 1 MEASIR (Mutual Exclusion Algorithm Based on SIngle Request)

```
1:  Give token to one of the router.
2:  Initialize $cnt$, $mp$.
    CoBegin // for a vehicle $i$
    On entering the monitoring area:
3:  if Vehicle $i$ is an ambulance, fire truck, police car, or a vehicle of similar kind then
4:      Vehicle $i$ broadcasts token request to all routers ($R_{i,j,k,l}$, $e = 1$).
5:      if $R_{i,j,k,or\,l}$ then has token
6:          $R_{i,j,k,or\,l}$ sends token to vehicle $i$.
7:      end if
8:      if $R_{i,j,k,l}$ does not have token then
9:          $R_{i,j,k,or\,l}$ acquires token from a vehicle (on their lane) in hold of the token.
10:         $R_{i,j,k,or\,l}$ sends token to vehicle $i$.
11:     end if
12: end if
13: Vehicle $i$ is in a WAITING state.
14: if Vehicle $i$ is not associated with any router. then
15:     Vehicle $i$ sends token request to the router in its lane ($R_i$, $e = 0$).
16:     Vehicle $i$ waits for response from $R_i$.
17: end if
    On receiving token request($R_i$, $i$) from $i$
18: $R_i$ puts $i$ on its queue.
19: if $R_i$ has token then
20:     $R_i$ sends token to vehicle $j$ on top of its queue ($R_i$, j).
21:     Increase $cnt$ by 1.
22: end if
23: if $R_i$ does not have token then
24:     $R_i$ sends token request to its neighbours $\{R_{j,k,l}\}$.
25:     if $R_{j,k,or\,l}$ has token AND $cnt$ greater than or equal to $mp$ then
26:         $R_{j,k,or\,l}$ sends token to ($R_i$).
27:     end if
28:     if $R_{j,k,or\,l}$ has token AND $cnt$ less than $mp$ AND the lane is empty then
29:         $R_{j,k,or\,l}$ sends token to ($R_i$).
30:     end if
31:     if $R_{j,k,or\,l}$ has token AND $cnt$ less than $mp$ AND the lane is not empty then
32:         Delay $R_i$ for some time.
33:         $R_{j,k,or\,l}$ sends token to ($R_i$).
34:         Repeat from step 19.
35:     end if
36: end if
    On exiting the intersection
37: Vehicle $j$ sends back token to $R_i$.
38: $R_i$ removes vehicle $j$ from its queue.
    CoEnd return
```

queue. This is performed up to the number of request received by R_i.

- If R_i does not have token, R_i broadcasts token request to its neighbors.
- If any of its neighbors $R_{j,k,or\,l}$ has token and counter greater than or equal to mp, then the token is sent to R_i, where counter holds the number of vehicles already granted a lock and mp be the minimum number of vehicles that should be granted a lock at a time.
- If any of the neighbors has the token and counter less than mp but the lane is empty, then the neighbor with the token sends the token to R_i.
- If any of the neighbors has the token, counter less than mp and the lane is not empty, R_i is being delayed for some time in such a way that counter will be greater than or equal to mp. Afterward, the token will be send to R_i.

3) OPERATION OF PASSING

If $e = 1$, then routers send the token to vehicle i if they are in hold of it, else send the token to vehicle i after acquiring it from any vehicle in the intersection that is holding token. Vehicle i proceeds to pass the critical section afterward. On the other hand, if $e = 0$ and the token is being received by the router R_i, then the router sends the token to vehicle j, where vehicle j represents the vehicle on top of the queue of R_i. The vehicle then proceeds to pass the critical section.

4) OPERATIONS OF RELEASE

This is the same whether the emergency parameter is set to either $e = 0$ or $e = 1$. After a vehicle (i or j) passes the critical section, the vehicle sends back (release) the token to its router R_i or the router it received the token from $R_{i,j,k,or\,l}$ in the case of an emergency vehicle.

D. MEAPRI BASED ON PRIORITY

The step by step procedures of MEAPRI illustrate in Algorithm 2.

1) SPECIFICATION

Communication occurs all the time i.e., all routers send a token request to other routers whether they are in need of the token or not. To avoid deadlock, this is handled using a priority. A priority is given to each and every router and often receiving a request. The token holding router sends the token to the requested router with a higher priority. The key difference between MEASIR and MEAPRI is how the token is being given to the routers upon requesting from other routers. A control mechanism [58]–[60] is being used to set up the priority of the routers. In addition, the priority function PR_SETTING is required for MEAPRI.

2) OPERATIONS

In fact, the operations performed on MEAPRI is the same as that of MEASIR. The only difference between the two algorithms is that the former gives a permit (granted lock) to the router with a higher priority. The priority of the router is controlled and can be set by dint of a control mechanism. The function PR_SETTING provides an overview of how the

Algorithm 2 MEAPRI (Mutual Exclusion Algorithm Based on PRIority)

```
1:  Give token to one of the router.
2:  Randomly assign other routers priority in order of 4, 3, 2.
3:  Initialize $cnt$, $mp$.
    CoBegin // for a vehicle $i$
    On entering the monitoring area:
4:  if vehicle $i$ is an ambulance, fire truck, police car, or a vehicle of similar kind then
5:      vehicle $i$ broadcasts token request to all routers ($R_{i,j,k,l}$, $e = 1$).
6:      if $R_{i,j,k,or\ l}$ then has token
7:          $R_{i,j,k,or\ l}$ sends token to vehicle $i$.
8:      end if
9:      if $R_{i,j,k,l}$ does not have token then
10:         $R_{i,j,k,or\ l}$ acquires token from a vehicle (on their lane) in hold of the token.
11:         $R_{i,j,k,or\ l}$ sends token to vehicle $i$.
12:     end if
13: end if
14: Vehicle $i$ is in a WAITING state.
15: if vehicle $i$ is not associated with any router then
16:     vehicle $i$ sends token request to the router in its lane ($R_i$, $e = 0$).
17:     vehicle $i$ waits for response from $R_i$.
18: end if
    On Receiving token request($R_i$, $i$) from $i$
19: $R_i$ puts $i$ on its queue.
20: if $R_i$ has token then
21:     $R_i$ sends token to vehicle $j$ on top of its queue ($R_i$, $j$).
22:     Increase $cnt$ by 1.
23: end if
24: if $R_i$ does not have token then
25:     $R_i$ sends token request to its neighbours $\{R_{j,k,l}\}$.
26:     if $R_{j,k,or\ l}$ has token AND $cnt$ greater than or equal to $mp$ then
27:         Call PR_SETTING().
28:     end if
29:     if $R_{j,k,or\ l}$ has token AND $cnt$ less than $mp$ AND the lane is empty then
30:         Call PR_SETTING().
31:     end if
32:     if $R_{j,k,or\ l}$ has token AND $cnt$ less than $mp$ AND the lane is not empty then
33:         Delay $R_i$ for some time.
34:         Call PR_SETTING().
35:         Repeat from step 20.
36:     end if
37: end if
    On exiting the intersection
38: Vehicle $j$ sends token to $R_i$.
39: $R_i$ removes vehicle $j$ from its queue.
    CoEnd return
```

```
function PR_SETTING()
    Send token to router with higher priority among $R_{i,j,k,l}$
    if $R_i$ has higher priority then
        Subtract 1 from the priority of $R_{j,k,l}$.
        Set the priority of $R_i$ to be 4.
    end if
    if $R_j$ has higher priority then
        Subtract 1 from the priority of $R_{i,k,l}$.
        Set the priority of $R_j$ to be 4.
    end if
    if $R_k$ has higher priority then
        Subtract 1 from the priority of $R_{i,j,l}$.
        Set the priority of $R_k$ to be 4.
    end if
    if $R_l$ has higher priority then
        Subtract 1 from the priority of $R_{i,j,k}$.
        Set the priority of $R_l$ to be 4.
    end if
end function
```

priority function is being set. Initially, priority is assigned randomly from 1 to 4 to the routers R_i, R_j, R_k, and R_l. Based on that, the router with higher priority will be given the token. For instance, assuming R_i is the router with a higher priority of 4, R_j is a router with a priority of 3, R_k is a router with a priority of 2, and R_l is the router with the lowest priority of 1. The router R_i will send the token to R_j, if there is no vehicle on R_i's lane, all vehicles have passed, or the maximum number of vehicles that should pass the intersection at a time have passed from R_i's lane. If one of these conditions is satisfied, R_j will be set to be the router with higher priority, R_k will be set to have a priority of 3, R_l will be set to have a priority of 2, and R_i will be set to have to lowest priority of 1. The usage of priority, in this case, prevents the possibility of deadlock occurrence.

E. MEAMAS BASED ON MAS

We have proposed MEAMAS to handle the problem of mutual exclusion by allowing vehicles to pass as a group. Basically, we have used the idea of MAS [40] to increase the efficiency of MEASIR and MEAPRI by proposing a new algorithm called MEAMAS. It approves vehicles at intersections to pass in group or move in platoon. Fig. 4 shows an illustrative example of the intersection using MAS, where vehicles 1 and 2 at lane number 4 as well as vehicles 3 and 4 at lane number 1 are grouped as multi-agent. The router at lane 4 sends the token to the two grouped vehicles (1 and 2) simultaneously. Both groups proceed to the critical section to pass the intersection simultaneously. While vehicles 1 and 2 enter the critical section all with the goal of passing the intersection at the same time, MEAMAS does not violate the rule of mutual exclusion as vehicle 1 and vehicle 2 are both exiting the intersection from a different lane. Vehicle 1 exits the intersection from lane 2 controlled by router 2. Vehicle

2 exits the intersection from either lane 1 or lane 3, which are controlled by router 1 and router 3, respectively. Unlike both MEASIR and MEAPRI, MEAMAS does not only allow vehicles to pass the lane in an efficient way but also ensures that all vehicles receive the token. Each vehicle sends *ack* after they receive the token from their associated router. The router enters the waiting state to wait for vehicles to send the *ack*. It sends the token again to any vehicle that did not send the *ack* after a fixed time. If still there exists any vehicle that did not send the *ack* after sending it again, then the router sends a stop to that vehicle or those vehicles and sends a traffic violation signal to the control center.

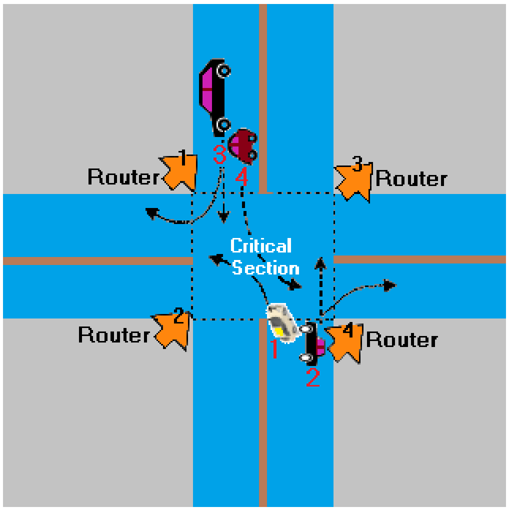


FIGURE 4. **Vehicles 1 and 2 at lane number 4 along with vehicles 3 and 4 at lane number 1 are grouped as multi-agent.**

F. FLOWCHART AND ALGORITHM OF MEAMAS

The flowchart and implementation details of MEAMAS are illustrated in Fig. 5 and Algorithm 3, respectively.

The flowchart in Fig. 5 focuses on how a vehicle *i* proceed to pass the intersection using MEAMAS. Apart from sending a token to a group rather than to a particular vehicle, the same operation in MEASIR (i.e., Fig. 3) is carried here until a group has been sent the token. A router will send a token to a group and after successfully getting the token the group will send an *ack* message to the route. On receiving an *ack* message, the group can proceed to pass the CS. If the message will be lost initially when the token will be sent, then it is important to check whether the waiting time to receive a token will be exceeded or not, which will be very unlikely to occur in the initial stage. If the token will not receive and the waiting time will not be exceeded, then the sender router will resend the token after a fixed waiting time and without receiving *ack* message. If this situation will continue to occur simultaneously, then a traffic violation alert will be sent to the control center. If the token will be received but the *ack* message will be lost, the group cannot proceed to pass the CS and the router will assume that the group will not receive the token after a certain time. The router will resend the token to a vehicle as long as the waiting time will not be exceeded. Note that no vehicle can be granted more than one token at a time. If a vehicle will possess a token and the router will resend the token due to a lost assumption, then the token will be revoked by default.

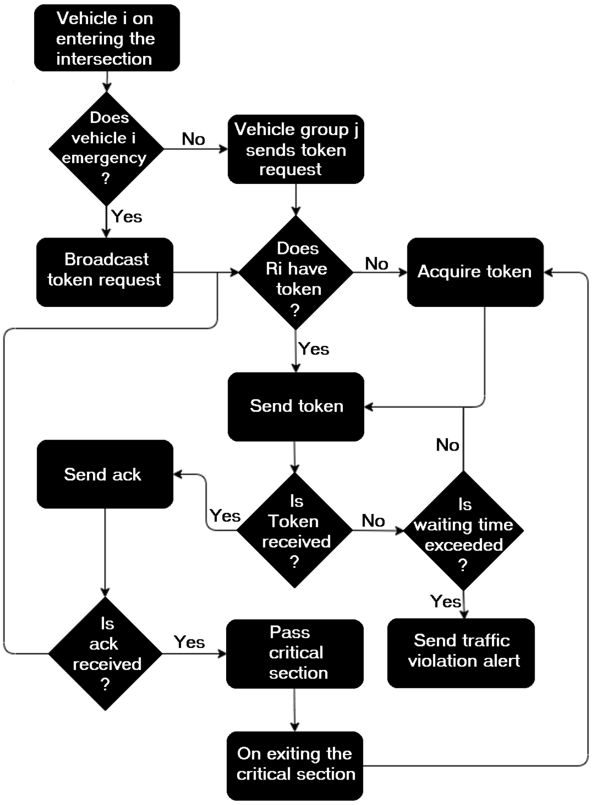


FIGURE 5. **Flowchart of MEAMAS.**

1) OPERATION OF REQUEST

Unless if the vehicle is an ambulance or a fire truck or police car, whereby vehicle *i* sends token request by setting $e = 1$. The rest vehicles send token request by setting $e = 0$. A vehicle that sends a request by setting $e = 1$ usually broadcasts the request to all the routers. When the routers receive a broadcast request from vehicle *i*, the following items resulted.

- If $R_{i,j,k,or\ l}$ has token, $R_{i,j,k,or\ l}$ sends token to vehicle *i*.
- If $R_{i,j,k,l}$ does not have token, $R_{i,j,k,or\ l}$ acquires token from a vehicle (on their lane) that is in hold of it.
- $R_{i,j,k,or\ l}$ sends token to vehicle *i*.

On the other hand, when any other vehicle that sends a request by setting $e = 0$ enters the queue area. It switches from IDLE to WAITING and sends a token request to its router. When a router R_i receives the request by the vehicle, the following items are resulted.

- R_i adds *i* to its queue.
- If R_i has the token, R_i sends the token to vehicle $group_j$, where vehicle $group_j$ is the group on top of R_i's queue,

and this is performed up to the number of request received by R_i.

- If R_i does not have token, R_i broadcasts token request to its neighbors.
- If any of its neighbors $R_{j,k,or\ l}$ has token and counter is greater than or equal to mp, then the token is sent to R_i; where counter holds the number of group already granted a lock and mp is the minimum number of group that should be granted a lock at a time.
- If any of the neighbors has the token and counter less than mp but the lane is empty, the neighbor with the token sends the token to R_i.
- If any of the neighbors has the token and counter less than mp and the lane is not empty, then R_i is being delayed for some time in such a way that counter will be greater than or equal to mp. Afterward, the token will be sent to R_i.

If there is no vehicle on lane number 1 or lane number 2 of a particular router, the router considers only the lane with vehicle as a group and allows vehicles to pass the critical section one after the other.

2) OPERATION OF PASSING

If $e = 1$, the routers send the token to vehicle i if they are in hold of it, else the routers send the token to vehicle i after acquiring it from any vehicle in the intersection that is in hold of it. Vehicle i proceeds to pass the critical section afterward. On the other hand, if $e = 0$, when the token is being received by the router R_i and the following cases are observed.

- R_i sends token to $group_j$, where $group_j$ represents the vehicle group on top of its queue.
- Two vehicles at lane 1 and 2 of $group_j$ sends *ack* to R_i.
- If R_i receives *ack* from both vehicle at lane 1 and 2, R_i continues serving vehicles to send token.
- Else R_i enters a waiting state to wait for *ack* from vehicle(s) at $group_j$.
- If still R_i did not receive *ack* from one or both vehicle at $group_j$, then R_i sends token again to vehicles lane in which *ack* is not received.
- If R_i receives *ack* from both vehicle at lane 1 and 2, then R_i continues serving vehicles to send token. Otherwise, R_i sends stop to vehicles to which lane it did not receive *ack*.
- R_i sends traffic violation alert to the control center and R_i continues its operation of serving vehicles to send token.

Vehicles that will send *ack* to R_i then proceed to pass the CS.

3) OPERATIONS OF RELEASE

This is the same whether the emergency parameter is set to 0 or 1. After a vehicle (i or j) passes the critical section, the vehicle or vehicles in case of a group sends back (release) the token to its router R_i or the router it received the token from $R_{i,j,k,or\ l}$ in the case of an emergency vehicle.

Algorithm 3 MEAMAS (Mutual Exclusion Algorithm Based on MAS)

```
1:  Give token to one of the router.
2:  Initialize $cnt$ and $mp$. CoBegin // for a vehicle $i$ On entering the monitoring area:
3:  if vehicle $i$ is ambulance, fire truck, police car, or similar kind then
4:      vehicle $i$ broadcast token request to all routers ($R_{i,j,k,l}$, $e = 1$).
5:      if $R_{i,j,k,or\ l}$ has token then
6:          $R_{i,j,k,or\ l}$ sends token to vehicle $i$.
7:      end if
8:      if $R_{i,j,k,l}$ does not have token then
9:          $R_{i,j,k,or\ l}$ acquires token from a vehicle (on their lane) in hold of the token.
10:         $R_{i,j,k,or\ l}$ sends token to vehicle $i$.
11:     end if
12: end if
13: vehicle $group_i$ is in a WAITING state.
14: if vehicle $group_i$ is not associated with any router. then
15:     vehicle $group_i$ sends token request to the router in its lane ($R_i$, $e = 0$).
16:     vehicle $group_i$ waits for response from $R_i$.
17: end if
    On Receiving token request($R_i$, $i$) from $group_i$
18: $R_i$ puts $group_i$ on its queue.
19: if $R_i$ has token then
20:     $R_i$ sends token to vehicle $group_j$ on top of its queue.
21:     vehicle $group_j$ sends $ack$ from $j$, lane 1, and lane 2.
22:     if $R_i$ receives $ack$ from $j$, lane 1, and lane 2 then
23:         continue.
24:     end if
25:     if $R_i$ is not received $ack$ from $j$, lane 1, OR lane 2 then
26:         $R_i$ waits
27:         if $R_i$ is not received $ack$ from $j$, lane 1, OR lane 2 then
28:             flag = unack_lane
29:             $R_i$ sends token to $j$, lane = flag.
30:             Vehicle $j$ sends $ack$ from $j$ and flag.
31:             if $R_i$ receives $ack$ from $j$ and lane = flag then
32:                 continue.
33:             end if
34:         end if
35:         $R_i$ sends stop to $i$ and flag.)
36:         $R_i$ sends traffic violation alert of $i$ and flag.)
37:         Continue.
38:     end if
39:     Increase $cnt$ by 1.
40: end if
41: if $R_i$ does not have token then
42:     $R_i$ sends token request to its neighbours $\{R_{j,k,l}\}$.
43:     if $R_{j,k,or\ l}$ has token AND $cnt$ greater than or equal to $mp$ then
44:         $R_{j,k,or\ l}$ sends token to $R_i$.
45:     end if
46:     if $R_{j,k,or\ l}$ has token AND $cnt$ less than $mp$ AND the lane is empty then
47:         $R_{j,k,or\ l}$ sends token to $R_i$.
48:     end if
49:     if $R_{j,k,or\ l}$ has token AND $cnt$ less than $mp$ AND the lane is not empty then
50:         Delay $R_i$ for some time.
51:         $R_{j,k,or\ l}$ sends token to $R_i$.
52:         Repeat from step 19.
53:     end if
54: end if
    On exiting the intersection
55: Vehicle $group_j$ sends token to $R_i$.
56: $R_i$ removes vehicle $group_j$ from its queue.
    CoEnd return
```

VI. RESULTS AND DISCUSSION

In this section, we discussed outputs of our proposed algorithms followed by various discussion, comparison with the existing approaches, simulation results based on message cost, statistical tests, our findings as well as observation, and future study.

A. HARDWARE SPECIFICATION

No hardware framework was used for the token exchange, instead the token is assumed to be exchanged across different processes. An 8 GB RAM HP 64-bit workstation with an Intel Core i5-7200U CPU utilizing Windows 10 Pro was used throughout our experiments to evaluate various models.

B. EXPERIMENTAL SETUP

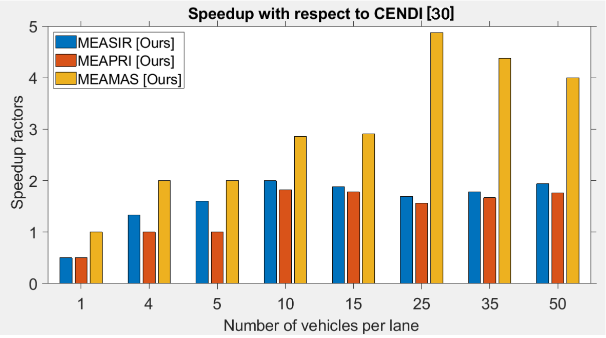
The assumption of Fig. 1 was that all vehicles with different numbers arrive at the intersection at different times. In algorithms of MEASIR and MEAPRI, the moment vehicle number 1 arrives, that vehicle will be enqueued to router 1's queue and router 1 will send a token request if it is not in hold of the token. The same scenario is applied to vehicle number 2, except that vehicle number 2 communicates with router 2. But when the two vehicles numbered 3 one at router 3's and the other at router 4's intersection arrives at the intersection, after being enqueued to the queue of their associated routers. In MEASIR, one of the routers (either router 3 or 4) will be selected randomly and be allowed to send a token request first. In MEAPRI, the router with the higher priority as set by the priority setting mechanism will be allowed to send the token request first. We have implemented MEASIR, MEAPRI, and MEAMAS in Java with an assumption that all vehicles have arrived at the intersection and the CS is empty. So, the execution time is the average wait time that takes all vehicles on the lane to pass the intersection. As only one token is issued at a time in both MEASIR and MEAPRI, the calculation of average wait time should be a serious issue for them. It is also a great deal to them for a comparative study of average wait times.

C. PRACTICAL RESULTS DEEMING NON-EMERGENCY

1) EXECUTION TIME

We have implemented our proposed algorithms and CENDI [30] in Java with an assumption that all vehicles have arrived at the intersection and the critical section is empty. The results obtained for their execution times (average wait times) in seconds with different number of vehicles on the lane have been shown in Table 1 and its associated Fig. 6(a). When the execution time of MEASIR and MEAPRI (algorithms that use routers) are compared with that of CENDI [30], the algorithm of CENDI [30] tends to perform better than both MEASIR and MEAPRI with less number of vehicles on the lane. But as the number of vehicles on the lane increases, both MEASIR and MEAPRI tend to perform better than CENDI [30]. But MEAMAS always works better than CENDI [30]. The MEAMAS showed the best average performance among MEASIR, MEAPRI, and CENDI [30].

(a) Execution time (average wait time).



(b) Speedup factors.

FIGURE 6. Performance of various algorithms without considering emergency cases.

2) SPEEDUP FACTORS

Table 1 and its associated Fig. 6(b) demonstrate the calculated speedup factors for MEASIR, MEAPRI, and MEAMAS with respect to CENDI [30]. The speedup factors of MEASIR and MEAPRI are very close to each other. MEAMAS showed the maximum speedup factor close to 5, but the average speedup factor approximates to 3. But without a shadow of doubt, we can conclude that our proposed algorithms performed better than that of CENDI [30] in general.

D. PRACTICAL RESULTS CONSIDERING EMERGENCY

1) EXECUTION TIME

We have compared the execution time of MEASIR, MEAPRI, and MEAMAS with other emergency algorithms such as EMEV [35], EVSP [37], and DTLS [36]. Table 2 and its associated Fig. 7(a) show their execution times (average times) in seconds. It has been observed that initially EMEV [35] performed better than MEASIR and MEAPRI with less number of emergency vehicles, but with an increasing number of the emergency vehicles per lane, both MEASIR and MEAPRI tend to outperform EMEV [24]. Yet, MEAMAS became the best performative algorithm.

2) SPEEDUP FACTORS

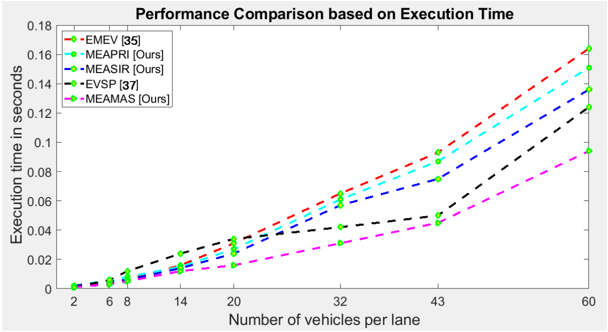
Table 1 and its associated Fig. 7(b) indicate the calculated speedup factors for MEASIR, MEAPRI, and MEAMAS with respect to DTLS [36]. If there were five normal vehicles

TABLE 1. Execution time (eTime) of our algorithms and speedup with respect to CENDI [30].

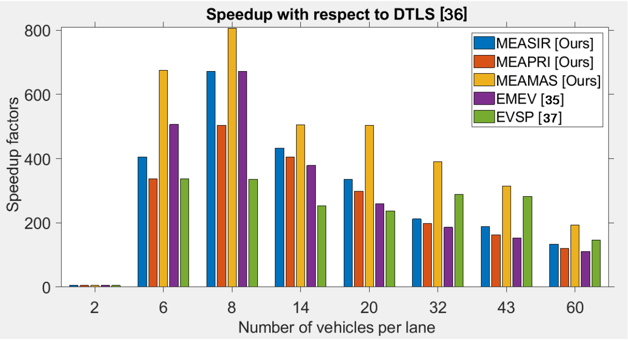
Number of normal vehicle(s) per lane	MEASIR		MEAPRI		MEAMAS		CENDI [30]
	eTime	Speedup	eTime	Speedup	eTime	Speedup	eTime
1	0.002	0.500	0.002	0.500	0.001	1.000	0.001
4	0.003	1.333	0.004	1.000	0.002	2.000	0.004
5	0.005	1.600	0.008	1.000	0.004	2.000	0.008
10	0.010	**2.000**	0.011	**1.818**	0.007	2.857	0.020
15	0.017	1.882	0.018	1.778	0.011	2.909	0.032
25	0.046	1.696	0.050	1.560	0.016	**4.875**	0.078
35	0.059	1.779	0.063	1.667	0.024	4.375	0.105
50	0.097	1.938	0.107	1.757	0.047	4.000	0.188
Average Performance							
19	0.030	1.591	0.033	1.385	0.014	**3.002**	0.055

TABLE 2. Results of (normal + emergency) vehicles per lane comparison based on their execution times (average wait times) in seconds.

Number of vehicles per lane	Our Proposed Algorithms						Existing Algorithms				
	MEASIR		MEAPRI		MEAMAS		EMEV [35]		EVSP [37]		DTLS [36]
	eTime	Speedup	eTime	Speedup	eTime	Speedup	eTime	Speedup	eTime	Speedup	eTime
1+1 = 2	0.002	0.500	0.002	0.500	0.001	1.000	0.001	1.000	0.001	1.000	0.001
4+2 = 6	0.005	405.4	0.006	337.8	0.003	675.7	0.004	506.8	0.006	**337.8**	2.027
5+3 = 8	0.006	**672.2**	0.008	**504.1**	0.005	**806.6**	0.006	**672.2**	0.012	336.1	4.033
10+4 = 14	0.014	433.1	0.015	404.2	0.012	505.3	0.016	378.9	0.024	252.6	6.063
15+5 = 20	0.024	336.2	0.027	298.9	0.016	504.3	0.031	260.3	0.034	237.3	8.069
25+7 = 32	0.057	212.3	0.061	198.4	0.031	390.4	0.065	186.2	0.042	288.2	12.103
35+8 = 43	0.075	188.5	0.087	162.5	0.045	314.1	0.093	152.0	0.050	282.7	14.134
50+10 = 60	0.136	133.7	0.151	120.4	0.094	193.4	0.164	110.9	0.124	146.6	18.180
Average Performance											
24	0.040	297.7	0.045	253.4	0.026	**423.8**	0.048	283.5	0.037	235.3	8.076



(a) Execution time (average wait time).



(b) Speedup factors.

FIGURE 7. Performance of various algorithms by considering emergency cases.

and three emergency vehicles, then the maximum speedup factors of MEASIR, MEAPRI, and MEAMAS were 672, 504, and 806, respectively. MEAMAS showed the maximum average speedup with respect to DTLS [36]. Explicitly, our proposed algorithms performed better than that their alternative algorithms. In brief, from the obtained results in Table 1, we can infer that MEASIR and MEAPRI have an approximate performance with a slight margin between them. Although it can be seen that the algorithm which allows single request at a time outperforms that with a priority with a very little margin in some cases. This is due to the resources wasted to determine which router should be given a priority in order for a vehicle in its lane to pass the intersection. But Figs. 6 and 7 suggest that the performance of MEAMAS is better than that of MEASIR and MEAPRI.

E. THEORETICAL COMPUTATIONAL COMPLEXITY

As any vehicle arrives at the lane, the vehicle sends a token request to its associated router. Communication between the vehicle and the router is processed in the form of a queue, resulting in a complexity of $(n - 1)$, where n represents the number of all vehicles in a particular intersection. If the router of a vehicle is not holding the token, the router will send a token request to other routers to get the token and pass it to the queue of waiting vehicles. The time complexity between the routers is $(4 - 1)$, as only 4 routers are required. Consequently, the complexity to complete the overall process of a car passing is $(n - 1)(4 - 1) = 3n - 3$. Henceforth, the computational complexity of MEASIR or MEAPRI is $O(n)$. On the other hand, MEAMAS has a time complexity of $4n/2$, which leads to $O(n)$. Table 3 gives information of theoretical computational complexity for various algorithms.

TABLE 3. Theoretical computational complexity of miscellaneous algorithms.

Type	Algorithms	Complexity
Ours	MEASIR	$O(n)$
	MEAPRI	$O(n)$
	MEAMAS	$O(n)$
Existing	CENDI [30]	$O(n^2)$
	EMEV [35]	$O(n)$
	EVSP [37]	$O(n)$
	DTLS [36]	$O(n^3)$

Although MEASIR, MEAPRI, MEAMAS, EMEV [35], EVSP [37] have the same limiting behavior, MEAMAS demonstrates the minimum practical execution time and henceforth maximum speedup factor.

F. SIMULATION RESULTS BASED ON MESSAGE COST

Table 4 and its associated Fig. 8 demonstrate the message cost with different numbers of vehicles on the lane obtained from our simulation results for our proposed algorithms as compared to CENDI [30], EMEV [35], EVSP [37], and DTLS [36]. An increase in the traffic volume has a direct effect on the message cost for EMEV [35] and DTLS [36]. As the traffic volume increases, the message cost increases simultaneously from 4 up to 16. On the other hand, our proposed algorithms need to exchange only 3 messages (REQUEST, PASSING, and RELEASE) in cases whereby the vehicle's router is in hold of the token at the time of the request and at most 4 messages in a situation whereby the associated router will need to send a token request to acquire token from routers on other lanes. Although our proposed MEAMAS is more efficient than MEASIR and MEAPRI, it has comparatively high message complexity. In the best case, MEAMAS exchanges four different messages (REQUEST, ACK, PASSING, and RELEASE). But in the worst case, it exchanges five different messages (REQUEST, WAIT, ACK, PASSING, and RELEASE). Nevertheless, it still has a better message complexity than MEASIR, CENDI [30], EMEV [35], and DTLS [36].

G. MULTIPLE COMPARISON WITH STATISTICAL TESTS

How can it be possible to show that one algorithm is better than its alternatives? It is possible statistically. Usually, multiple comparisons with a control algorithm are applied to statistically demonstrate that one algorithm is better than its alternatives in areas related to computer science [61]. The key concept of applying the non-parametric tests [62] includes that they can deal with probabilistic as well as non-probabilistic methods without any imposing any restriction. We have considered the execution time in TABLE 2 as well as simulation results in TABLE 4 for conducting tests for multiple comparisons along with a set of post-hoc procedures to compare a control algorithm with others (i.e., $1 \times N$ comparisons) and to perform all possible pairwise comparisons (i.e., $N \times N$ comparisons). For these purposes, we have used the open-source statistical software applications from the University of Granada [63].

1) MISCELLANEOUS NONPARAMETRIC TESTS

In the case of $1 \times N$ comparisons, the post-hoc procedures consist of Bonferroni-Dunn's [64], Holm's [65], Hochberg's [66], Hommel [67] and Hommel and Bernhard's [68], Holland's [69], Rom's [70], Finner's [71], and Li's [72], procedures; whereas in the case of $N \times N$ comparisons, they make up of Nemenyi's [73], Shaffer's [74], and Bergmann and Hommel's [75] procedures. In the case of Bonferroni-Dunn's procedure [64], the performance of two algorithms is considerably divergent if the corresponding mean of rankings is at least as large as its discriminating divergence. A better one is Holm's procedure [65], which examines in a consecutive manner all hypotheses ordered based on their *p-values* from inferior to superior. All hypotheses for which the *p-value* is less than α divided by the number of algorithms minus the number of a successive step are rejected. All hypotheses having larger *p-values* are upheld. Holm's procedure [65] adjusts α in a step-down manner. Similarly, both Holland's [69] and Finner's [71] procedures adjust α in a step-down method. But the Hochberg's procedure [66] functions in the opposite direction to Holland's procedure [69]. It compares the largest *p-value* with α, the next largest with $\alpha/2$, and so on until it encounters a hypothesis it can reject. The Rom [70] proposed a modification to Hochberg's step-up procedure [66] to enhance its power. In turn, Li [72] suggested a two-step rejection procedure.

2) MULTIPLE COMPARISON NONPARAMETRIC TESTS

Table 5 shows the average ranking computed by using Friedman [76], Friedman's aligned rank test [77], and Quade [78] non-parametric tests. To achieve the test results Friedman [76], Friedman's aligned rank test [77], and Quade [78] non-parametric tests are applied to the average number of estimated execution times of various algorithms. The aim of applying Friedman [76], Friedman's aligned rank test [77], and Quade [78] non-parametric tests is to find whether there are significant differences among various algorithms considered over a given set of data [78], [79]. These tests give a ranking of the algorithms for each individual dataset, i.e., the best performing algorithm receives the highest rank of 1, the second-best algorithm gets the rank of 2, etc. The mathematical equations and further explanation of the non-parametric procedures of Friedman [76], Friedman's aligned rank test [77], and Quade [78] can be found in Quade [78] and Westfall and Young [79].

Based on the obtained results in Table 5, MEAMAS is the best performing algorithm of the comparison, with an average ranking of 1.1875, 22.1875, and 1.0417 for Friedman test [76], Friedman's aligned rank test [77], and Quade test [78], respectively. This indicates that MEAMAS gives great performance for the solution of intersection problems. Friedman statistic considered reduction performance

TABLE 4. **Simulation results of algorithms based on their message complexity.**

Traffic Volume	Our Proposed Algorithms			Existing Algorithms			
	MEASIR	MEAPRI	MEAMAS	CENDI [30]	EMEV [35]	EVSP [37]	DTLS [36]
8	4	3	3	4	5	4	7
16	6	3	4	4	7	4	7
24	8	4	4	4	8	3	9
32	10	4	4	4	9	4	11
40	12	4	4	5	11	4	11
48	14	4	4	5	13	4	11
56	15	4	4	5	15	4	15
64	16	4	4	5	16	4	15

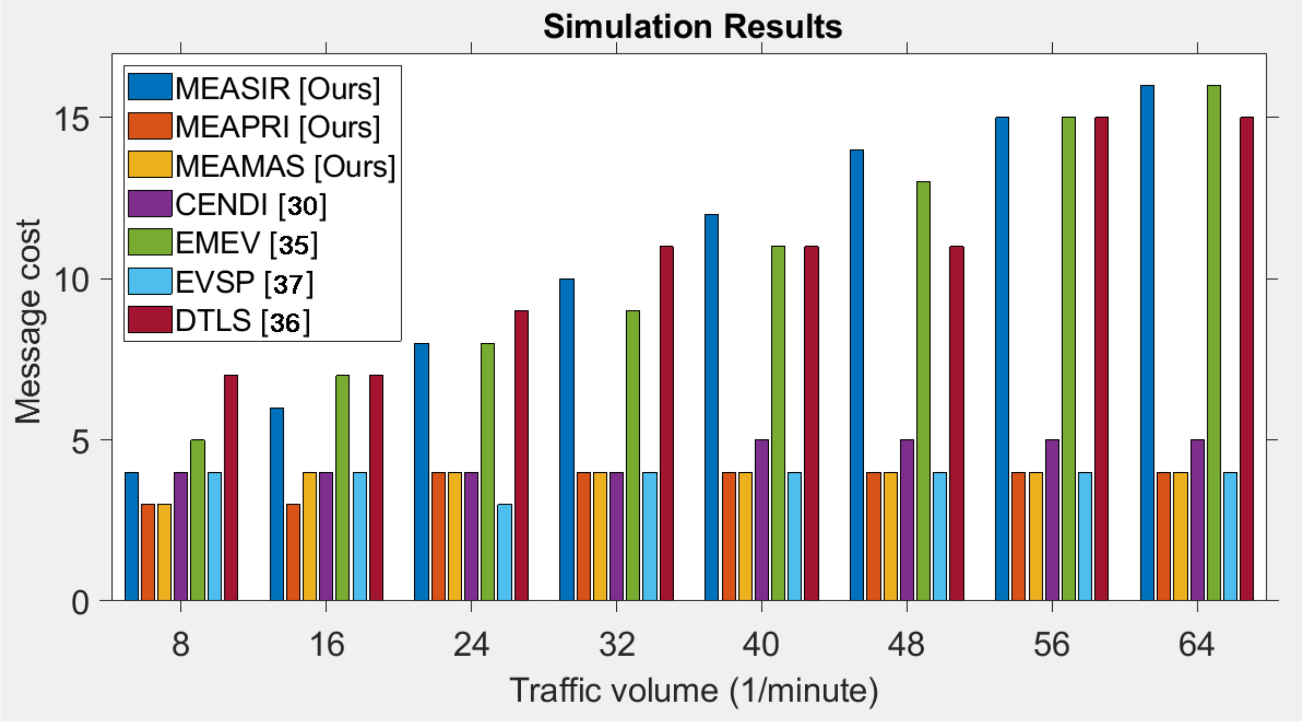


FIGURE 8. **Plotting of simulation results from Table 4.**

(distributed according to chi-square with 6 degrees of freedom) of 29.517857. Aligned Friedman statistic considered reduction performance (distributed according to chi-square with 6 degrees of freedom) of 34.346703. Quade statistic considered reduction performance (distributed according to F-distribution with 6 and 42 degrees of freedom) of 11.869278. The *p-values* computed through Friedman statistic, aligned Friedman statistic, and Quade statistic are 0.000048534165, 0.000005765853, and 0.000000095478, respectively.

3) POST-HOC PROCEDURES FOR 1 × *N* COMPARISONS

In the case of 1 × *N* comparisons, the post-hoc procedures consist of Bonferroni-Dunn's [64], Holm's [65], Hochberg's [66], Hommel [67] and Hommel and Bernhard's [68], Holland's [69], Rom's [70], Finner's [71], and Li's [72] procedures. In these statistical analysis tests, multiple comparison post-hoc procedures considered for comparing the control algorithm MEAMAS with the other algorithms. The results are shown by computing *p-values* for each comparison. Tables 6 depicts obtained *p-values* using the ranks computed by the Friedman [76], Friedman's aligned rank test [77], and Quade [78] non-parametric tests, respectively. Based on the computed results, all tests show significant improvements of MEAMAS over MEASIR, MEAPRI, CENDI [30], EMEV [35], EVSP [37], and DTLS [36] for all the post-hoc procedures considered. Besides this, Li's [72] procedure does the greatest performance, reaching the lowest *p-values* in the comparisons.

4) POST-HOC PROCEDURES FOR *N* × *N* COMPARISONS

In the case of *N* × *N* comparisons, the post-hoc procedures consist of Nemenyi's [73], Shaffer's [74], as well as Bergmann and Hommel's [75] procedures. Table 7 presents 21 hypotheses of equality among the 6 different algorithms and the *p-values* achieved. Using the level of significance

TABLE 5. **Average rankings using the non-parametric statistical procedures, statistics, and *p-values*.**

Ranking	Algorithms	Multiple Comparison Tests		
		Friedman [76]	Friedman's aligned rank [77]	Quade [78]
1	**MEAMAS [Ours]**	**1.1875**	**22.1875**	**1.0417**
2	MEASIR [Ours]	3.1250	24.1250	2.8194
3	EVSP [37]	3.7500	24.7500	3.2917
4	EMEV [35]	3.7500	24.7500	4.3056
5	MEAPRI [Ours]	4.1875	25.1875	3.8889
6	CENDI [30]	5.5625	26.5625	5.7778
7	DTLS [36]	6.4375	51.9375	6.8750
Various Statistics		29.517857	34.346703	11.869278
p-value		0.000048534165	0.000005765853	0.000000095478

TABLE 6. **Adjusted *p-values* for various tests considering MEAMAS [Ours] as control method.**

Tests	Algorithms	Unadjusted *p-values*	$1 \times N$ post-hoc procedures							
			One/Two step procedures		Step-down procedures			Step-up procedures		
			p_{Bonf} [64]	p_{Li} [72]	p_{Holm} [65]	p_{Hol} [69]	p_{Finn} [71]	p_{Hoch} [66]	p_{Hom} [67]	p_{Rom} [70]
Friedman [76]	DTLS [36]	0.000001	0.000007	0.000001	0.000007	0.000007	0.000007	0.000007	0.000007	0.000007
	CENDI [30]	0.000051	0.000307	0.000055	0.000256	0.000256	0.000153	0.000256	0.000256	0.000243
	MEAPRI [Ours]	0.005479	0.032871	0.005874	0.021914	0.021735	0.010927	0.021914	0.021914	0.020896
	EMEV [35]	0.017672	0.106034	0.018704	0.053017	0.052085	0.026391	0.035345	0.035345	0.035345
	EVSP [37]	0.017672	0.106034	0.018704	0.053017	0.052085	0.026391	0.035345	0.035345	0.035345
	MEASIR [Ours]	0.072849	0.437093	0.072849	0.072849	0.072849	0.072849	0.072849	0.072849	0.072849
F. al. rank [77]	DTLS [36]	0.000264	0.001585	0.001404	0.001585	0.001584	0.001584	0.001585	0.001585	0.001507
	CENDI [30]	0.591615	3.549691	0.759048	2.958076	0.988641	0.931889	0.812198	0.812198	0.812198
	MEAPRI [Ours]	0.712960	4.277759	0.791508	2.958076	0.993212	0.931889	0.812198	0.812198	0.812198
	EMEV [35]	0.753343	4.520059	0.800454	2.958076	0.993212	0.931889	0.812198	0.812198	0.812198
	EVSP [37]	0.753343	4.520059	0.800454	2.958076	0.993212	0.931889	0.812198	0.812198	0.812198
	MEASIR [Ours]	0.812198	4.873188	0.812198	2.958076	0.993212	0.931889	0.812198	0.812198	0.812198
Quade [78]	DTLS [36]	0.005459	0.032757	2.382607	0.032757	0.032313	0.032313	0.032757	0.032757	0.031147
	CENDI [30]	0.024073	0.144441	0.038396	0.120367	0.114710	0.070496	0.120367	0.120367	0.114468
	EMEV [35]	0.120020	0.720119	0.166021	0.480080	0.400359	0.225635	0.397101	0.360060	0.397101
	MEAPRI [Ours]	0.175030	1.050177	0.224994	0.525089	0.438545	0.250697	0.397101	0.397101	0.397101
	EVSP [37]	0.283835	1.703011	0.320091	0.567671	0.487108	0.330091	0.397101	0.397101	0.397101
	MEASIR [Ours]	0.397101	2.382607	0.397101	0.567670	0.487108	0.397101	0.397101	0.397101	0.397101

$\alpha = 0.05$, Nemenyi's [73] procedure rejects those hypotheses that have an unadjusted *p-value* $\leq$ 0.002381. Similarly, Holm's [65] procedure rejects those hypotheses that have an unadjusted *p-value* $\leq$ 0.002778. Shaffer's [74] procedure rejects those hypotheses that have an unadjusted *p-value* $\leq$ 0.002381. Bergmann and Hommel's [75] procedure rejects hypotheses of MEASIR vs. DTLS [36], MEAMAS vs. CENDI [30], and MEAMAS vs. DTLS [36]. On the other hand, considering $\alpha = 0.10$, Nemenyi's [73] procedure rejects those hypotheses that have an unadjusted *p-value* $\leq$ 0.004762. Similarly, Holm's [65] procedure rejects those hypotheses that have an unadjusted *p-value* $\leq$ 0.005882. Shaffer's [74] procedure rejects those hypotheses that have an unadjusted *p-value* $\leq$ 0.004762. Bergmann and Hommel's [75] procedure rejects hypotheses of MEASIR vs. DTLS [36], MEAPRI vs. MEAMAS, MEAMAS vs. CENDI [30], and MEAMAS vs. DTLS [36].

In sum and substance, based on the aforementioned experimental, simulation, and statistical test results, it would be easy to make an explicit conclusion that the MEAMAS outperforms over MEASIR, MEAPRI, CENDI [30], EMEV [35], EVSP [37], and DTLS [36]. Intuitively speaking, it is observed that the performance of our proposed MEAMAS surpasses those of other distributed mutual exclusion algorithms for solving intersection problems.

H. OUR FINDINGS AND OBSERVATION

The findings and observations on our proposed algorithms have been discussed below.

1) FAIRNESS

Our proposed MEASIR, MEAPRI, and MEAMAS have shown more fairness [80], [81] among vehicles than that of CENDI [30]. A vehicle might be able to pass the intersection as it arrives there before other vehicles, which are on the same lane and have been there already. But in this case, vehicles on the same lane are given permission to use the token in FIFO order. As a result, we can say that our proposed algorithms ensure fairness among vehicles in the same lane.

2) BETWEEN ROUTERS

A router can generate the token for its own group only if another router is not in hold of the token.

3) PERFORMANCE METRIC

Our proposed MEASIR, MEAPRI, and MEAMAS do not violate any rule of the mutual exclusion algorithms. The performance of each algorithm can be measured by both execution times and the number of messages. For a vehicle to enter a critical section, there are two types of messages needed: (i) Message between vehicles and the routers; and (ii) Message between routers.

TABLE 7. Adjusted *p-values* for tests for multiple comparisons among all methods.

Index	Hypothesis	$N \times N$ post-hoc procedures				
		Unadjusted	Nemenyi [73]	Holm [65]	Shaffer [74]	Bergmann [75]
1	MEAMAS [Ours] vs. DTLS [36]	0.000001	0.000025	0.000025	0.000025	0.000025
2	MEAMAS [Ours] vs. CENDI [30]	0.000051	0.001073	0.001022	0.000767	0.000767
3	MEASIR [Ours] vs. DTLS [36]	0.002164	0.04544	0.041112	0.032457	0.032457
4	MEAPRI [Ours] vs. MEAMAS [Ours]	0.005479	0.11505	0.098614	0.082178	0.060264
5	EMEV [35] vs. DTLS [36]	0.012841	0.269667	0.218301	0.192619	0.141254
6	EVSP [37] vs. DTLS [36]	0.012841	0.269667	0.218301	0.192619	0.141254
7	MEAMAS [Ours] vs. EMEV [35]	0.017672	0.371118	0.265084	0.265084	0.15905
8	MEAMAS [Ours] vs. EVSP [37]	0.017672	0.371118	0.265084	0.265084	0.15905
9	MEASIR [Ours] vs. CENDI [30]	0.024028	0.504581	0.31236	0.265084	0.240277
10	MEAPRI [Ours] vs. DTLS [36]	0.037243	0.782093	0.446911	0.409668	0.240277
11	MEASIR [Ours] vs. MEAMAS [Ours]	0.072849	1.529826	0.801337	0.801337	0.509942
12	CENDI [30] vs. EMEV [35]	0.093338	1.960088	0.933375	0.933375	0.560025
13	CENDI [30] vs. EVSP [37]	0.093338	1.960088	0.933375	0.933375	0.560025
14	MEAPRI [Ours] vs. CENDI [30]	0.203017	4.263359	1.624137	1.42112	0.812068
15	MEASIR [Ours] vs. MEAPRI [Ours]	0.325271	6.830691	2.276897	2.276897	2.276897
16	CENDI [30] vs. DTLS [36]	0.417887	8.775626	2.507322	2.507322	2.276897
17	MEASIR [Ours] vs. EMEV [35]	0.562834	11.819509	2.814169	2.814169	2.276897
18	MEASIR [Ours] vs. EVSP [37]	0.562834	11.819509	2.814169	2.814169	2.276897
19	MEAPRI [Ours] vs. EMEV [35]	0.685443	14.394313	2.814169	2.814169	2.276897
20	MEAPRI [Ours] vs. EVSP [37]	0.685443	14.394313	2.814169	2.814169	2.276897
21	EMEV [35] vs. EVSP [37]	1	21	2.814169	2.814169	2.276897

4) MESSAGE COMPLEXITY

Algorithms of MEASIR, MEAPRI, MEAMAS, EMEV [35], EVSP [37] have the same theoretical computational complexity of $O(n)$; nevertheless, CENDI [30] functions with $O(n^2)$ and DTLS [36] owns $O(n^3)$. Although MEASIR, MEAPRI, MEAMAS, EMEV [35], EVSP [37] behave the same asymptotic behavior, both MEAPRI and MEAMAS minimize their message complexity in a better way. As a result, MEAPRI and MEAMAS outperform over MEASIR, CENDI [30], EMEV [35], EVSP [37], and DTLS [36].

5) DEADLOCK IS NOT POSSIBLE

Deadlock is impossible because in MEASIR, only one router is in allowed to request for the token from other routers at a time. For MEAPRI, a priority is given to all the routers and only the router with a higher priority will be given permission to use the token at a particular time. MEAMAS is based on MAS to enhance the efficiency of both MEASIR and MEAPRI. It is also free from deadlock.

6) ABNORMAL SITUATION HANDLED

In the very extreme case, for example, if two or more vehicles would fall an accident at one of the lanes in the intersection. How would this case be handled by routers or so? The routers of MEASIR and MEAPRI do not handle such cases. But since the mutual exclusion algorithm does not allow simultaneous access to the CS which is the shared resource in this case, it is not possible that an accident will occur. However, assuming a vehicle spoils or an accident occurred (which is very unlikely), MEAMAS handles such event by sending a traffic violation signal to the control center.

I. FUTURE WORKS

The MEASIR allows only a single instruction to be sent at a time. While this is controlled in the proposed MEASIR algorithm, it might not be the case in real-world situations as multiple vehicles might arrive at the intersection at the same time. The MEAPRI algorithm addresses this issue by using a priority-setting mechanism to determine which router should be granted the token based on their priority. This allows concurrent requests of the token at the same time. Nevertheless, the resources wasted in determining the priority increases the turnaround time of the MEAPRI algorithm when compared to the MEASIR algorithm. In the case of message lost, the MEAMAS algorithm uses a counter to wait for a certain amount of time and resend the token only if the counter ends and no ack has been received. While this solves the problem, it might as well increase the turnaround time of the algorithm if the situation occurs concurrently. Thus, future work would address these issues by using a different approach to set priority in MEAPRI and automatically detect a message lost in MEAMAS rather than waiting for the counter so as to achieve an optimal result. We made the assumption that all vehicles have arrived at the intersection and the CS is empty. Thus, the execution time was the average wait time that took all vehicles on the lane to pass the intersection. Nevertheless, the average queue time has been highlighted as one of the limitations of our current work, since the time it might take a vehicle to pass the intersection varies for each vehicle depending on the arrival time. We did not calculate the average queue time of our algorithms. As we used the FIFO order, the average queue time will be definitely different for all vehicles depending on the time the vehicle arrived at the intersection. Future work would also include this average queue time. In addition, carefully optimized code can always give a better performance [82]. The codes of MEASIR, MEAPRI, and MEAMAS are not optimized. Thus code would be optimized by using manual and software optimization techniques [83] to obtain an optimal execution time of each algorithm. Furthermore, the work in this article is limited to single-intersection interaction. Our future work includes multi-intersection interaction (e.g., [84]–[86]) for smart traffic purposes. Our proposed algorithms in this article will be adopted for those purposes.

VII. CONCLUSION

We introduced three distinct algorithms namely MEASIR, MEAPRI, and MEAMAS to reduce traffic congestion. Our deadlock-free algorithms differed in important ways from traditional approaches used for smart traffic lights and intersection controllers. In doing so, it was able to effectively tackle practical challenges like computational complexity, potentially making it better suited for real-time applications. Although these concepts were not new, they had been applied well to this specific problem and incremental improvements had been made. Besides, the use of the MAS framework allowed them to overcome key challenges through its request-reply feature and also relaxed assumptions, henceforth, generalizing the method for real-life situations. Technical justifications were used in most cases to back up qualitative claims about those three algorithms, and these were quantified whenever possible. Furthermore, the conducted rigorous statistical analysis helped boosting confidence in the simulation results and confirmed their statistical significance. This analysis also helped interpreting the insights in a better way and shed some light on why certain algorithms performed better than others. The complete statistical test showed that MEAMAS became the best performative algorithm among its alternative algorithms of MEASIR, MEAPRI, CENDI [30], EMEV [35], DTLS [36], and EVSP [37]. Another validation of our algorithms was that the simulated and theoretical computational complexities were approximately agreed, both in terms of execution time and message cost.

We described both the conceptual idea behind the algorithms and their practical implementation via communication infrastructure. The combination of flowcharts, graphs, and pseudo-code clearly conveyed how all three algorithms worked in practice and made for a comprehensive explanation. We conducted a quite thorough and exhaustive review of intelligent traffic algorithms. In addition, covering a variety of solution methods, we also highlighted the key strengths and weaknesses of existing approaches. Consequently, motivating our work well with a view to achieving lower computational burden, lower waiting times, better fairness, better accuracy, better precision, and so on. Yet, future work will include the upgrading of our current algorithms for multi-intersection interaction along with code optimization.

ACKNOWLEDGMENT

Shehu thanks to Dr. Elif Haytaoğlu for offering a distributed system course. The authors thank Dr. Mnaouer Kachout for assisting some software applications. They also thank the anonymous reviewers for their appreciative and constructive comments on the draft of this article.

REFERENCES

[1] S. M. M. Ali, J. C. Augusto, and D. Windridge, "A survey of user-centred approaches for smart home transfer learning and new user home automation adaptation," *Appl. Artif. Intell.*, vol. 33, no. 8, pp. 747–774, Jul. 2019.
[2] M. H. Sharif, I. Despot, and S. Uyaver, "A proof of concept for home automation system with implementation of the Internet of Things standards," *Periodicals Eng. Natural Sci.*, vol. 6, no. 1, pp. 95–106, 2018.
[3] G. M. Toschi, L. B. Campos, and C. E. Cugnasca, "Home automation networks: A survey," *Comput. Standards Interfaces*, vol. 50, pp. 42–54, Feb. 2017.
[4] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 630–638, Sep. 2010.
[5] SCATS. (2019). *The Benchmark in Urban Traffic Control*. [Online]. Available: https://www.scats.com.au/
[6] VICS. (2019). *Vehicle Information and Communication System*. [Online]. Available: https://www.vics.or.jp/en/index.html
[7] VII. (2019). *Vehicle Infrastructure Integration*. [Online]. Available: http://www.vehicle-infrastructure.org/
[8] D. Zhao, Y. Dai, and Z. Zhang, "Computational intelligence in urban traffic signal control: A survey," *IEEE Trans. Syst., Man, Cybern., C (Appl. Rev.)*, vol. 42, no. 4, pp. 485–494, Jul. 2012.
[9] P. Lertworawanich, M. Kuwahara, and M. Miska, "A new multiobjective signal optimization for oversaturated networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 967–976, Dec. 2011.
[10] B. Zhou, J. Cao, X. Zeng, and H. Wu, "Adaptive traffic light control in wireless sensor network-based intelligent transportation system," in *Proc. IEEE 72nd Veh. Technol. Conf. Fall*, Sep. 2010, pp. 1–5.
[11] B. P. Gokulan and D. Srinivasan, "Distributed geometric fuzzy multiagent urban traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 714–727, Sep. 2010.
[12] P. La and S. Bhatnagar, "Reinforcement learning with function approximation for traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 412–421, Jun. 2011.
[13] E. Bingham, "Reinforcement learning in neurofuzzy traffic signal control," *Eur. J. Oper. Res.*, vol. 131, no. 2, pp. 232–241, Jun. 2001.
[14] M. H. Sharif and O. Gursoy, "Parallel computing for artificial neural network training using java native socket programming," *Periodicals Eng. Natural Sci.*, vol. 6, no. 1, pp. 1–10, 2018.
[15] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural networks for real-time traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 3, pp. 261–272, Sep. 2006.
[16] J. Qiao, N. Yang, and J. Gao, "Two-stage fuzzy logic controller for signalized intersection," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 41, no. 1, pp. 178–184, Jan. 2011.
[17] L. Zhao, X. Peng, L. Li, and Z. Li, "A fast signal timing algorithm for individual oversaturated intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 280–283, Mar. 2011.
[18] F. Galip, M. Caputcu, R. H. Inan, M. H. Sharif, A. Karabayir, S. Kaplan, M. Ozuysal, B. Sengoz, A. Guler, and S. Uyaver, "A novel approach to obtain trajectories of targets from laser scanned datasets," in *Proc. 18th Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dec. 2015, pp. 231–236.
[19] M. H. Sharif, "Particle filter for trajectories of movers from laser scanned dataset," in *Proc. 3rd Medit. Conf. Pattern Recognit. Artif. Intell. (MedPRAI)* (Communications in Computer and Information Science), vol. 1144. Cham, Switzerland: Springer, 2019, pp. 133–148. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-37548-5_11
[20] C. M. Institute. (2019). *P vs NP Problem*. [Online]. Available: https//:www.claymath.org
[21] M. H. Sharif, "A numerical approach for tracking unknown number of individual targets in videos," *Digit. Signal Process.*, vol. 57, pp. 106–127, Oct. 2016.
[22] B. van Arem, C. J. G. van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 429–436, Dec. 2006.
[23] F. McGinley, "An intersection control strategy for a short-headway P.R.T. network," *Transp. Planning Technol.*, vol. 3, no. 1, pp. 45–53, 1975.
[24] R. J. Caudill and J. N. Youngblood, "Intersection merge control in automated transportation systems," *Transp. Res.*, vol. 10, no. 1, pp. 17–24, Feb. 1976.
[25] T. Blackwell and J. Kaufman, "Response time effectiveness: Comparison of response time and survival in an urban emergency medical services system," *Academic Emergency Med., Official J. Soc. Academic Emergency Med.*, vol. 9, no. 4, pp. 288–295, Apr. 2002.

[26] O. Braun, "EMS system performance: The use of cardiac arrest timelines," *Ann. Emergency Med.*, vol. 22, no. 1, pp. 52–61, 1993.

[27] A. C. of Surgeons, *Atls, Advanced Trauma Life Support Program for Doctors*. Chicago, IL, USA: American College of Surgeons, 2008.

[28] J. Campbell, *International Trauma Life Support for Emergency Care Providers*. London, U.K.: Pearson, 2018, p. 12.

[29] X. Wu, S. Deng, X. Du, and J. Ma, "Green-wave traffic theory optimization and analysis," *World J. Eng. Technol.*, vol. 02, no. 03, pp. 14–19, 2014.

[30] W. Wu, J. Zhang, A. Luo, and J. Cao, "Distributed mutual exclusion algorithms for intersection traffic control," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 1, pp. 65–74, Jan. 2015.

[31] T.-T. Ngo, T. Huynh-The, and D.-S. Kim, "A novel VANETs-based traffic light scheduling scheme for greener planet and safer road intersections," *IEEE Access*, vol. 7, pp. 22175–22185, 2019.

[32] R. Wu, "GeoTAR: A map-based and traffic-aware VANET routing in urban scenario," in *Proc. IEEE 11th Int. Conf. Commun. Softw. Netw. (ICCSN)*, Jun. 2019, pp. 34–40.

[33] D. Markert, P. Parsch, and A. Masrur, "Analyzing the impact of probabilistic estimates on communication reliability at intelligent crossroads," in *Proc. 22nd Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2019, pp. 206–213.

[34] K. Raymond, "A tree-based algorithm for distributed mutual exclusion," *ACM Trans. Comput. Syst.*, vol. 7, no. 1, pp. 61–77, Jan. 1989.

[35] G. Singh, J. Singh, and A. Z. Richa, "Algorithm for effective movement of emergency vehicles from traffic control signal," *Int. J. Eng. Res. Appl.*, vol. 7, pp. 9–13, 2017.

[36] A. A. Krishna, B. A. Kartha, and V. S. Nair, "Dynamic traffic light system for unhindered passing of high priority vehicles: Wireless implementation of dynamic traffic light systems using modular hardware," in *Proc. IEEE Global Humanitarian Technol. Conf. (GHTC)*, Oct. 2017, pp. 1–5.

[37] R. Anil, M. Satyakumar, and A. Salim, "Emergency vehicle signal preemption system for heterogeneous traffic condition : A case study in trivandrum city," in *Proc. 4th Int. Conf. Intell. Transp. Eng. (ICITE)*, Sep. 2019, pp. 306–310.

[38] M. Gradinariu and S. Tixeuil, "Conflict managers for self-stabilization without fairness assumption," in *Proc. 27th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2007, p. 46.

[39] A. Housni and M. Trehel, "Distributed mutual exclusion token-permission based by prioritized groups," in *Proc. ACS/IEEE Int. Conf. Comput. Syst. Appl.*, Beirut, Lebanon, Jun. 2001, pp. 253–259.

[40] H. Xu, Y. Zhang, L. Li, and W. Li, "Cooperative driving at unsignalized intersections using tree search," 2019, *arXiv:1902.01024*. [Online]. Available: http://arxiv.org/abs/1902.01024

[41] U. M. Borghoff and J. H. Schlichter, *Computer-Supported Cooperative Work-Introduction to Distributed Applications*. Berlin, Germany: Springer, 2000.

[42] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, "A survey on content-oriented networking for efficient content delivery," *IEEE Commun. Mag.*, vol. 49, no. 3, pp. 121–127, Mar. 2011.

[43] M. Amad, A. Meddahi, and G. Vanwormhoudt, "A self-adaptive ALM architecture for P2P media streaming," in *Proc. Int. Conf. Protocol Eng. (ICPE) Int. Conf. New Technol. Distrib. Syst. (NTDS)*, Austin, TX, USA, Jul. 2015, pp. 1–6.

[44] G. Araniti, M. Condoluci, P. Scopelliti, A. Molinaro, and A. Iera, "Multicasting over emerging 5G networks: Challenges and perspectives," *IEEE Netw.*, vol. 31, no. 2, pp. 80–89, Mar. 2017.

[45] H. Hartenstein and K. P. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Commun. Mag.*, vol. 46, no. 6, pp. 164–171, Jun. 2008.

[46] M. Elhadef, "An adaptable inVANETs-based intersection traffic control algorithm," in *Proc. IEEE Int. Conf. Comput. Inf. Technol., Ubiquitous Comput. Commun., Dependable, Autonomic Secure Comput., Pervasive Intell. Comput.*, Oct. 2015, pp. 2387–2392.

[47] K. A. R. Bharath and P. B. Umesh, "An improved algorithm for distributed mutual exclusion by restricted message exchange in voting districts," in *Proc. Int. Conf. Inf. Technol.*, Dec. 2008, pp. 41–44.

[48] M. Maekawa, "A square root N algorithm for mutual exclusion in decentralized systems," *ACM Trans. Comput. Syst.*, vol. 3, no. 2, pp. 145–159, 1985.

[49] M. Haidar Sharif and C. Djeraba, "An entropy approach for abnormal activities detection in video streams," *Pattern Recognit.*, vol. 45, no. 7, pp. 2543–2561, Jul. 2012.

[50] X. Cheng and Z. Yang, "Distributed traffic signal control approach based on multi-agent," in *Proc. 6th Int. Conf. Fuzzy Syst. Knowl. Discovery*, Tianjin, China, 2009, pp. 582–587.

[51] H. Mu, J. Yu, and L. Liu, "Traffic signals control of urban traffic intersections group based on fuzzy control," in *Proc. 7th Int. Conf. Fuzzy Syst. Knowl. Discovery*, Aug. 2010, pp. 763–767.

[52] S. U. Park, B. K. Kim, S. C. You, and J. Y. Lee, "Group mutual exclusion algorithm for intersection traffic control of autonomous vehicle," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2017, pp. 1633–1636.

[53] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA J. Automatica Sinica*, vol. 3, no. 3, pp. 247–254, Jul. 2016.

[54] L. Qi, M. Zhou, and W. Luan, "Emergency traffic-light control system design for intersections subject to accidents," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 170–183, Jan. 2016.

[55] Y.-S. Huang, Y.-S. Weng, and M. Zhou, "Design of regulatory traffic light control systems with synchronized timed Petri nets," *Asian J. Control*, vol. 20, no. 1, pp. 174–185, Jan. 2018.

[56] W. C. Tchuitcheu, C. Bobda, and M. J. H. Pantho, "Internet of smart-cameras for traffic lights optimization in smart cities," *Internet Things*, vol. 11, Sep. 2020, Art. no. 100207.

[57] M. H. Sharif, "An eigenvalue approach to detect flows and events in crowd videos," *J. Circuits, Syst. Comput.*, vol. 26, no. 07, Jul. 2017, Art. no. 1750110.

[58] D. Saha, S. Roy, S. Bandyopadhyay, T. Ueda, and S. Tanaka, "A distributed feedback control mechanism for priority-based flow-rate control to support QoS provisioning in ad hoc wireless networks with directional antenna," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, Jun. 2004, pp. 4172–4176.

[59] H.-Y. Kung, C.-Y. Chen, and C.-I. Wu, "Differentiated object priority control mechanism for MPEG-4 streaming," *Comput. Commun.*, vol. 29, no. 9, pp. 1522–1537, May 2006.

[60] D. Nobayashi, Y. Fukuda, and T. Ikenaga, "A simple priority control mechanism for performance improvement of mobile ad-hoc networks," in *Proc. IEEE 36th Conf. Local Comput. Netw.*, Oct. 2011, pp. 167–170.

[61] H. Kusetogullari, M. H. Sharif, M. S. Leeson, and T. Celik, "A reduced uncertainty-based hybrid evolutionary algorithm for solving dynamic shortest-path routing problem," *J. Circuits, Syst. Comput.*, vol. 24, no. 05, Jun. 2015, Art. no. 1550067.

[62] B. Trawiński, M. Smętek, Z. Telec, and T. Lasota, "Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms," *Int. J. Appl. Math. Comput. Sci.*, vol. 22, no. 4, pp. 867–881, Dec. 2012.

[63] U. of Granada. (2020). *Soft Computing and Intelligent Information Systems*. [Online]. Available: https://sci2s.ugr.es/sicidm

[64] O. J. Dunn, "Multiple comparisons among means," *J. Amer. Stat. Assoc.*, vol. 56, no. 293, pp. 52–64, Mar. 1961.

[65] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandin. J. Statist.*, vol. 6, no. 2, pp. 65–70, 1979.

[66] Y. Hochberg, "A sharper bonferroni procedure for multiple tests of significance," *Biometrika*, vol. 75, pp. 800–803, Dec. 1988.

[67] G. Hommel, "A stagewise rejective multiple test procedure based on a modified bonferroni test," *Biometrika*, vol. 75, no. 2, pp. 383–386, 1988.

[68] G. Hommel and G. Bernhard, "A rapid algorithm and a computer program for multiple test procedures using logical structures of hypotheses," *Comput. Methods Programs Biomed.*, vol. 43, pp. 213–216, Jun. 1994.

[69] M. Holland, "An improved sequentially rejective Bonferroni test procedure," *Biometrics*, vol. 43, pp. 417–423, Jun. 1987.

[70] D. M. Rom, "A sequentially rejective test procedure based on a modified bonferroni inequality," *Biometrika*, vol. 77, no. 3, pp. 663–665, 1990.

[71] H. Finner, "On a monotonicity problem in step-down multiple test procedures," *J. Amer. Stat. Assoc.*, vol. 88, no. 423, pp. 920–923, Sep. 1993.

[72] J. (David) Li, "A two-step rejection procedure for testing multiple hypotheses," *J. Stat. Planning Inference*, vol. 138, no. 6, pp. 1521–1527, Jul. 2008.

[73] P. Nemenyi, "Distribution-free multiple comparisons," Ph.D. dissertation, Dept. Math., Princeton Univ., Princeton, NJ, USA, 1963.

[74] J. P. Shaffer, "Modified sequentially rejective multiple test procedures," *J. Amer. Stat. Assoc.*, vol. 81, no. 395, pp. 826–831, Sep. 1986.

[75] G. Bergmann and G. Hommel, "Improvements of general multiple test procedures for redundant systems of hypotheses," in *Multiple Hypotheses Testing*, E. S. P. Bauer and G. Hommel, Eds. Springer, 1988, pp. 100–115.

[76] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Amer. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, Dec. 1937.

[77] J. Hodges and E. Lehmann, "Rank methods for combination of independent experiments in analysis of variance," *Ann. Stat.*, vol. 33, pp. 482–497, 1962.

[78] D. Quade, "Using weighted rankings in the analysis of complete blocks with additive block effects," *J. Amer. Stat. Assoc.*, vol. 74, no. 367, pp. 680–683, Sep. 1979.

[79] P. Westfall and S. Young, *Resampling-Based Multiple Testing: Examples Methods for P-Value Adjustment*. Hoboken, NJ, USA: Wiley, 2004.

[80] S. Cantarell, A. K. Datta, and F. Petit, "Self-stabilizing atomicity refinement allowing neighborhood concurrency," in *6th Int. Symp. Self-Stabilizing Syst. (SSS)*, Jun. 2704. San Francisco, CA, USA, 2003, pp. 102–112.

[81] M. H. Karaata, "Self-stabilizing strong fairness under weak fairness," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 4, pp. 337–345, Apr. 2001.

[82] M. H. Sharif, A. Basermann, C. Seidel, and A. Hunger, "High-performance computing of $1/\sqrt{x_i}$ and $\exp(\pm x_i)$ for a vector of inputs xi on Alpha and IA-64 CPUs," *J. Syst. Archit.-Embedded Syst. Des.*, vol. 54, no. 7, pp. 638–650, 2008.

[83] M. H. Sharif, "High-performance mathematical functions for single-core architectures," *J. Circuits, Syst. Comput.*, vol. 23, no. 04, Apr. 2014, Art. no. 1450051.

[84] M. Hausknecht, T.-C. Au, and P. Stone, "Autonomous intersection management: Multi-intersection optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 4581–4586.

[85] F. Yan, M. Dridi, and A. E. Moudni, "Autonomous vehicle sequencing problem for a multi-intersection network: A genetic algorithm approach," in *Proc. Int. Conf. Adv. Logistics Transp.*, May 2013, pp. 215–220.

[86] N. Li, Y. Yao, I. Kolmanovsky, E. Atkins, and A. Girard, "Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections," 2019, *arXiv:1904.05423*. [Online]. Available: http://arxiv.org/abs/1904.05423

HARISU ABDULLAHI SHEHU (Graduate Student Member, IEEE) received the B.Sc. degree (Hons.) in computer engineering from Gediz University, Turkey, and the M.Sc. degree in computer engineering from Pamukkale University, Turkey. His M.Sc. thesis focused on sentiment analysis of Turkish Twitter texts. He is currently a Ph.D. Researcher with the Engineering and Computer Science Department, Victoria University of Wellington, New Zealand. His research interest includes algorithms and operation research.

MD. HAIDAR SHARIF was born in Jashore, Bangladesh, in 1977. He received the B.Sc. degree in electronics and computer science from Jahangirnagar University, Bangladesh, in 2001, the M.Sc. degree in computer engineering from the University of Duisburg–Essen, Germany, in 2006, and the Ph.D. degree in computer science from the University of Lille, Lille, France, in 2010. From January 2011 to January 2016, he worked as an Assistant Professor at University in İzmir, Turkey. From April 2016 to June 2017, he worked as an Assistant Professor at the International University of Sarajevo, Sarajevo, Bosnia and Herzegovina. From October 2017 to September 2018, he worked as an Associate Professor at International Balkan University, North Macedonia. He has been working at the University of Hail, Saudi Arabia, since November 2018. His research interests include applications of computer vision, brain–computer interface (BCI) technologies, computer architecture, and computational intelligence algorithms.

RABIE A. RAMADAN received the Ph.D. degree in computer engineering from Cairo University and Southern Methodist University (SMU), Dallas, TX, USA, in 2007. He is currently the Chair of the Computer Engineering Department, Hail University, Hail, Saudi Arabia. He is an Associate Professor with Cairo University, Cairo, Egypt, and Hail University. He is the Director of the Industrial Partnership Program (CISCO, Oracle, and Microsoft), College of Computer Science and Engineering, Hail University. He is the Founder of the FabLab and the Center of Programming and Applications at Hail University sponsored by the Ministry of Education, Saudi Arabia. He is also the Founder of Triple Tech expert house sponsored by Hail University. He is the author of more than 125 articles in the field of big data, cloud computing, the Internet of Things (IoT), and computational intelligence. He has served as the General Chair, the Program Committee Chair, and a TPC Member for many of the conferences and journals, including the Web of Science and IEEE TRANSACTIONS journals. He has served as the Co-Chair of the International Conference on Recent Advances in Computer Systems (RACS-2015) and the Second National Computing Colleges Conference (NC3 2017) held at Hail University. He was also the General Chair of the International Conference on New Computer Science and Engineering Trends (NCSET2020). He was also the Chair of the ACM Programming Competition in Saudi Arabia associated with NC3 2017 Conference. He is also a Co-Founder of the IEEE Computational Intelligence, Egypt Chapter.

• • •