

CHAPTER 3

METHODOLOGY

3.1 Introduction

In this chapter, the proposed methodology to achieve the research objectives and the used methods are defined in detail. The traffic light management system has two entities: standard operations/values and decision making entity. In this thesis, the main focus will be on the decision making entity while the standards' properties and values will be assumed unchangeable as will be illustrated later.

After listing down the assumptions that the work was based upon, a short overview about the traffic light general design and controlling mechanisms has been written followed by the suggested approach. Finally, an explanation of how the real traffic light management system has been mapped into the simulated one has been presented. At the end of the chapter, the reader will be ready to see how the developed approach has been tested and see the results which are explained in the next chapter.

The traffic light decision maker has two decisions to make; the next phase green lights and the next phase time. Both of these decision making algorithms will be illustrated within this chapter after describing how they were performed by other approaches. To understand how the traffic light management system is looked at, all of the features and properties are defined in Table 3.1 and Table 3.2 followed by the considered assumptions.

Table 3.1: Terms of the Input Features and Properties Used Within This Thesis

PROPERTY NAME	PROPERTY DESCRIPTION
L_D	L_D (Refers to the on-Duty flag) is a flag that can take the values (1 or 0) to indicate whether a special vehicle is on duty or not.
Load/Weight	Two terms frequently used in the developed method definition and they refer to the effect of a specific factor or all of the factors on the road priority. Some factors have positive effects on the load priority, while others have a negative effect.
L_P	L_P (Stands for Lane priority) is a factor which represents the maximum existing level of a vehicle's importance on the road.
L_T	L_T stands for Total Load. It is the total load of a lane after combining the effect of all of the road factors.
L_W	L_W (Refers to the Loaded lane Waiting time) is the factor which refers to the total count of seconds since the first vehicle arrived at a red traffic light.
V_C	V_C (Stands for Confirmed Vehicles) is a road status factor which represents the number of vehicles arriving randomly (poison) to the queuing area.
V_{lg}	V_{lg} (Stands for Lagging vehicles) is a road status factor that can be calculated by subtracting V_C from V_N ($V_{lg} = V_N - V_C$).
V_N	V_N (Stands for the Number of Vehicles) is a road status factor which refers to the total number of vehicles entering a lane.
V_{NQB}	V_{NQB} (Stands for Number of Vehicles Queuing Behind) is a combined road status factor that can be determined by adding the lagging vehicle factor with the two confirmed vehicle factors of the roads behind sending their vehicles to this road.
V_{TNN}	V_{TNN} (Stands for Total Number of Vehicles on the Next road) is a secondary road status factor that represents the number of vehicles entering the front road which receive the outgoing vehicles of this road as an input.

Table 33.2: Terms of the Output Features and Properties Used Within This Thesis

Property Name	Property Description
Direction's Green Phase Time	How long a phase would take until the traffic light changes into another phase.
Direction's Average Queue Length	An average number resulting from dividing the summation of one direction's queue length every one second divided by the number of seconds within an hour of time.
Direction's Average Waiting Time	A number resulting from dividing the summation of one direction's waiting time every one second divided by the number of seconds within an hour of time.
Intersection Capacity	The number of vehicles crossing through an intersection during one hour of time.
Intersection's Average Queue Length	A number which can be determined by dividing the summation of all of the directions' average queue length within an hour by the number of directions in that intersection.
Intersection's Average Waiting Time	A number which can be determined by dividing the summation of all of the directions' average waiting time within an hour by the number of directions in that intersection.

Two sets of assumptions have been used as the basis when creating the system; those are categorized into a geometrical setup of the vehicles' flow. The geometrical setup assumptions included:

1. All the intersections are as shown in Figure 3.1.
2. All the intersections are flat (Slope = 0).
3. Each approach flowed in three directions (Left, Through, and Right).
4. Each lane had one direction only. Whilst each direction had at least one lane.
5. The vehicles on a lane follow its direction.
6. A vehicle would decide on which lane it would queue 150 meters before the queuing place and stay driving over it till it leaves it.
7. No accidents available on the road.
8. All vehicles following the driving rules strictly and no traffic offences.
9. The left side lane was considered a slip lane (No traffic Light needed).
10. All the lanes were complete lanes. No short lanes were considered.
11. Each green phase had two not-intersected green lights and six red lights.
12. Each approach was equipped with a well-established VANET network infrastructure as shown in Figure 3.2. And Road Side Equipment was separated by 150 meters. A suggested addition to the VANET infrastructure was that each RSE was connected to a per-lane-vehicle detection belt.
13. All vehicles were equipped to achieve VANET communication with the VANET network infrastructure.

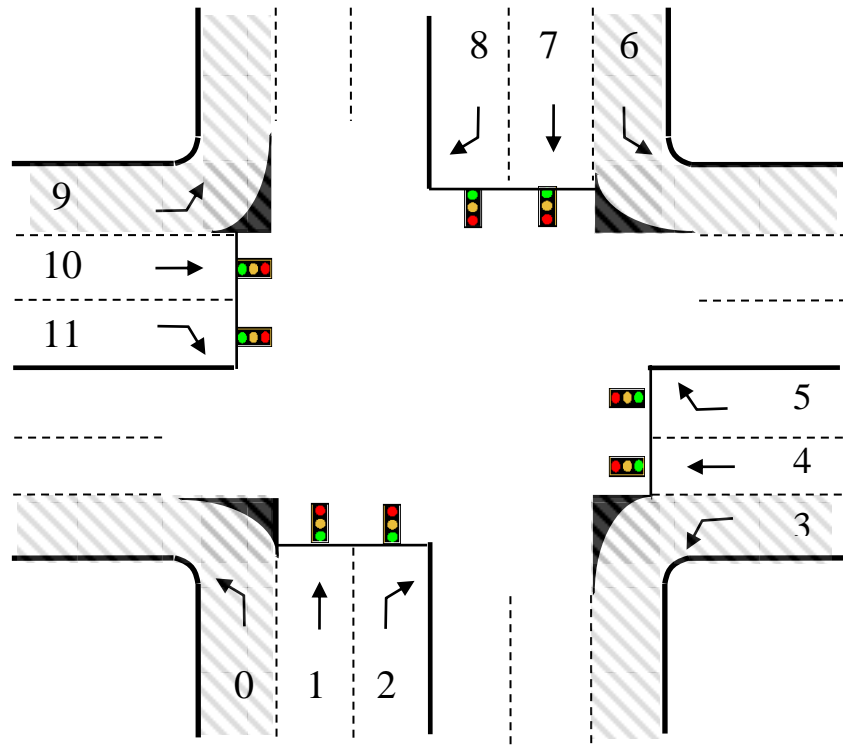


Figure 3.1: Standard Four-Legs Intersection [95, 98, and 112]

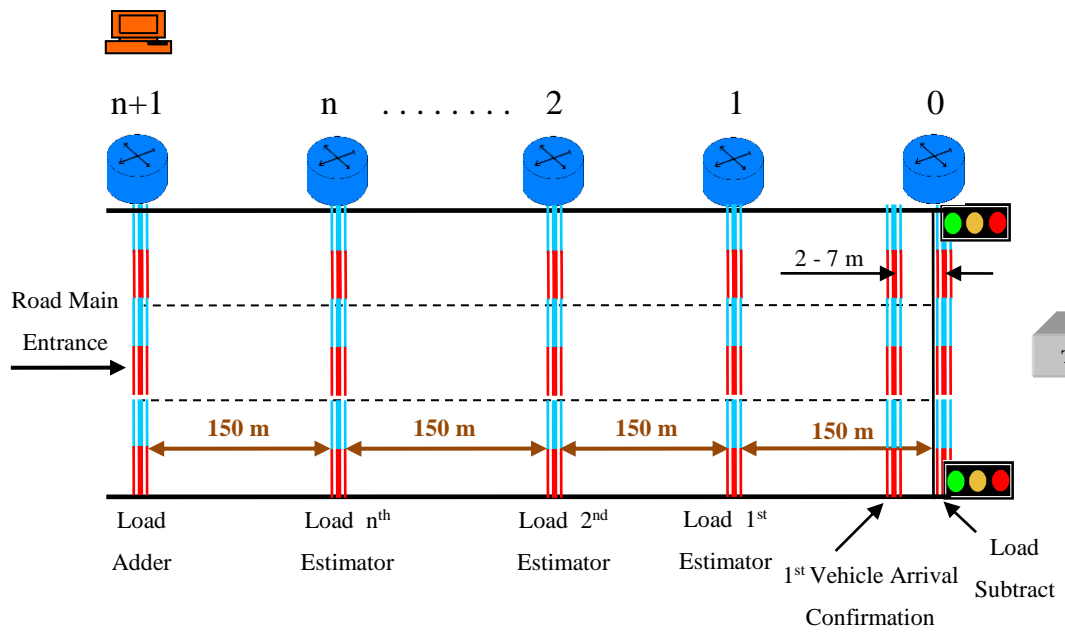


Figure 3.2: VANET Network Infrastructure with Additional Per-Lane-Vehicle Detection Belts

Whilst, the vehicle flow assumptions are as follow:

1. Any vehicle entering the road would leave the road.
2. Vehicle speeds were considered as similar.
3. All vehicles were of a passenger vehicle size (no Commercial vehicles) and all on four wheels (no motorcycles.)

3.2 The Developed System Specification

The approach suggested in this thesis consists of algorithms which need a specific network architecture setup. The developed algorithm does the road's data collection first, and then it determines the traffic light phase decision based upon those collected data. This section will illustrate in detail both the road's data collection system and the decision making algorithms.

3.2.1 Road's Data Collecting System (RDCS)

The road status can be represented through several factors, such as the vehicles queue lengths, queue waiting time, and emergency vehicle existence. Nevertheless, in real scenarios, more data from the surrounding roads and intersections would add some more accuracy to the decision of the traffic light controller, as will be illustrated in the following sections.

3.2.1.1 VANET Network Architecture Setup

The Network setup suggested in Figure 3.2 collected the status data about the road and the nearby roads. Those collected data were sent to the Traffic Light Controller (TLC) which was placed at the intersection to help make an optimum decision about the next green phase.

The setup of the Road's Data Collection System (RDCS) can be described as a set of Road Side Equipment (RSE) that was sited beside the road, each connected to a sensor belt which was set on or under the road pavement. The job of the sensor belt depended on its position. The job of the belt at the road's main entrance was to detect the total number of vehicles arriving in each lane; it was named the Load Adder. Whilst, the traffic light stopping line's belt job was to detect the number of vehicles leaving each lane; it was named the Load Subtract. Six meters before the stopping line, another belt was positioned to confirm the first vehicle's arrival on each lane. The last type of belt was the Load Estimators. The main duty of the Load Estimators was to detect the queue length on each lane. The number of the Load Estimators depended on the length of the street, as they were separated by 150 meters. An important point to highlight about the Load Estimators is that only one Load Estimator was performing the queue length detection per lane at a time, and that depended on the queue length. For example, considering the case when the queue length on the upper lane, in Figure 3.2, was less than 30 vehicles whilst the queue length of the middle lane had more than 30 vehicles, then the first Load Estimator would detect the queue length for the upper lane, whilst the second load estimator would do it for the middle lane. The coordination of the belts' work was the duty of the RSEs.

The arrival of any vehicle to a road's entrance would start a side to side communication between the vehicles and the road side equipment which would help in the emergency vehicle's arrival detection. As soon as the hand shaking completed between the two sides, a report would be sent by the vehicle to the RSE holding some information about the vehicle, including the type of vehicle and its ON-DUTY flag.

After collecting the data from along the roads of an intersection, they would be compiled into 8 variables per direction. Finally, those directions' eight variables would be transferred to its last destination; that was the Traffic Light Controller of that intersection where the decision was being made.

3.2.1.2 Queue Length Detection

The queue length detection is the job of the Load Estimator belts, shown in Figure 3.2, which are coordinated by the RSEs. Each estimation belt has several sensors, two sensors for each lane. RSEs might turn ON or OFF the sensors depending on the lanes' queue lengths. For example, in Figure 3.3, the 1st Load Estimator had two parts to it (A-B and E-F) ON because their lane queue lengths were less than 30 vehicles while the C-D part was OFF because its lane queue length was higher than 30 vehicles. The system shut down the C-D part on the 1st Load Estimator and turned it ON at the 2nd Load Estimator which could detect a 61-vehicle queue length. RSEs used three messages, listed in Table 3.3, to achieve the queue length detection and make sure of the successful delivery of the queue length updated data to the TLC.

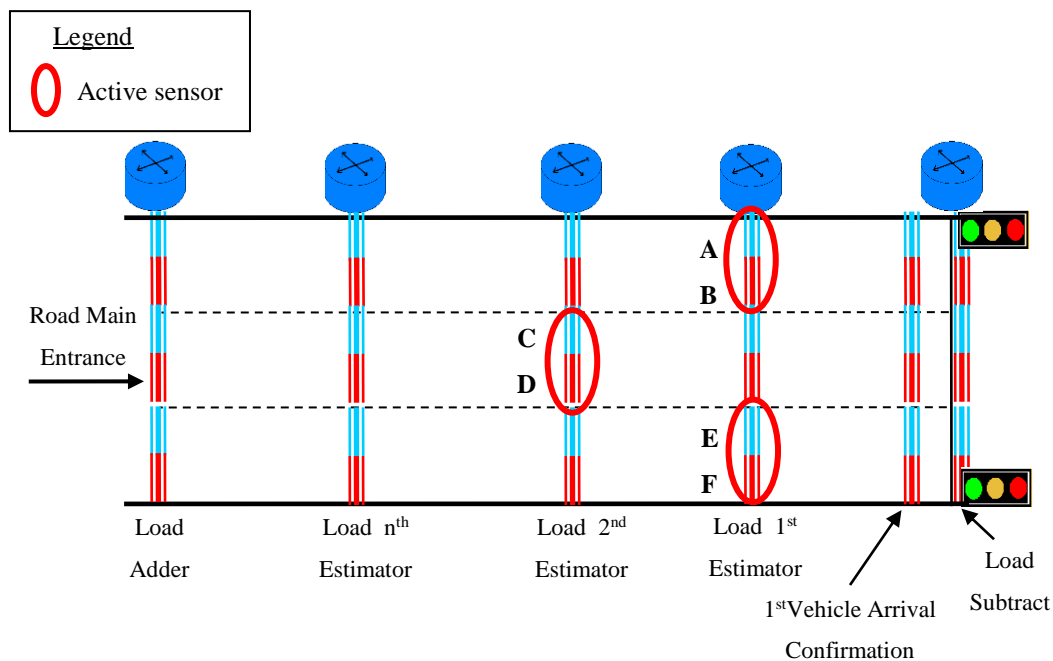


Figure 3.3: Vehicles Detection (Sensor Belt Example)

Table 3.3: Queue Length Detection Required Messages

Message name	Description
MtAD	MtAD is a timed message that is created and sent by the RSEs (those are activated to act as the Load Estimator) every 1 second to the last RSE on the road.
MtAE	MtAE is an event-based message that is created and sent by an RSE either to the next or behind-RSE to turn ON the queue length sensors for the lane(s).
MtAF	MtAF is an event-based message that is created and sent by the last RSE backward to the RSEs on that road when a traffic light changes from green to red light.

In Figure 3.4., each RSE with a vehicle detection belt has at least two vehicle step counters for each lane. The vehicle arrival confirmation area (between one Load Estimator and another) can be occupied by a maximum of 30 vehicles. If the vehicles' arrival confirmation area of any lane gets filled by vehicles, then the currently active RSE responsible for queue length determination sends MtAE to the behind-RSE to activate its sensors for that specific lane and make it responsible for the queue length count.

Whilst, if an RSE (Not the first Load Estimator) has been elected to act as the queue length counter for a specific lane, and the number of vehicles goes under half of its maximum capacity, then it sends MtAE to the next RSE to give it back the responsibility of confirming the vehicle's arrival.

In Figure 3.5., each traffic light has a green time, during which some vehicles leave the queue. That count is detected by the Last Road detector for each lane. That count is sent by the last RSE on the road within the MtAF message back to all of the RSEs on that road. Each RSE subtracts the amount of vehicles from the total count of the responsive counter.

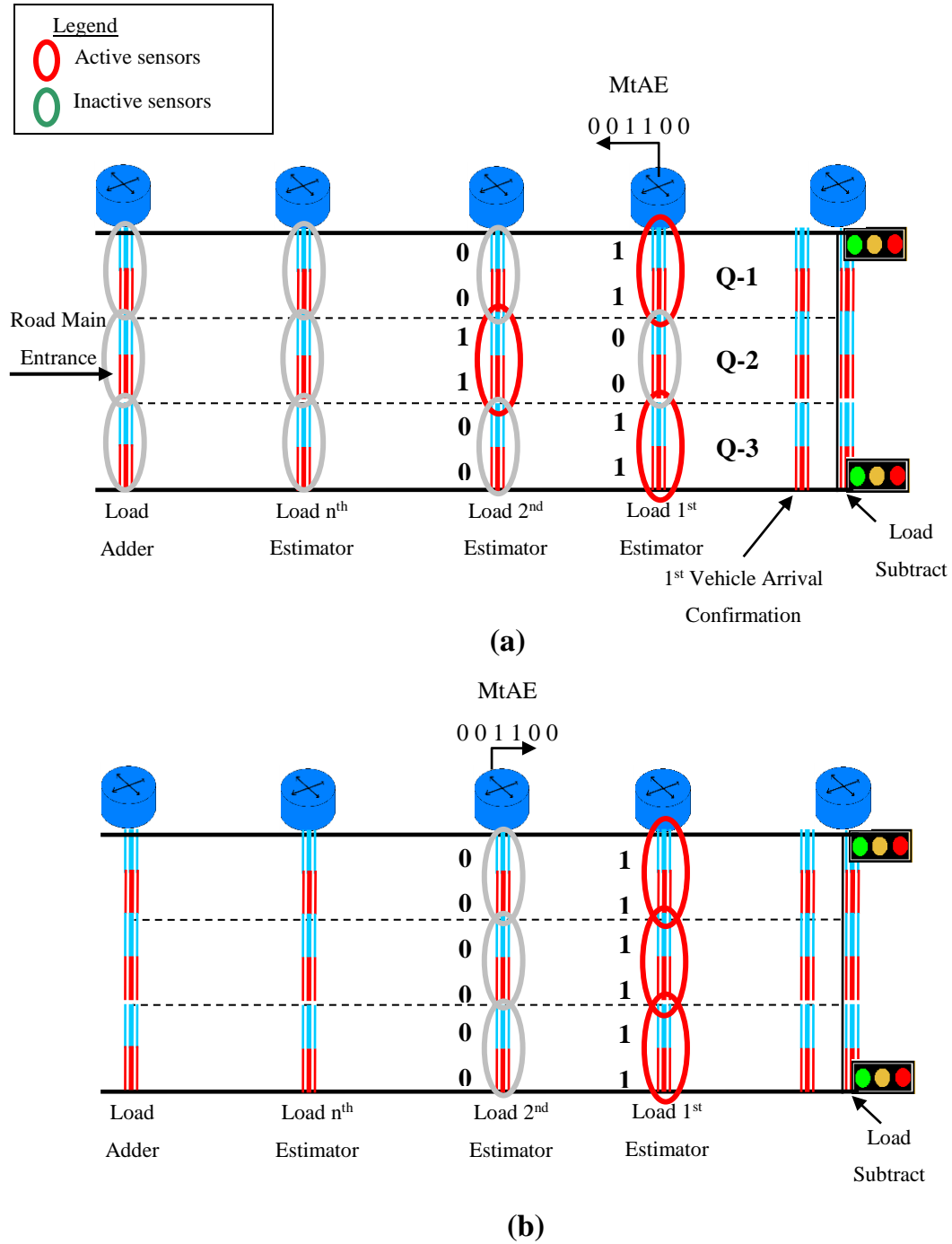
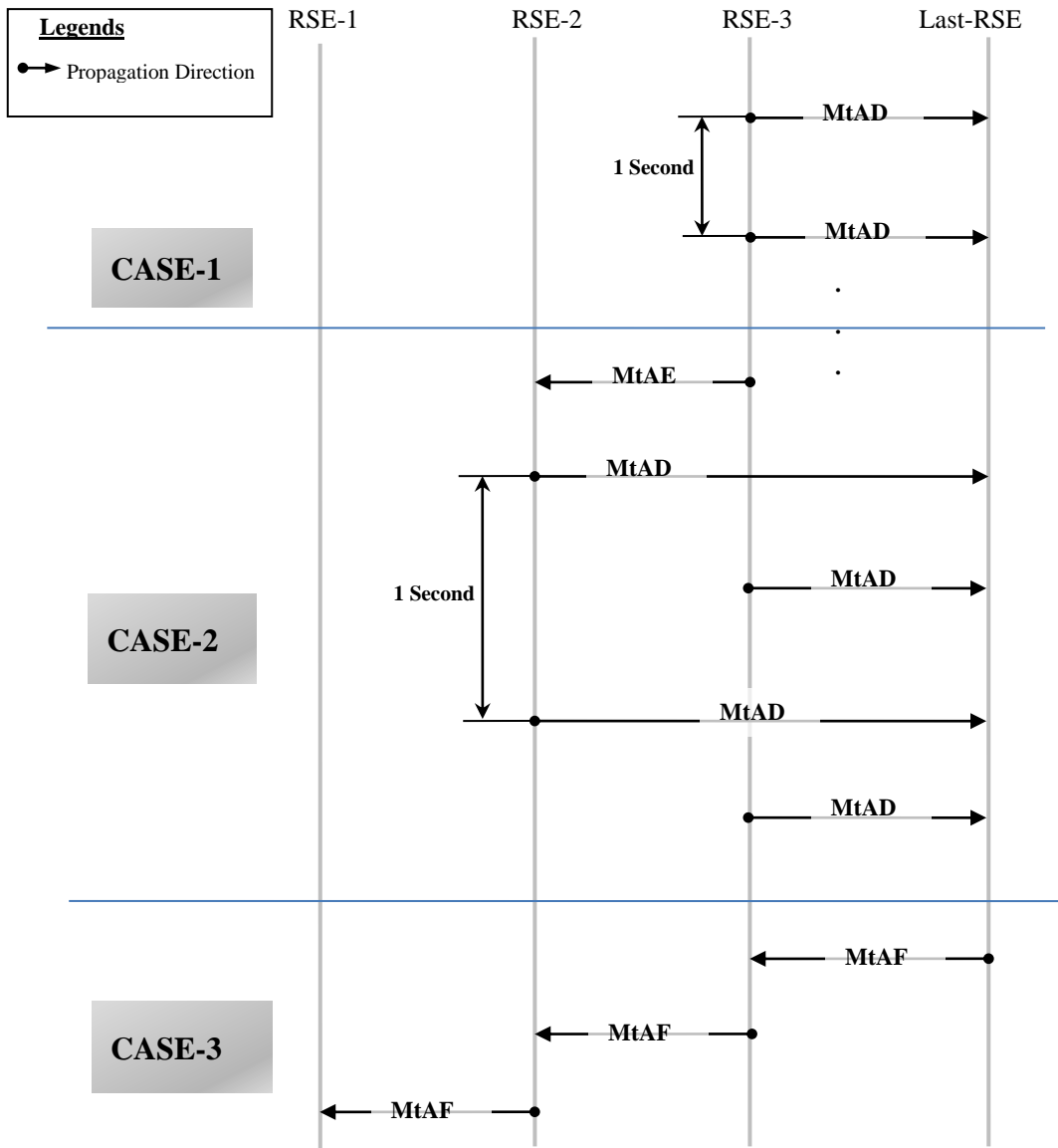


Figure 3.4: Message Type MtAE Function Examples: (a) Queue-2 Length Is More Than 30 Vehicles; (b) Queue-2 Length Is Less Than 30 Vehicles.



Note: Each device needs some time to process the incoming messages to make a decision on what to do next. The exact amount of time is not important for this research and this is why it has been neglected.

Figure 3.6: Sequence Diagram for the “Relation between MtAD, MtAE, and MtAF”

3.2.1.3 Emergency Vehicle detection

Every vehicle approached a road entrance; a hand shaking procedure took place to connect the vehicle to the VANET infrastructure. See Figure 3.7. The hand shaking procedure was initiated by the RSEs which continuously broadcast a hello message, named MtAA, as Illustrated in Table 3.4. When a vehicle received MtAA from an RSE carrying a new road ID with a new RSE sequence number, it would reply with a half-reply message, named MtAB, to the hello message as it had not yet confirmed the existence of the vehicles on that road. Whilst, when a vehicle received MtAA from an RSE carrying the same road ID with a new RSE sequence number, then it would send back a full reply message, named MtAC, to the hello message as its existence on that specific road had been confirmed. MtAC carrying the vehicles type is one of its data fields. As soon as an RSE received MtAC from an emergency vehicle, it would send a unicast alert message, named MtAG, to the last RSE on the road declaring the emergency vehicle existence.

Table 3.4: The Messages Used When Hand Shaking and During Emergency Vehicle Detection

Message Name	Message Description
MtAA	MtAA is a timed broadcast message sent by RSEs repeatedly to initiate a hand shake connection with the nearby vehicles. MtAA carries the Road_ID and the RSE_ID with a time stamp.
MtAB	MtAB is a reply message created by a vehicle when it receives MtAA coming from an RSE that carries a new road ID with a new RSE sequence number. MtAB contains an instant report about the vehicle that originated the message. The report has information that includes the Vehicle's VANET Device ID (VVD_ID) and some of the vehicle's other information (such as Vehicle's registration number). Those are configured when buying the car by some authorized people. These data cannot be changed by normal users; this is why it is called VVD_Fixed_Info. Another data field is the time stamp that refers to the time when the message was created. The last field (VVD_Status) holds the instant status of the vehicle, such as the speed at which the vehicle is being driven and if the vehicle is a special type of vehicle other than the ON-Duty flag.
MtAC	MtAC is a reply to a second MtAA coming from a second RSE residing on the same road as the first MtAA. The RSE sequence number of the second MtAA should be more than the RSE sequence number held by the first MtAA by at least one. MtAC carries similar information as MtAB.
MtAG	MtAG is an event based message carrying the news of the emergency case existence and what level it is from any RSE to the last RSE. See Table 3.5.

Whilst a vehicle was driving on a road, it might have received many MtAAs coming from many different RSEs residing on different roads. The reason behind sending MtAB first, and then MtAC was to confirm the direction of the vehicle (MtAC would be sent only if the vehicle had received at least two MtAA messages coming from the same road's RSEs).

When a vehicle replied with MtAB or MtAC to an RSE, it would send the vehicle's information including two types of information: VVD_ID and On-Duty flag. These two types of information would help to detect the emergency case existence. If a vehicle was an Ambulance, Fire, Police, or Governmental vehicle, then its VVD_ID would start with AVD, FVD, PVD, or GVD, respectively. So, if any of those vehicles were detected on a road, the RSE would detect the vehicle's On-Duty flag; whether it was 1 or 0. If the On-Duty flag was 0, then the vehicle was treated as a civilian vehicle. Whilst, if the On-Duty flag was 1, then the RSE would create and send Message type AG (MtAG).

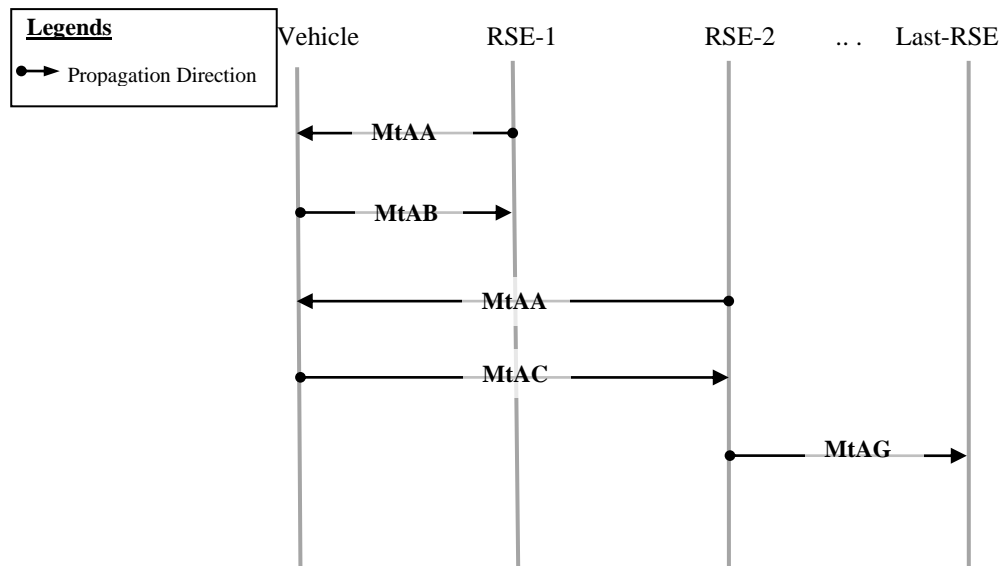


Figure 3.7: Sequence Diagramme for “The Relation between MtAA, MtAB, and MtAC”

Table 3.5: Emergency Level Calculation

VVD_ID	ON-Duty Flag	Emergency_Level
AVD	0	0
FVD	0	0
PVD	0	0
GVD	0	0
CVD	0	0
AVD	1	4
FVD	1	3
PVD	1	2
GVD	1	1
CVD	1	0

3.2.1.4 Integration with nearby roads

For an intersection to integrate with the surrounding roads, the general statuses of those roads were required for that intersection's TLC. That information was carried over two messages, MtAH and MtAI, which are explained in Table 3.6.

Table 3.6: The Messages Used to Deliver the Nearby Road's Status

Message Name	Description
MtAH	MtAH is a timed message being sent by the first RSE on the road to the Back intersection's TLC. MtAH carries the latest queue lengths (Total number of vehicles on each lane of the Next road or VTNN) and the capacity for a road to the back-side intersection. See Figure 3.8.
MtAI	MtAI is a timed message being sent from the last RSE on a road to the traffic light controller (TLC) on the next road. MtAI carries the latest queue lengths of a road to the next road's TLC (Total Number of vehicles Queuing on the lanes of the Next road or VNQB) and its capacity. See Figure 3.8.

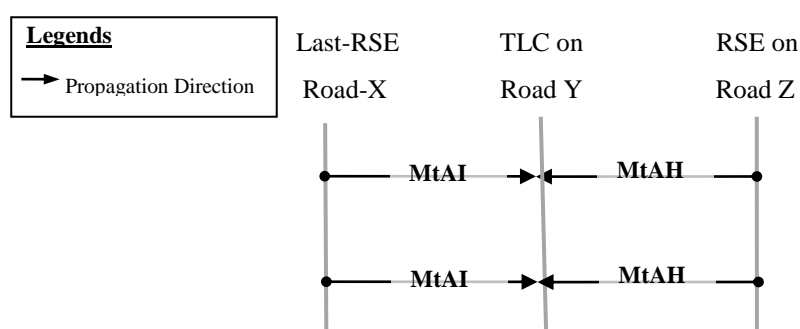


Figure 3.8: Sequence Diagram for Nearby Road Status Delivery

After collecting all of the data in the last RSE of each lane, they were sent to the Traffic Light Controller (TLC) to make a decision for the next phase plan. In the next section, the traffic light controller functionality (including the decision making algorithms) will be illustrated in details.

3.2.2 Traffic Light Phase Plan Decision Making

The Traffic Light Controller had eight sets of variables. Each set represented one flow direction. Each set contained eight values that represented the road's latest status. The 8 by 8 data matrix was used to make two decisions. The first decision was which pair of flow directions would be the next green phase, whilst, the second decision to be made by the Traffic Light Controller was the next phase green time. Figure 3.9 shows the block diagram of the modified Traffic Light Controller running the developed protocol.

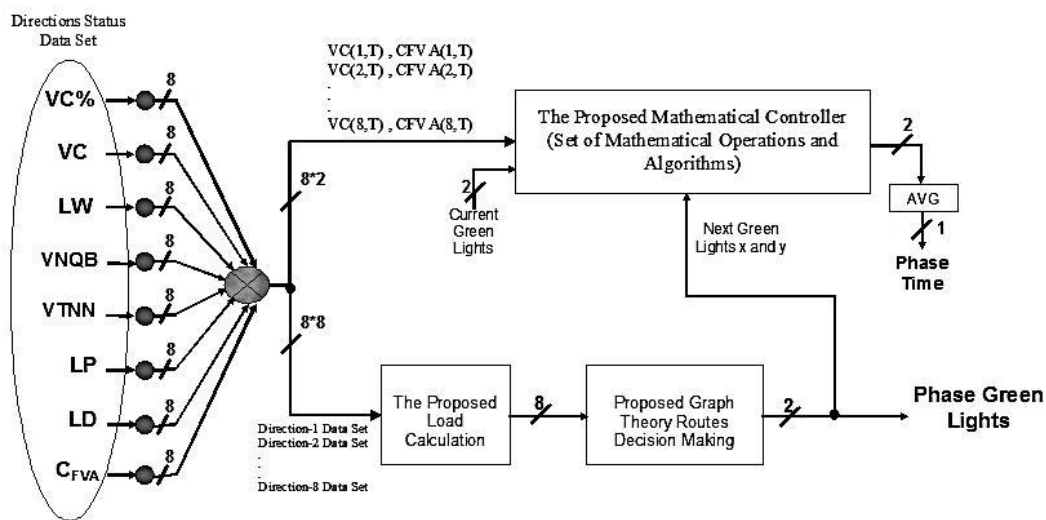


Figure 3.9: Traffic Light Controller with Developed Algorithm Core

3.2.2.1 Determination of the Next Phase Green Lights

The developed approach considers the priority of each lane and then the priority of each phase. After collecting the variables representing the road condition at a point of time T , the load applied on each direction (i , where i is the Direction index number $= \{0,1,2,3,4,5,6,7,8,9,10,11\}$ and $i \in I$) was calculated within the TLC using those variables in addition to some static values configured when installing the system over the road, for instance, the maximum number of vehicles to be occupied within a lane and the maximum number of vehicles to be occupied within one single detection area. See Figure 3.10.

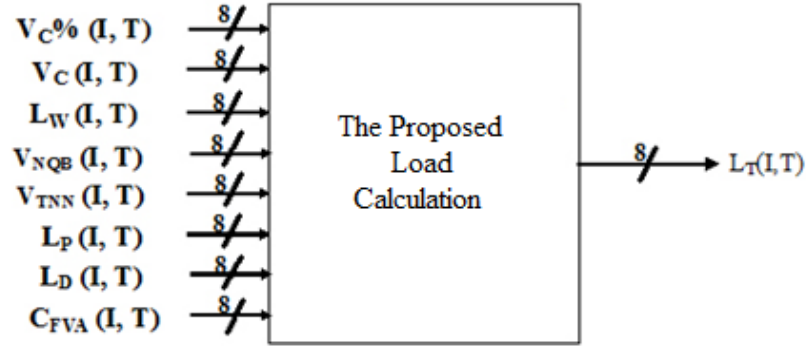


Figure 3.10: Developed Protocol Direction's Load Calculation Stage

The variables used to calculate each direction's load/weight have been defined briefly in Table 3.1 and will be defined in more detail in this section. Starting with $V_C(i,T)$, which can be defined as the Vehicles Count confirmed to be within the Queuing area of direction i at the point of time T , the first vehicle arrival confirmation Flag on direction i at the point of time T is abbreviated as $C_{FVA}(i,T)$. Whereas, $V_C\%(i,T)$ stands for what percentile of the vehicles first queuing area of the direction i 's road was occupied at the point of time T . When the first vehicle arrived to a red traffic light, a timing counter started counting the Waiting time ($L_W(i,T)$) for the first vehicle in the queue of direction i 's road at the point of time T . For detecting the emergency vehicle's existence, two variables were collected; these were the vehicle's priority $L_P(i,T)$ and the flag $L_D(i,T)$ for the special vehicle (driving on direction i at the point of time T) whether it was on Duty ($LD = 1$) or not ($LD = 0$). Two variables were collected for integration purposes; those were the $V_{NQB}(i,T)$ (Vehicle's Total Number Queuing on the Back-road traffic lights, those leading to the direction i 's road at the point of time T), and the $(100\% - V_{TNN}\%(i,T))$ (how much percentile of the next road (the road which receives vehicles coming from the direction i) is instantly occupied by vehicles at the point of time T).

Eventually, using the above collected variables, the load of each direction was calculated, as illustrated in Figure 3.11, then it was sent to the Graph Theory Routes Decision Making (Within the TLC) for the next phase's green light decision making, see Figure 3.12.

Directions Load Calculation Algorithm

```
1. GET the collected road status variable values.
2. FOR i = 1 to 11
    IF i = 3 or i = 6 or i = 9, THEN
        CONTINUE
    ELSE
        CALCULATE the Total direction i's Load at the point of time T:
        
$$L_T(i, T) \leftarrow V_C(i, T) * C_{FVA}(i, T) * V_C\%(i, T) * (L_W(i, T) + (L_P(i, T) * L_D(i, T)) +$$


$$V_{NQB}(i, T) * (100\% - V_{TNN}\%(i, T))$$

        END IF
    ENDFOR
```

Figure 3.11: Direction's Load Calculation Algorithm

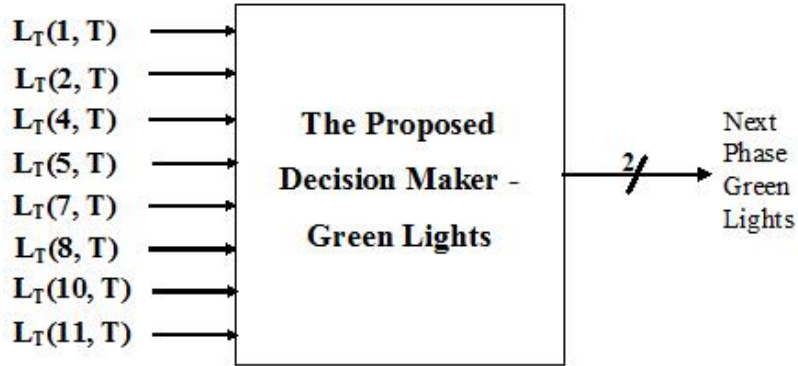


Figure 3.12: The Developed Protocol for the Decision Making of the Next Phase Green Lights

In [98], the total number of phase options was only four, which would reduce the chances to choose the most optimum phase sequence. The developed approach has solved this problem as it looks to each direction at the intersection as a node in a graph. See Figure 3.13. Each node had four relations with the four nodes intersecting with its direction; those were called adjacent nodes or neighbors.

When the time came to choose which two directions should become green next, the developed approach tried to find a solution using Dijkstra's Algorithm by looking to this problem as the shortest path problem [113]. Then it added some

additional pairing technique. Two initialization steps were implemented. The first step was to set the list of adjacent nodes for each node on the graph. See Figure 3.14. The second initialization step was assigning the values of the calculated loads of each direction to a node on the graph in Figure 3.13. The process of determining the next green lights started firstly, with a one-to-all pairing operation happening between the elements of the first current green's adjacent nodes and the elements of the second current green's adjacent nodes to get a list of 16 combinations. Secondly, an elimination operation happened to remove any paired-to-self combinations (e.g., Node H is paired to itself). Thirdly, another elimination operation was launched to remove the intersected paired nodes (e.g., if AC was a paired combination, and A intersected with C, then AC was removed from the list). This left the list of the available phase combinations.

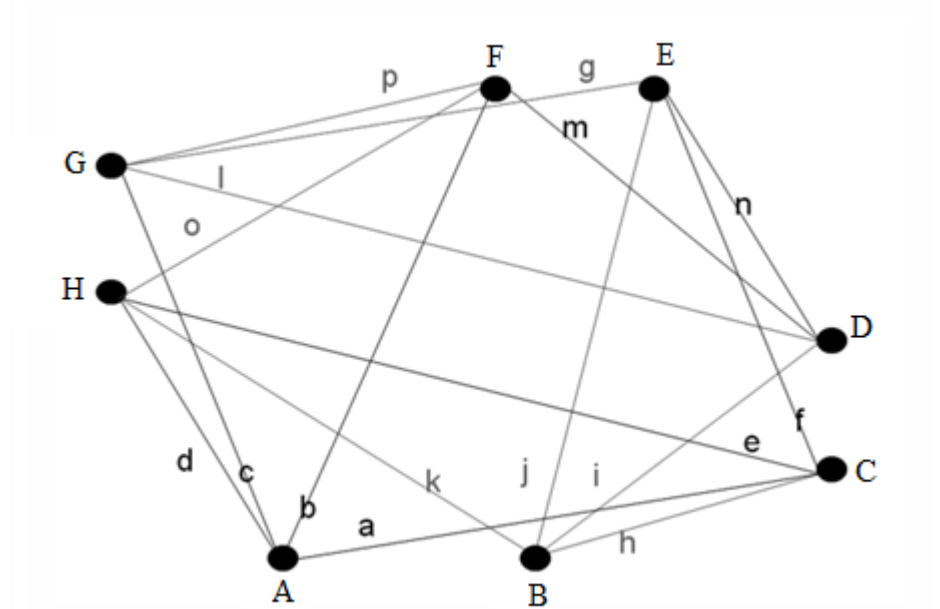


Figure 3.13: Phase Transition Map As Seen By the Developed Approach

Finally, the rest of the combinations were sorted according to their load summations in descending order. The first phase combination from the list was the next green phase as stated in Figure 3.14.

Next Phase Green Light Algorithm

1. **INIT** Adjacent Node (Neighbor) List of:
Node A \leftarrow C,F,G,H
Node B \leftarrow C,D,E,H
Node C \leftarrow A,B,E,H
Node D \leftarrow B,E,F,G
Node E \leftarrow B,C,D,G
Node F \leftarrow A,D,G,H
Node G \leftarrow A,D,E,F
Node H \leftarrow A,B,C,F
2. **SET** A \leftarrow L_T(1,T) , B \leftarrow L_T(2,T), C \leftarrow L_T(4,T) , D \leftarrow L_T(5,T) , E \leftarrow L_T(7,T), F \leftarrow L_T(8,T), G \leftarrow L_T(10,T), H \leftarrow L_T(11,T)
3. **DETERMINE** the available full-Mesh element-to-element pairs from the two currently green's adjacent node lists.
4. **ELIMINATE** the pair-to-itself combinations.
5. **ELIMINATE** the intersected (unavailable) pairs.
6. **ELIMINATE** any duplicated pairs.
7. **DECENDING SORT** the rest of the pairs in the list.
8. **SET** the first pair two elements on the list as the next phase green lights.

Figure 3.14: The Developed Next Phase Green Light Decision Making Algorithm

In Figure 3.15, an example of the developed next phase green light decision making algorithm was implemented. It started with the initial green phase as AB, going through all of the steps listed in Figure 3.14 and ending with the new phase green lights, G and C.

Current green phase is AB

First Step: Adjacent List of Node A: C,F,G,H

Adjacent List of Node B: C,D,E,H

Pair List:

CC,CD,CE,CH,FC,FD,FE,FH,GC,GD,GE,GH,HC,HD,HE,HH

Second Step: Paired-to-Itself elimination

Pair List:

CC,CD,CE,CH,FC,FD,FE,FH,GC,GD,GE,GH,HC,HD,HE,HH

New Pair List:

CD,CE,CH,FC,FD,FE,FH,GC,GD,GE,GH,HC,HD,HE

Third Step: Intersected pair elimination

Pair List:

CD,CE,CH,FC,FD,FE,FH,GC,GD,GE,GH,HC,HD,HE

New Pair List:

CD,FC,FE,GC,GH,HD,HE

Last Step: Sorting of the pair lists in descending order (Assuming that the load summation of Nodes C and G are the biggest) then:

Pair list after sorting: GC, CD,FE,FC,HD,GH ,HE

Then, the next green phase is GC.

Figure 3.15: Next Traffic Light Phase Light Decision Making Example

To change a phase, both of the two currently green nodes checked their Adjacency List to find the two maximum weighted nodes which were not adjacent to act as the next green phase.

3.2.2.2 Determination of the Next Phase Time

Choosing the next green light is one of the two phase decisions to be made. Now, the second decision to be determined is how long the green phase should be. The condition of each direction at the intersections was represented by 8 values. Only two values (V_C and C_{FVA}) among those 8 values were sent as inputs to the Phase time decision maker as shown in Figure 3.16. This was in addition to the indexes of the two next green lights and the two currently green lights, as can be seen in Figure 3.16.

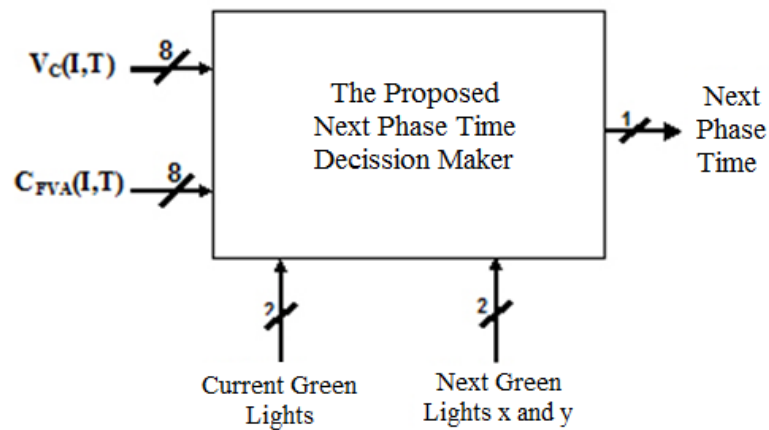


Figure 3.16: The Developed Approach's Next Phase Time Decision Maker

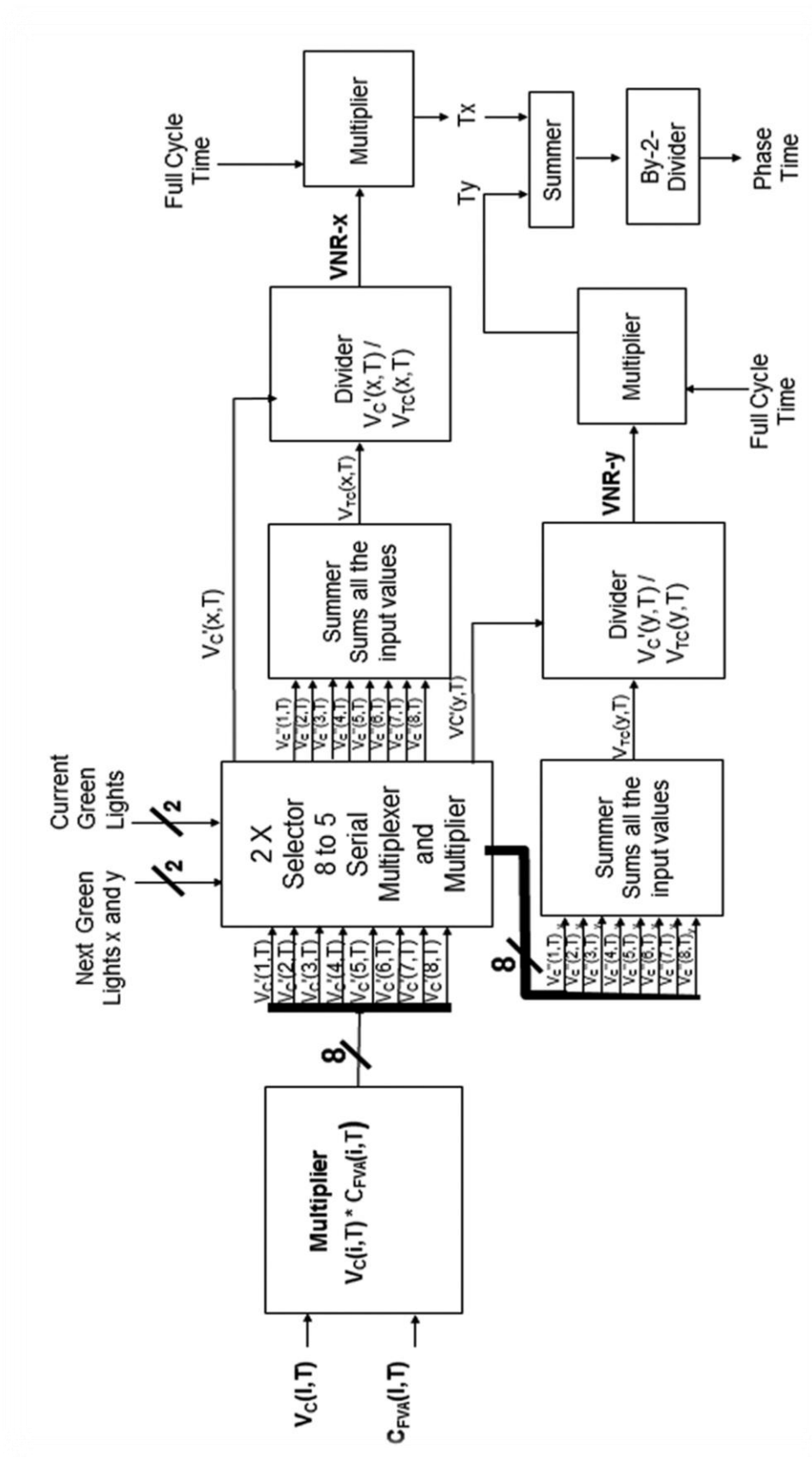


Figure 3.17: The Developed Algorithm Core Internal Architecture

Figure 3.17 represents the developed mathematical method for determining the next phase time as a block diagram. At the point of time T , twenty values would have arrived at the developed mathematical controller. Eight of them represented the number of vehicles ($V_C(I,T)$) that had been confirmed to have arrived at the queuing area of the lane indexed I ; where $I = \{0,1,2,3,4,5,6,7,8,9,10,11\}$. Whilst, the second eight inputs represented the first vehicle arrival confirmation flag ($C_{FVA}(i,T)$) for each direction i ; $i \in I$. The last four inputs to the Next Phase Time decision maker were divided into two sets; the first set carried the two index numbers of the current green directions, and the second set of values held the two index numbers of the next phase elected directions to be green.

The first step the controller performed was to confirm that there was at least one vehicle (at each of the 8 directions) queuing and waiting for the traffic light to become green. If no vehicle had already arrived at the traffic light of the direction i , then the $V_C(i,T)$ value would be neglected; otherwise, it would be passed to the next level as $V_C'(i,T)$. This operation was to be implemented inside a multiplier that received 2 sets of 8 values each, $V_C(I,T)$ and $C_{FVA}(I,T)$. Each element in the $V_C(I,T)$ set would be multiplied by its equivalent element in $C_{FVA}(I,T)$.

The second stage was to find out which two subsets of the inputs $V_C'(i,T)$ were adjacent to the two chosen green lights and if any of the subset members were currently green or not. The reason behind multiplying the $V_C'(i,T)$ by the Adjacency flag was to make sure that the correct mates were chosen for a more accurate time decision, as illustrated later in this section. Whilst, multiplying by the "Is i NOT currently green?" flag (G_{Cx}) was to make sure to not include the currently green lights into the consideration when deciding the next phase time as they were not among the next phase green lights.

The main steps that were to be performed by the mathematical algorithm are shown in Figure 3.18 and can be described as:

Step 1. Getting the queue length for each of the eight lanes ($V_C(1,T) \dots, V_C(8,T)$).

Step 2. Getting which two lanes will be green in the coming phase (G_{N1} and G_{N2}).

- Step 3. Summing the queue lengths of each of the two elected lanes with the queue lengths of its crossing lanes; those which are not flagged among the current green lights.
- Step 4. Getting the summation of the Current Green Light flags for each of the crossing lanes for the elected lanes including themselves.
- Step 5. Finding the ratio of each of the elected directions to become the next green phase to the total queue length with its crossings as seen in step 5 in Figure 3.18.
- Step 6. Multiplying the results gotten from the previous step by the summation of the current green flags' summation, and by the time for the single green phase (for example, the basic green time for a traffic light is 30 Seconds) by the number of legs of that intersection. See step 6 in Figure 3.18.
- Step 7. Finally, calculating the maximum, average, or minimum time among the two results. If the main purpose was to reduce the overall wasted time at the intersection, then choosing the Minimum Phase time would be the best choice. Whilst, if the main purpose was to reduce the queue lengths to the minimum, then choosing the maximum required phase time would be the best option. But, if the main aim was to maintain both the wasted time and the queue lengths, then it would be better to average both results to get the next phase green time as has been performed in step 7 in Figure 3.18.

Next Green Phase Time Algorithm

1. **GET** each direction's queue length (V_C), the first vehicle's arrival flag (C_{FVA}), the two Currently Green directions' IDs (G_{C1} , G_{C2}), the selected next phase two green lights' IDs (G_{N1} , G_{N2}), and the standard Full Cycle Time.
2. **DETERMINE** which direction's queues are confirmed to have arrived:
$$V_C'(i,T) \leftarrow V_C(i,T) * C_{FVA}(i,T)$$
3. **DETERMINE**, for each direction, which of its adjacent queues should be considered in calculating the next phase time whilst setting the rest (The Non-Adjacent or currently green) of them to zero.
$$V_{C(G_{N1})}''(i,T) \leftarrow V_C'(i,T) * (Is\ Node\ i\ adjacent\ to\ G_{N1}) * (i \neq G_{C1} \text{ and } i \neq G_{C2})$$
$$V_{C(G_{N2})}''(i,T) \leftarrow V_C'(i,T) * (Is\ Node\ i\ adjacent\ to\ G_{N2}) * (i \neq G_{C1} \text{ and } i \neq G_{C2})$$
4. **CALCULATE**, for each of the two directions, the summation of the results in step 3.
$$V_{CT}(G_{N1},T) \leftarrow Sum(V_{C(G_{N1})}''(i,T)) \text{ for } i = 1 \text{ to } 11$$
$$V_{CT}(G_{N2},T) \leftarrow Sum(V_{C(G_{N2})}''(i,T)) \text{ for } i = 1 \text{ to } 11$$
5. **CALCULATE**, for each of the two directions, the division of the chosen direction's queue length over the total summation found for that direction in step 4.
$$VNR_{G_{N1}} \leftarrow V_C(G_{N1},T) / V_{CT}(G_{N1},T)$$
$$VNR_{G_{N2}} \leftarrow V_C(G_{N2},T) / V_{CT}(G_{N2},T)$$
6. **CALCULATE**, for each of the two directions, what percent of the full cycle time must be given to that direction.
$$G_{TN1} \leftarrow VNR_{G_{N1}} * Full_Cycle_Time\ (120\ Seconds)$$
$$G_{TN2} \leftarrow VNR_{G_{N2}} * Full_Cycle_Time\ (120\ Seconds)$$
7. **DETERMINE** the next phase time.
$$Next_Phase_Time \leftarrow Average(G_{TN1}, G_{TN2})$$

Figure 3.18: The Developed Next Phase Time Decision Making Algorithm

A verification example for the above algorithm is shown in Table 3.7 where the current green phase is (1-2) and it is the time to make decision about the phase time when it was determined to be (4-5), (7-8), or (10-11). The detailed calculations are shown in Appendix E.

Table 3.7: Next Phase Green Time Determination Examples

Phase Transit from the Southern Road to the Eastern Road								
Parameters	Southern Road		Eastern Road		Northern Road		Western Road	
	Lane 1	Lane 2	Lane 4	Lane 5	Lane 7	Lane 8	Lane 10	Lane 11
Initial Queue	0	0	100	100	75	75	50	50
Currently Green (G _C)	1	1	-	-	-	-	-	-
Next Green Light (G _N)	-	-	1	1	-	-	-	-
V _C '	0	0	100	100	75	75	50	50
V _{C(4)} ''	-	-	100	-	75	-	-	50
V _{C(5)} ''	-	-	-	100	75	75	50	-
VCT	-	-	225	300	-	-	-	-
VNR	-	-	0.445	0.334	-	-	-	-
GTN	-	-	53.4	40.08	-	-	-	-
Next Phase Time	-	-	47 (Rounded)		-	-	-	-

Phase Transit from the Southern Road to the Northern Road								
Parameters	Southern Road		Eastern Road		Northern Road		Western Road	
	Lane 1	Lane 2	Lane 4	Lane 5	Lane 7	Lane 8	Lane 10	Lane 11
Initial Queue	0	0	100	100	75	75	50	50
Currently Green (G _C)	1	1	-	-	-	-	-	-
Next Green Light (G _N)	-	-	-	-	1	1	-	-
V _C '	0	0	100	100	75	75	50	50
V _{C(7)} ''	-	-	100	100	75	-	-	50
V _{C(8)} ''	-	-	-	100	-	75	50	50
VCT	-	-	-	-	325	275	-	-
VNR	-	-	-	-	0.23	0.273	-	-
GTN	-	-	-	-	27.6	32.76	-	-
Next Phase Time	-	-	-		30 (Rounded)		-	-

Phase Transit from the Southern Road to the Western Road								
Parameters	Southern Road		Eastern Road		Northern Road		Western Road	
	Lane 1	Lane 2	Lane 4	Lane 5	Lane 7	Lane 8	Lane 10	Lane 11
Initial Queue	0	0	100	100	75	75	50	50
Currently Green (G _C)	1	1	-	-	-	-	-	-
Next Green Light (G _N)	-	-	-	-	-	-	1	1
V _C '	0	0	100	100	75	75	50	50
V _{C(10)} ''	-	-	-	100	75	75	-	50
V _{C(11)} ''	-	-	100	-	-	75	-	50
VCT	-	-	-	-	-	-	300	225
VNR	-	-	-	-	-	-	0.167	0.223
GTN	-	-	-	-	-	-	20.04	26.76
Next Phase Time (Sec)	-	-	-		-	-	23 (Rounded)	

After making the second decision, the next phase plan was ready to be applied on the intersection; then, it was expected to see a smoother traffic flow, less congested roads and a better traffic light performance as listed in chapter one. The developed approach's duty was performed at the point when both of the traffic light phase decisions were made. So, from the features of the protocol functionality, it was suggested to name it the "Dynamic Traffic-light Phase Plan Protocol" and shortened to DT3 P.

3.3 Simulation / Evaluation Procedures

Till this moment, the developed approach is just a hypothesis. This section will describe the simulation tool used to examine how effective the developed protocol is to manage the situation at a signalized intersection at different levels of congestion.

3.3.1 Customized Traffic Light Simulation Tool Development

The reason behind developing the customized simulation tool, even though other traffic light simulators exist, was because they did not offer the researchers the ability to write down and modify the controller algorithm. The customized simulation tool in this work has followed the design of the Sidra Intersection simulator with the addition of the ability to choose which algorithm is to be performed by the Traffic Light Controller. For other researchers to use this customized simulation tool, all they need to do is override the controller functions with their proposed ones.

The customized simulation tool was developed using the programming language provided by MATLAB platform to write a series of functions. The customized simulation tool's functions were written inside m-Files which can be executed to take inputs and produce outputs. Figure 3.19 shows the simulation tool functions map.

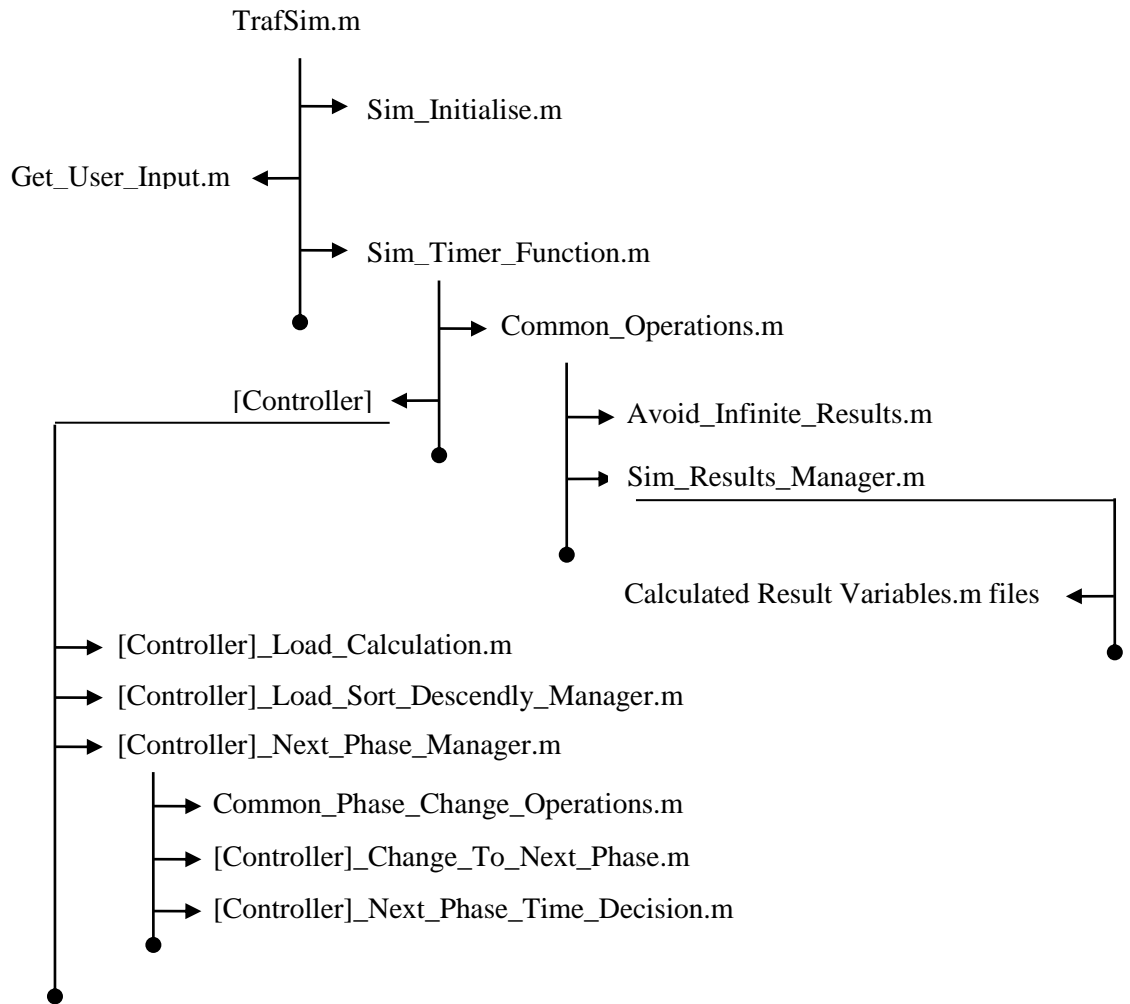


Figure 3.19: The Customized Simulation Tool Function (.m Files) Map

The customized simulation tool provides the ability for the user to decide how many times a case study scenario is to be repeated to achieve a specific level of confidence. In addition, it prompts the user to enter the total time for one scenario simulation (Sim_Time). Another facility for the user to use is the vehicle arrival pattern as he can use any one from among three (Deterministic Arrival, Uniform-Random, Poisson-Random).

The results of the customized simulation tool will be given as MATLAB tables. For the users to analyze the data, they need to copy those results from the MATLAB tables and paste them into any statistics software tables so that they are presented by graphs or simplified tables.

3.3.2 Traffic Light Simulation Tool Architecture

In order to examine its framework functionality and compare its performance with other approaches, this simulation tool has been created using MATLAB Programming Language. This customized simulation tool acts as a four leg intersection and performs three main models as shown in Figure 3.20. The simulation tool was created following the models concepts used in the Sidra Intersection simulator [24].

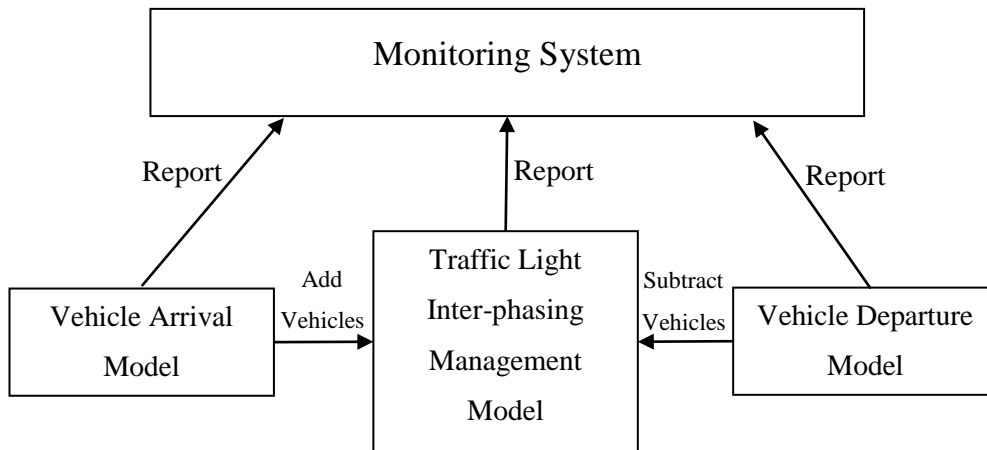


Figure 3.20: Simulation Tool's Traffic Light Block Diagram Model

The simulation tool has a monitoring system which has the collective values for each variable. In this section, the traffic light model used by the simulation tool will be illustrated then the monitored variables will be briefly illustrated along with how they are updated by the simulation tool's monitoring system.

3.3.2.1 Vehicle Arrival Model (VAM)

Vehicles arrive to queues in a Poisson distribution manner. Poisson distribution is a discrete random variable distribution that shows the probabilities regarding the number of an event's occurrence at a point of time. The customized simulation tool uses Poisson distribution to randomize the vehicles' arrivals to each direction. As mentioned before, the customized simulation tool was created by MATLAB, so the Poisson distribution function used was `poissrnd (Lambda)`; where, Λ was the average rate of the value.

3.3.2.2 Traffic Light Inter-Phasing Management Model

The duty of this model is to do the switching between the traffic light phases. As when a direction becomes green, the Vehicles_Departure_Model (VDM) would be instantiated to start running; whilst, for those with red lights, the Vehicles_Departure_Model would be stopped.

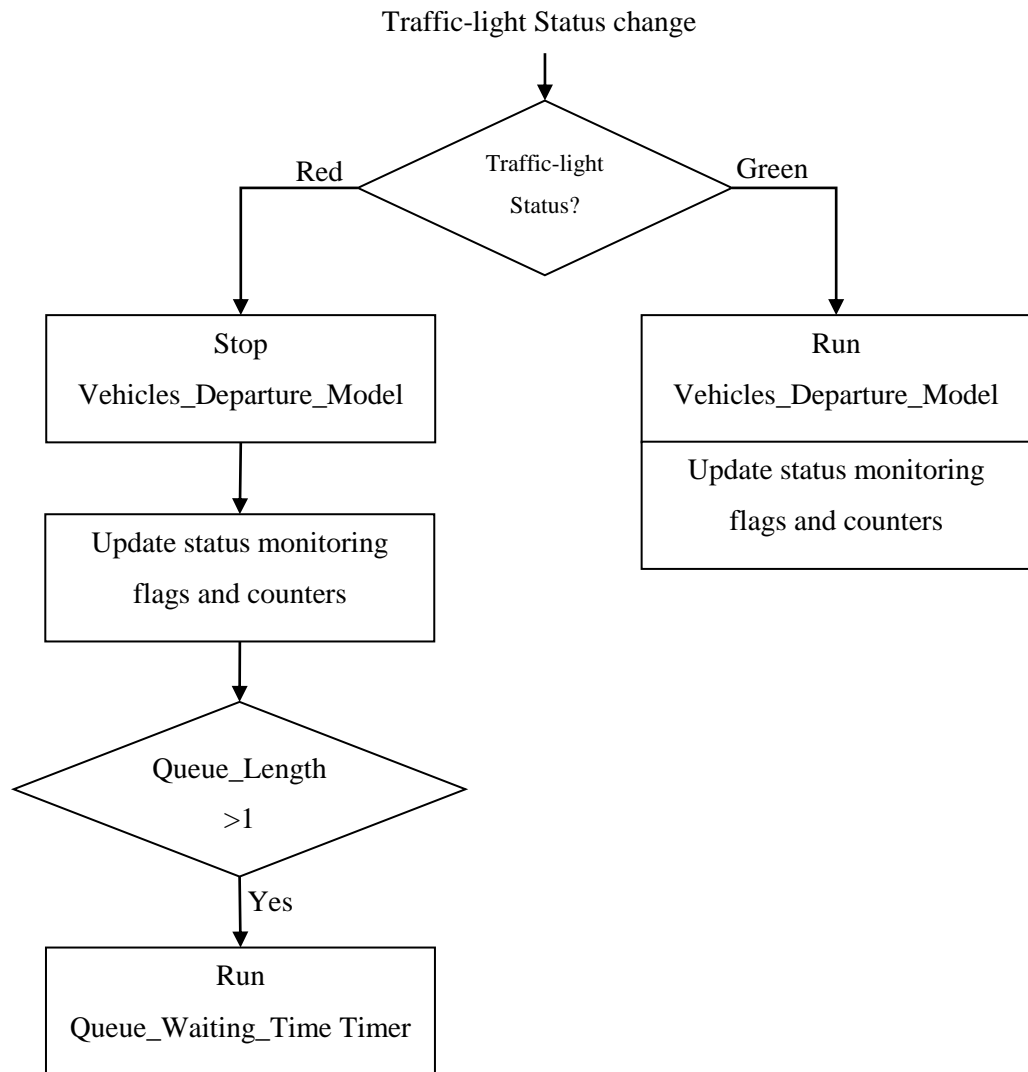


Figure 3.21: Traffic Light Phase Management Model

3.3.2.3 Vehicle Departure Model (VDM)

In this section, the distribution of the count of the vehicles leaving a queue when the traffic light is green will be determined as illustrated in Figure 3.22. When a traffic light phase changes from red to be green, the first vehicle needs a time delay of 2 seconds to 4 seconds (averaged as 3 seconds) to respond to the traffic light phase change (including the time for the drivers physical response and engaging the vehicles gears). In addition to that 3 seconds, one more second is needed to pass the traffic light stopping line which makes the first vehicle out of the queue. So the first 4 seconds named as the time delay for the first vehicle in the queue (TD_{FV}) and it is a must to waste time. Nevertheless, TD_{FV} is not counted as part of the given green time.

According to the Highway Capacity Manual [107] which used the approximate approach described in [114] model, the first vehicle on the queue would take an average delay (TD_{FV}) of 4 seconds to leave the queue, and each vehicle behind would leave the queue after an average time (TD_{1V}) of one second of the vehicle in front. The time delay for each vehicle to leave the queue would be calculated using the equation (3.1).

$$T_{VL} = TD_{FV} + O_{VQL} * TD_{1V} \quad (3.1)$$

Where the T_{VL} is the vehicle's leaving time, TD_{FV} is the average time delay for the first vehicle, O_{VQL} is the vehicle's order within the queue, and the TD_{1V} is the average delay for each vehicle to leave after the first vehicle, which is averaged to be one second.

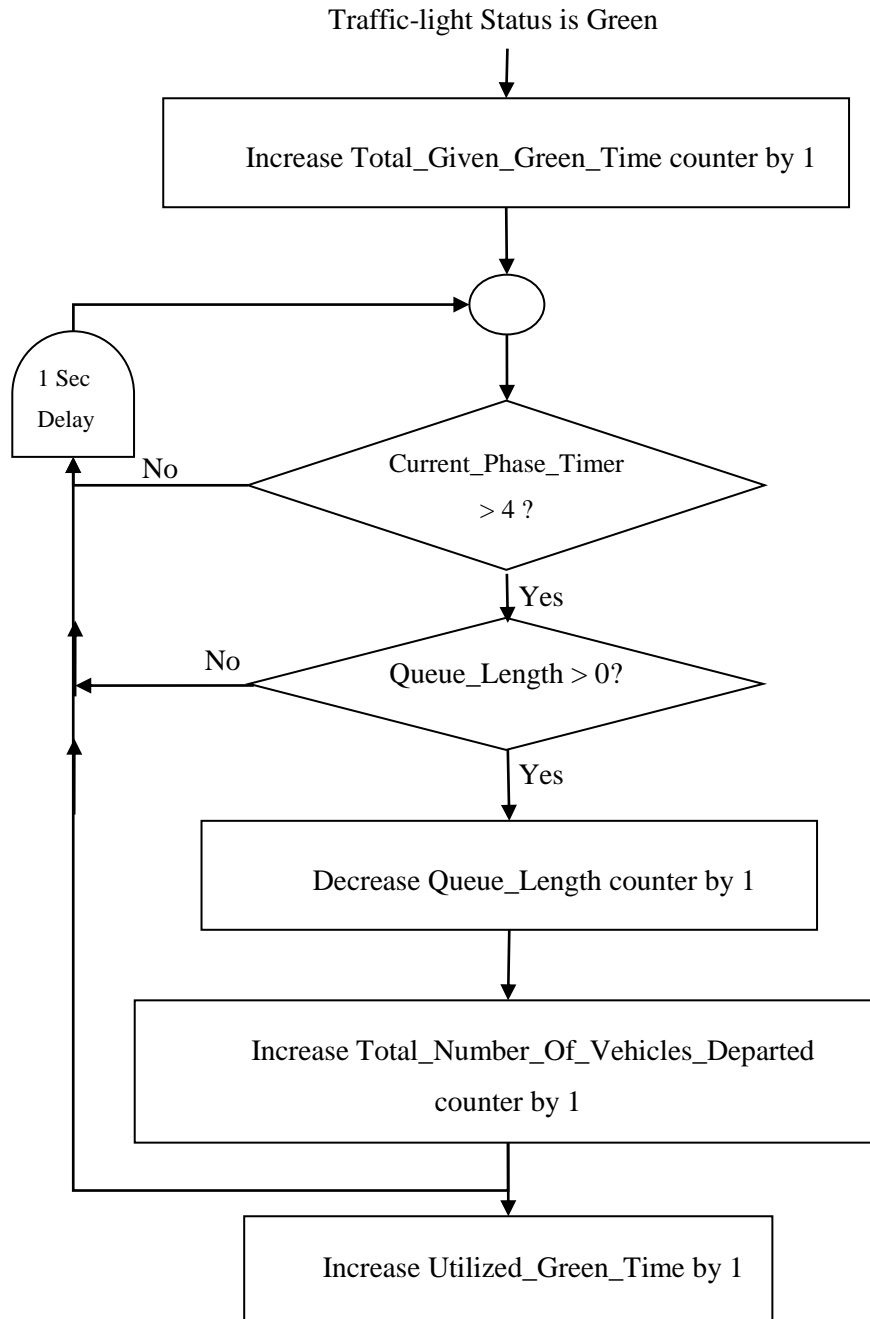


Figure 3.22: Vehicle Departure Model

In case all the intersection legs are free of vehicles, DT3P was programmed to revert to work as a fixed time controller giving 9 seconds for each of the four standard phases as a minimum green time till a vehicle appears on any of the lanes. Unless those 9 seconds are being used by vehicles to pass the intersection, they will be considered as wasted time by the simulation tool.

3.3.2.4 Simulation Tool's Monitoring System

The customized simulation tool has a monitoring entity which will keep the log of all the variables, those categorized into three categories and use them to produce the results report. In this section, the usage of the logged variables and the production of the final results report will be illustrated.

Mainly, the tool has three categories of variables: road's status variables, collected result variables, and calculated result variables. Table 3.8 shows the road's status variables that would change during the simulation run-time according to the simulation scenario and the decisions made during the simulation time.

The second type of variables is shown in Table 3.9 which is the simulation tool's collected result variables. The list is updated based on event occurrence during the simulation run-time to participate later at the end of the simulation time in calculating the final results report variables shown in Table 3.10. The final results report consists of seven variables those are:

1. Departed/arrived vehicles ratio, which will be calculated based on two of the collected variables; $Total_Number_of_Vehicles_Passed$ and $Total_Number_of_Vehicles_Arrived$ as shown in equation (3.2).

Departed_to_Arrived_Vehicles_Ratio

$$= \frac{Total_Number_of_Vehicles_Passed}{Total_Number_of_Vehicles_Arrived} \quad (3.2)$$

2. Average_Waiting_Time it would be determined at the end of the simulation from the division of the $Total_Waiting_Time$ (recorded during the one hour of simulation) by the number of the $Total_Number_of_Phase_Changes$ during the simulation time, as shown in equation (3.3).

$$Average_Waiting_Time = \frac{Total_Waiting_Time}{Total_Number_of_Phase_changes} \quad (3.3)$$

3. Maximum_Waiting_Time would be determined and logged during the simulation run-time.

4. *Average_Queue_Length* would be calculated from the division of the *Total_Queue_Length* (recorded during the one hour of simulation) by the *Total_Number_of_Phase_Changes*, as can be seen in the equation (3.4).

$$\text{Average_Queue_Length} = \frac{\text{Total_Queue_Length}}{\text{Total_Number_of_Phase_changes}} \quad (3.4)$$

5. *Maximum_Queue_Length*, this variable will be determined and recorded during the simulation run-time.
6. *Utilized_Green_Time_Ratio*, which will be the result of dividing the *utilized_Green_Time* to the *Total_Given_Green_Time*, as shown in equation (3.5).

$$\text{Utilized_Green_Time_Ratio} = \frac{\text{Utilized_green_time}}{\text{Total_Given_Green_Time}} \quad (3.5)$$

7. *Output-to-Input Response ratios*, this will give a general representation on how accurate the controlling system is in giving green time when responding to the levels of demand (input). It can be calculated by the equations (3.6-3.11).

$$\begin{aligned} &\text{Lane_Given_Green_Time_Ratio} \\ &= \frac{\text{Lane_Given_Green_Time}}{\text{Intersection_Total_Given_Green_Time}} \end{aligned} \quad (3.6)$$

$$\begin{aligned} &\text{Lane_Vehicles_Arrival_Ratio} \\ &= \frac{\text{Lane_No._of_Arrivals}}{\text{Intersection_Total_Arrivals}} \end{aligned} \quad (3.7)$$

$$\begin{aligned} &\text{Lane_Output_to_Input_Ratio} \\ &= \frac{\text{Lane_Given_Green_Time_Ratio}}{\text{Lane_Vehicles_Arrival_Ratio}} \end{aligned} \quad (3.8)$$

Intersection_Given_Green_Time_Ratio

$$= \sum_{index=1}^{11} \text{Lane_Given_Green_Time_Ratio (index)} \quad (3.9)$$

Intersection_Vehicles_Arrival_Ratio

$$= \sum_{index=1}^{11} \text{Lane_Vehicles_Arrival_Ratio (index)} \quad (3.10)$$

Intersection_Output_to_Input_Ratio

$$= \frac{\text{Intersection_Given_Green_Time_Total_Ratio}}{\text{Lane_Vehicles_Arrival_Total_Ratio}} \quad (3.11)$$

At the end of each scenario simulation, the whole results report's variables will be logged and kept in a matrix, reset the simulation tool's variables, then run the same scenario simulation again. The simulation tool will keep repeating the scenario according to the number of replications as will be illustrated in the results level of confidence section within this chapter. At the end, an overall results report would be created and shown to the user of the simulation tool.

Table 3.8: Road Status Variable List

Variable's Name	Variable's Description
CFVA	It refers to the first vehicle arrival flag (C_{FVA}). It can take two values (0: Empty queue, 1: at least one vehicle is queuing)
Green_Phase_ Time	This variable refers to the second Traffic Light Controller decision, the next phase time.
LD	It refers to the On-Duty Flag value (L_D). Its value was set to zero during the simulation scenarios in this thesis.
LN	It refers to the (V_c) Total Vehicle Count of a queue. The LN increases upon the arrival of a vehicle to the queue whilst it decreases when a vehicle leaves the queue.
LP	It refers to the Maximum priority (L_P) detected on the road. Its value was set to 1 during the simulation scenarios in this thesis as it was assumed that all of the vehicles were Civilians (No Emergency.)
LT	It refers to the over-all lane load which would be calculated within the Traffic Light Controller. Its value gives the priority for a direction to become green.
LW	It refers to the waiting time of the first vehicle in a queue. Its timer starts counting at the arrival of a vehicle (being the first in the queue) to a red traffic light. And, it would be reset to 0 when the traffic light for that specific direction becomes green.
Traffic_Light_ Status	This variable can take three values, each refers to a traffic light status color (0: RED, 1: AMBER, 2: GREEN). It is decided by the Traffic Light Controller.
VNQB	It refers to the V_{NQB} . Its value was set to zero (No affect) when performing the experiments in this thesis as its affect was needed for integration only.
VTNN	This variable refers to the V_{TNN} . Its value was set to zero (No affect) when performing the experiments in this thesis as its affect was needed for

	integration only.
--	-------------------

Table 3.9: Simulation Tool's Collected Result Variables

Variable's Name	Variable's Description
Total_Given_Green_Time	It is an incremental variable. It counts the total time for a traffic light when it is green.
Total_Number_of_Phase_Changes	It is an incremental variable which keeps the count of how many times a direction's traffic light changes its status from green to red.
Total_Number_of_Vehicles_Arrived	It is an incremental variable which keeps the count of the vehicles that had arrived to a direction's queue during the simulation time.
Total_Number_of_Vehicles_Passed	It is an incremental variable which keeps the count of vehicles that left a direction's queue during the simulation time.
Total_Queue_Length	This is an incremental variable which adds on the summation of a direction's Queue_Length (LN) each time the direction's traffic_light_status becomes green.
Total_Waiting_Time	This is an incremental variable which adds on the summation of queue's waiting time (LW) each time the direction's traffic light becomes green.
Utilized_Green_Time	Is an event-based variable which will be updated whenever a lane has a queue being processed while green phase.

Table 3.10: Simulation Tool's Calculated Result Variables List

Variable	Description
Average_Queue_Length	This variable was calculated at the end of the simulation. And it was determined from the division of the Total_Queue_Length by the Total_Number_of_Phase_Changes.
Average_Waiting_Time	This variable is calculated at the end of the simulation. And it is being determined from the division of the Total_Waiting_Time by the Total_Number_of_Phase_Changes.
Departed-to-Arrived Vehicles Ratio	This variable would represent the ratio of the Total_Number_of_Vehicles_Passed to Total_Number_of_Vehicles_Arrived
Max_Queue_Length	Is the maximum queue length value recorded during the simulation time. It would be updated every second and the final value would be determined at the end of the simulation.
Max_Waiting_Time	Is the maximum queue's waiting time recorded during the simulation time. It would be updated every second and the final value would be determined at the end of the simulation.
Utilized_Green_Time	This variable is calculated after the end of the simulation time. It was determined by dividing the Total_Used_Green_Time by the Total_Given_Green_Time. The nearer the value of this variable to 1, the better the performance.

3.3.3 Validation of the Traffic Light Simulation Tool

After creating the simulation tool by following the Sidra Intersection simulator vehicle flow concepts, it was validated using the same traffic simulator. The validation process performed a comparison between the Sidra Intersection simulator [24] and the customized simulation tool that was created using MATLAB. The traffic light system mainly had two types of activities: Traffic light standard activities and the Traffic Light Controller activities. The second part might have taken input from the first part to make decisions. If the Traffic Light Controller changed, the standard operations were not affected. The aim of the comparison was to validate the behavior similarity of the standard activities when running over the two simulators, using the Fixed-Mode as the Traffic Light Controller (Also known as pre-timed traffic controller).

The Sidra Intersection simulator is a traffic engineering simulator used mainly by civil engineers to evaluate road/intersection/roundabout designs. The Sidra Intersection simulator uses a set of data related to the geometrical design of the roads that were not necessary for the customized simulation tool validation process. But, there were two common variables depended on for the validation purpose: Cycle time and Maximum queue length.

Five case studies were used for the validation process. 5 different levels of vehicle arrival flow rates or what is called as level of demand were applied. And, the cycle time and the maximum queue length were checked. The validation hypothesis is “If the customized simulation tool gives exactly the same Cycle Time as the Sidra Intersection simulator whilst achieving almost the same Maximum Queue Length as the Sidra Intersection simulator, then the customized simulation tool’s standard activities are valid.”

3.3.4 Results' Level of Confidence

Level of confidence is mainly represented by a percentage which tells how likely it is to get similar results when repeating the same experiment [115]. The experiments in this thesis were performed with a level of confidence of 95%.

Level of confidence can be calculated using equation (3.12).

$$n = \frac{n_0 N}{n_0 + (N-1)} \quad (3.12)$$

Where n represents the number of replications for a finite population, N is the total population, and n_0 is the number of replications for unknown population which can be determined from equation (3.13).

$$n_0 = \left(\frac{Z \sigma}{e} \right)^2 \quad (3.13)$$

Where the Z is the confidence level ($Z=1.96$ for the 95%), σ is the standard deviation which take the default value of 0.5, and e represents the margin of error or confidence interval which is $\pm 5\%$ for the 95% level of confidence ($1 - 0.95 = 0.05$).

According to the calculations done using the equations (3.12) and (3.13), the number of replications of 152, 190, 254, 287, and 297 times were needed to achieve the 95% of confidence for the very-small, small, medium, large, and very large levels of demand, respectively. In other words, each experiment had to be repeated a number of times, equal to the calculated number of replications for each scenario, to be 95% confident of the results that were obtained from the valid simulation tool.

3.3.5 Traffic Light Performance Evaluation Procedures

As illustrated in Figure 3.23, a set of steps were taken to evaluate the efficiency, stability, and the accuracy of this work's methodology. In this section, a list and illustration of those steps are presented.

For the purpose of testing this methodology, the vehicles arrival flow rate on each of the intersection's directions were manipulated according to two patterns and then the results were checked. The first pattern gave a different arrival flow rate to each approach through one stage whilst the second Pattern gave all of the directions for the same arrival flow rates through five stages.

In the first pattern, four different levels of demand will be applied on the four legs of the intersection as a single scenario. The purpose of the first pattern is to show the high accuracy of the developed approach's response to the arrival rates and the flexibility in following the desired flow rate. The first pattern results will be shown as a bar chart representing the relationship between the arrival rates and the given green time rate for each of the intersection's lanes.

In the second pattern, the vehicle arrival flow rate was increased over five stages: very-small, small, medium, large, and very-large. Then, according to [24], [116], and [117], the arrival flow rate that was more than 1300 vehicles per hour per lane was considered at saturation level. Whilst, less than 250 vehicles per hour per lane was considered a very small arrival rate. So, the in between interval was divided into the five stages: very-small, small, medium, large, and very-large. They received around 250, 375, 750, 1125, and 1300 vehicles per hour per lane, respectively.

After setting up the simulation case studies, the experiments will run for one hour each, and then the following questions should be answered:

- a. **Green Time – Input Volume relationship:** Can DT3P optimize the Green Time given to each direction, according to the input (Vehicle Arrival Flow Rate), compared to the other controlling methods?
- b. **Intersection Capacity:** Does DT3P have the ability to pass more vehicles in one hour than the other controlling methods?
- c. **Average Queue Length:** Does DT3P reduce the Average Queue Length compared to the other controlling methods?
- d. **Average Vehicle Waiting Time:** Does DT3P reduce the Average Vehicle Waiting Time compared to the other controlling methods?

- e. **Maximum Waiting Time:** Does DT3P affect the Maximum Waiting Time positively or negatively as compared to the other control methods? If negatively, is it acceptable?
- f. **Maximum Queue Length:** Can DT3P reduce the Maximum Queue Length compared to the other controlling methods?
- g. **Given Green Time utilization:** How efficiently would the given green time be utilized by all of the controlling methods?

Each experiment was repeated as many times as the number of replications which was calculated to achieve a 95% level of confidence. At the end of the procedure, the collected results (from the second pattern experiments) were analyzed with the help of a simple ranking system (order-based ranking system). The ranking system was implemented to find out how much each method would score in terms of each of the above factors. The purpose of having this ranking system was to get a clearer picture on how efficient each method was compared to the other methods.

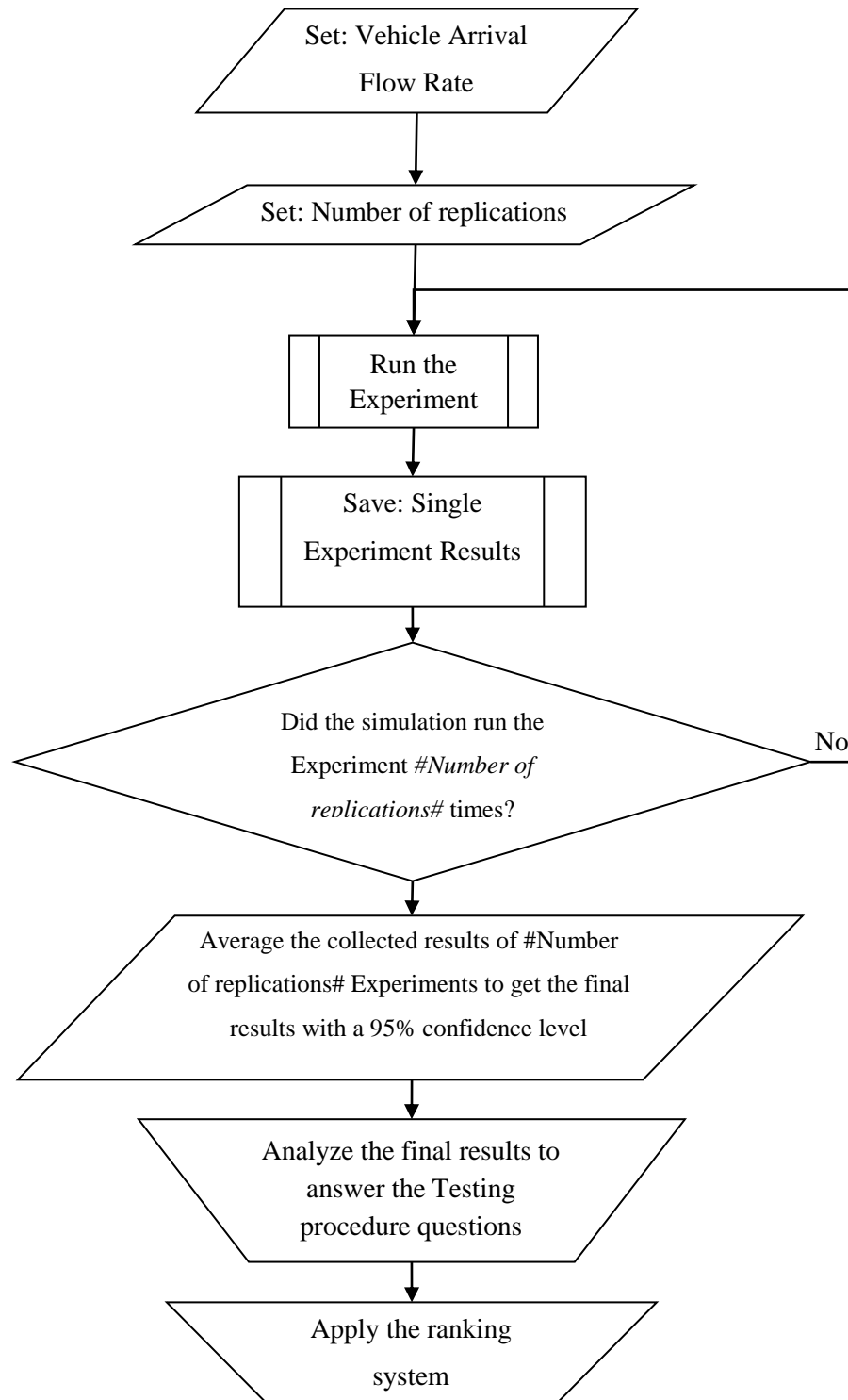


Figure 3.23: Testing Procedure Flow Chart

3.3.6 Traffic Light Performance Ranking System

Usually, a ranking system is used to indicate a winner member among a set of members. In this thesis, it was used as well to clarify the collected results from the testing procedures. The ranking values were represented by the efficiency decremented order. Basically, it is a point-based ranking system where the methods' performances were evaluated by giving each method an amount of points that is equal to the method's performance order value. The following example explains how the order-based ranking system works.

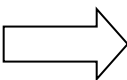
Methods	F.1	F.2	F.3	Mapped into 	Methods	F.1	F.2	F.3	Total Score
M.1	20	52	15		M.1	3	2	1	6
M.2	15	21	32		M.2	1	3	3	7
M.3	18	60	23		M.3	2	1	2	5

Figure 3.24: Ranking System Operational Example

An example on how the ranking system works, assume the example shown in Figure 3.24. As can be seen in the left-side table, there are three methods (M.1, M.2, and M.3) have been evaluated through three factors (F.1, F.2, and F.3) respectively. The evaluation values on the left-side table would be mapped into scores (the right-side table) according to the factors rules; for F.1, the more the better, for F.2, the less the better, and for F.3, the more the better. This is why, for F.1, M.1 have scored the highest and M.2 have scored the least. So after mapping all the performance values into scores, each method's total score would be calculated according to an equation designed based upon the usage purpose. In this example, the summation of all the three factors scores for each method will be calculated. The higher full-score a method performs, the better overall performance have achieved. That is, M.2 ranked as the best performer, followed by M.1, and the last is M.3.

3.4 Summary

In this chapter, the proposed methodology to achieve the thesis research objectives and the used methods were defined in detail including the traffic light standard operations/values and inter-phase decision making (Next phase green lights and next phase time). In addition, the customized simulation tool which was used to evaluate the developed approach (DT3P) performance was clarified with the evaluation procedures and the used ranking system.

In the next chapter, the results of the DT3P performance evaluation will be shown and discussed. At the end of the chapter a comprehensive analysis for the results will take place to check whether the desired thesis goals were reached or not.