Adaptive Fuzzy Control to Design and Implementation of Traffic Simulation System

Laheeb Mohammed Ibrahim
Software Engineering
Mosul University, Collage of Computer Sc. & Math.
Mosul , Iraq

Mohammed A. Aldabbagh
Software Engineering
Mosul University, Collage of Computer Sc. & Math.
Mosul , Iraq

***Abstract*—**
In this paper a Fuzzy Adaptive Traffic Signal System (FATSS) was designed and implemented to improve optimization and compare fix time traffic light controller. FATSS allows the user to select input parameters and tune rule base to improve optimization and compare fix time traffic light controller.

FATSS reducing the average waiting time for vehicles between 2% to 20%, and that indicate the adaptive traffic light controller based on fuzzy logic outperform is better when is compare with other fixed controller

FATSS was built using C# language in Microsoft Visual studio 2010 development environment. The simulation is implemented by Simulation for Urban Mobility(SUMO).

Keywords-component; formatting; style; styling; insert (key words)

I. INTRODUCTION

Transportation research has the goal to optimize transportation flow of people and goods. As the number of road users constantly increases, and resources provided by current infrastructures are limited, intelligent control of traffic will become a very important issue in the future. However, some limitations to the usage of intelligent traffic control exist. Avoiding traffic jams for example is thought to be beneficial to both environment and economy, but improved traffic-flow may also lead to an increase in demand [12].

Traffic light optimization is a complex problem. Even for a single intersection, there might be no obvious optimal solution. With multiple junctions, the problem becomes even more complex, as the state of one light influences the flow of traffic towards many other lights. Another complication is the fact that flow of traffic constantly changes, depending on the time of day, the day of the week, and the time of year. Roadwork and accidents further influence complexity and performance [38]. In fact, most traffic lights are controlled by fixed-time controllers. A cycle of configurations is defined in which all traffic gets a green light at a fixed point.

The split time determines how long the lights must keep on in each state. Heavy traffic roads can get preference by adjusting the split time. The cycle time is the duration of a complete cycle. In heavy traffic, longer cycles guide to good scenario.

Adaptive control strategies rely mainly on the prediction of the arriving flow [9]. There are two types of prediction. The first is based on the real-time data measured in the field to estimate the movement of vehicles detected. The other is based upon historical data to predict the future arriving flow. For convenience of referencing, we call the former *estimation* and the latter *prediction*. While long-term optimization is ideal for reaching the global optimums, the real-time data based control relies on a number of short-term optimizations to reduce uncertainty in traffic demand and improve accuracy in computation. The short-term optimization, usually in the order of 30 to 60 seconds, makes it possible for all the optimizing processes to be based on estimation rather than prediction. Effectiveness is largely dependent on the accuracy of flow prediction. No matter how the traffic information is obtained, there will always be some difference between the predicted and the field condition. Hence, a desirable adaptive control strategy should reduce reliance on prediction as much as possible.

A reliable estimation model must be developed to provide real-time traffic information for adaptive control. Vehicle arrival information is typically obtained from detectors placed upstream of the intersection, and the objective of estimation is to obtain the vehicle travel time between the upstream detector and the intersection stop line [2, 33]. For system optimization, the ability to estimate traffic conditions for a long duration is desirable, but because of geometric constraints and uncertainties in vehicle arrivals, there is most always a tradeoff between the estimation duration and data accuracy[37].

Similarly, Optimization Policies for Adaptive Control (OPAC) defines travel time as the time for the vehicle to travel between the upstream and the downstream signals [33] and the difficulty in travel time estimation must also be dealt with in congested traffic. This problem exists in the optimization process of every adaptive control system today. However, very limited up to date work is reported in the literature as to how the varying queue length will affect the arrival time estimation[37].

The effectiveness of adaptive control strategies also relies on reasonable estimation of system parameters governing queue formation/dissipation, start-up delay, and vehicle clearance. The start-up delay and the vehicle releasing rate

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

may be different from time to time due to the influence of construction, incident, and even the weather condition. The differences cannot be accounted for if static parameters are used in the model, and the cumulated error can become large enough to offset any systems advantage over other types of control. However, most of the existing adaptive control strategies do not contain a self-adjusting mechanism.

Fuzzy logic has been used to develop an adaptive traffic signal controller, because it allows qualitative modeling of complex systems, where it is not easy to solve using mathematical models [11, 10,2,21] and is good for systems that have inherent uncertainties[18].

The fundamental idea of this paper is to design and implement a general traffic light intersection simulation environment controlled by fuzzy logic with user ability to build its own fuzzy engine, including I/O parameters, membership functions, inference rule based and outputs.

This paper mainly aims to provide hand-outs in two fields, The first is environment for Traffic intersection designrs; and the second is the framework for fuzzy logic . So, there are two primary contributions made by this paper.

1- Combination of SUMO with Fuzzy Adaptive traffic controller represents the first simulation environment that can design any traffic intersection(s) controlled by fuzzy logic (The user can implement the fuzzy control to any SUMO traffic light intersection, can choose various parameters provided by SUMO simulation output manually and can tune the fuzzy control parameters on-line. The fuzzy control make new TLS every cycle.
2- Fuzzy engine classes used in adaptive control are a general fuzzy engine. So, it can be used in any other field of fuzzy logic applications, and it represents a tool for fuzzy logic systems researchers.

II. LITERATURE REVIEW

The literature review states the Intelligence Methods in Urban Traffic Signal Control. The literature review into Fuzzy logic traffic signal control are:

Pappis and Mamdani, 1977 [2] the first fuzzy logic controller was designed for an isolated two-phase intersection, which had three inputs and one output. Decisions were made every 10 seconds to decide whether to extend green time of current phase or change the way leave to the next phase according to traffic volume of all approaches. Simulations had shown that the above method was effective. This is the earliest example applying fuzzy logic to traffic controls.

Lee et al., 1995 [10] studied the use of fuzzy logic in controlling multiple junctions. Controllers received extra information about vehicles at the previous and next junctions, and were able to promote green waves. The system outperformed a fixed controller, and was at its best in either light or heavy traffic. The controller could easily handle changes in traffic flow, but required different parameter settings for each junction.

Choi et al.,2002 [4] used fuzzy logic controllers, and adapted them to cope with congested traffic flow. Comparisons with fixed fuzzy-logic traffic light controllers indicated that this enhancement could lead to larger traffic flow under very crowded traffic conditions.

Trabia et al., 1999 [18] presented a two-stage fuzzy logic control method. The controller was designed to be responsive to real-time traffic demands. The fuzzy controller used vehicle loop detectors, placed upstream of the intersection on each approach, to measure approach flows and estimate queues. These data were used to decide, at regular time intervals, whether to extend or terminate the current signal phase. In the first stage, observed approach traffic flows were used to estimate relative traffic intensities in the competing approaches. In the second stage, these traffic intensities were then used to determine whether the current signal phase should be extended or terminated.

Zhiyong et al. 1999 [13] designed a new traffic controller for a single intersection based on the people's strategic decision process to the multi-phase signal traffic control. The inputs of controller were the queue lengths on the contiguous phase lanes and the differences between the current queue lengths and the ones in next phase lanes. The outputs of one were extending green time of the current phases or changing into the successional phases. Fuzzy reasoning rulers were created according to the experiences of traffic police. This controller had been applied practically and worked very well.

Liu et al. 1997 [14] proposed a hierarchical fuzzy control method, and applied it to the arterial coordinated control. Its fundamental principle was to use hierarchical structure and fuzzy theory to solve real time coordinated control problems of traffic trunk road. It regarded all intersections on the trunk as subsystems. According to the cycle length provided by a coordinator, it adjusted the splits of all approaches on line by using fuzzy control methods. Traffic volumes detected at all intersections were sent to the coordinator. The cycle length and splits were determined by using fuzzy control method. The goal of the coordinator and subsystems was to minimize the queue length. Simulation results showed that it could shorten the queue, and reduce total traffic delay.

Aksaç et al. 2011 [1] implemented a real time traffic simulator with an adaptive fuzzy inference algorithm that arranges the foreseen light signal duration. It changes the time duration of lights depending on waiting vehicles behind green and red lights at crossroad. The simulation has also been supported with real time graphical visualization. it creates random traffic flows according to specified parameters.

Zade And Dandekar, 2012 [36] proposed a simulation of fuzzy traffic controller using SIMULINK environment of MATLAB. Results for green light extension time in seconds are obtained with SIMULINK model of fuzzy traffic controller which shows linear increment in the green light extension time for increasing values of traffic density and traffic flow rate. The simulation results showed linearity between inputs applied to the Fuzzy Inference System and output drawn from it.

Most researchers have worked on control an isolated intersection with fuzzy control method [2, 18, 13, 32, 25, 8, 33, 34, 20]. Few of them apply this method to the coordinated control of arterial or area traffic. Area traffic coordinated

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

control system is a complex large-scale system. There are many interactional factors, and it is difficult to describe the whole system using some qualitative knowledge. It is just the limitations of fuzzy control methods. In a word, it is more appropriate to use fuzzy control methods for traffic signal control of the isolated intersection.

III. FUZZY LOGIC

The problems of uncertainty, imprecision and vagueness have been discussed for many years, particularly by philosophers. The nature of vagueness and the ability of traditional Boolean logic to cope with concepts and perceptions that are imprecise or vague have been discussed at length[72,65].

Fuzzy systems are being used successfully in an increasing number of application areas; they use linguistic rules to describe the systems. These rule-based systems are more suitable for solving the complex system problems mathematically. One of the most important considerations for designing any fuzzy system, the generation of the fuzzy rules as well as the membership functions are applied for each fuzzy set. In most existing applications, the fuzzy rules are generated by expert system in the area, especially for control problems with only a few inputs[16].

Based on the study of Hoogendoorn et al.[6] and others (e.g., Sayers et al.[23,24]), the key benefits of a fuzzy logic (FL) approach are can fuse quantitative and qualitative information, Fuzzy systems can trade off potentially conflicting objectives with the help of expert knowledge, Fuzzy control successfully provides a transparent, flexible and adaptable control structure, The transparency and intuitive nature of the rule base and input variables adopted in FL control system make it relatively easy to develop, test, and modify, FL is well suited to deal with nonlinear input/output relationships, FL can handle the model even when the model parameters are not precisely known or when no prior knowledge about the system is present at all. In fact, FL techniques use measurement data from the process.

To implement fuzzy logic technique to a real application requires the following three steps, see Fig. 1 [35]:

1. Fuzzification – converting classical data or crisp data into fuzzy data or Membership Functions (MFs)
2. Fuzzy Inference Process – combining membership functions with the control rules to derive the fuzzy output
3. Defuzzification – using different methods to calculate each associated output and putting them into a table: the lookup table. Picking up the output from the lookup table based on the current input during an application

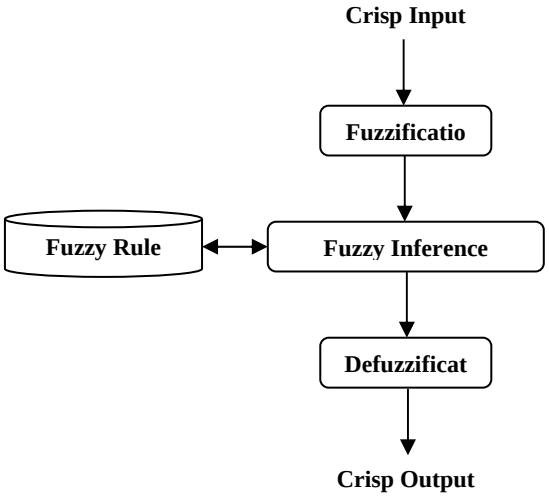


Figure 1. Structure of a Fuzzy Logic Model

IV. SIMULATION

The increasing power of computer technologies, the evolution of software engineering and the advent of the intelligent transport systems have prompted traffic simulation to become one of the most used approaches for traffic analysis in support of the design and evaluation of traffic systems. The ability of traffic simulation to emulate the time variability of traffic phenomena makes it a unique tool for capturing the complexity of traffic systems[28].

Simulation is used for scientific modeling of natural systems or human systems in order to gain insight into their functioning. It can be used to show the eventual real effects of alternative conditions and courses of action. It is also used when the real system cannot be engaged, because it may not be accessible, or it may be dangerous or unacceptable to engage, or it is being designed but not yet built, or it may simply not exist[3].

A computer simulation (or "sim") is an attempt to model a real-life or hypothetical situation on a computer so that it can be studied to see how the system works. By changing variables, predictions may be made about the behavior of the system. A good example of the usefulness of using computers to simulate can be found in the field of network traffic simulation. In such simulations, the model behavior will change each simulation according to the set of initial parameters assumed for the environment[3].

One of the systems that is best studied using a computer simulation is a traffic network. It is more common to experiment with traffic networks in a computer simulated environment because experimenting with traffic in the real environment is not practical[20].

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

V. TRANSPORTATION SYSTEM

Transportation is an essential element in the economic development of a society. Without good transportation, a nation or region cannot achieve the maximum use of its natural resources or the maximum productivity of its people. Progress in transportation is not without its costs, both in human lives and environmental damage, and it is the responsibility of the transportation engineer working with the public to develop high-quality transportation consistent with the available funds and social policy and to minimize damage. Transportation is a significant element in our national life. It accounts for about 18 percent of household expenditure and employs over 10 percent of the workforce[19].

The traffic or highway engineer must understand not only the basic characteristics of the driver, the vehicle, and the roadway, but how each interacts with the others. Information obtained through traffic engineering studies serves to identify relevant characteristics and define related problems. Traffic flow is of fundamental importance in developing and designing strategies for intersection control[19].

A. Intersection Control

An intersection is an area shared by two or more roads. Its main function is to allow the change of route directions. A simple intersection consists of two intersecting roads; a complex intersection serves several intersecting roads within the same area. The intersection is therefore an area of decision for all drivers; each must select one of the available choices to proceed. This requires an additional effort by the driver that is not necessary in non intersection areas of a highway. The flow of traffic on any street or highway is greatly affected by the flow of traffic through the intersection points on that street or highway because the intersection usually performs at a level below that of any other section of the road[19].

B. Traffic Signals

One of the most effective ways of controlling traffic at an intersection is the use of traffic signals. Traffic signals can be used to eliminate many conflicts because different traffic streams can be assigned the use of the intersection at different times. Since this results in a delay by vehicles in all streams, it is important that traffic signals be used only when necessary. The most important factor that determines the need for traffic signals at a particular intersection is the intersection's approach traffic volume, although other factors such as pedestrian volume and crash experience may also play a significant role. The Manual on Traffic Signal Design gives the fundamental concepts and standard practices used in the design of traffic signals[19]. In addition, the Manual on Uniform Traffic Control Devices (MUTCD) describes eight warrants in detail, at least one of which should be satisfied for an intersection to be signalized. However, these warrants should be considered only as a guide; professional judgment based on experience also should be used to decide whether or not an intersection should be signalized. The factors considered in the warrants are: (Warrant 1- Eight-hour vehicular volume, Warrant 2- Four-hour vehicular volume, Warrant 3- Peak hour, Warrant 4- Pedestrian volume, • Warrant 5- School crossing, Warrant 6- Coordinated signal system, Warrant 7- Crash experience, • Warrant 8- Roadway network)

The warrants described earlier will help the engineer only in deciding whether a traffic signal should be used at an intersection. [26]. however, there are numbers of terms commonly used in the design of signal times, these are (Controller, Cycle, Phase, Interval, Offset, Change and clearance interval, All-red interval, Peak-hour factor, lane group, Critical lane group, Saturation flow rate) [26].

A traffic signal that is properly designed and timed can be expected to provide benefits for the orderly and efficient movement of people, and effectively maximizes the volume movements served at the intersection. The degree to which these benefits are realized is based partly on the design and partly on the need for a signal. A poorly designed signal timing plan or an unneeded signal may make the intersection less efficient, less safe, or both[28].

C. Traffic controller (Types of Operation)

Despite the many variations in their design, traffic signals can be classified according to operational type as (Pre-Timed (or fixed time), Semi Actuated, Fully Actuated, Adaptive Traffic Signal Controls) .

D. Adaptive Traffic Signal Control Overview

Adaptive traffic signal control is a concept where vehicular traffic in a network is detected at an upstream and/or downstream point and an algorithm is used to predict when and where the traffic will be and to make signal adjustments at the downstream intersections based on those predictions. The signal controller utilizes these algorithms to compute optimal signal timings based on detected traffic volume and simultaneously implement the timings in real-time. This real-time optimization allows a signal network to react to volume variations, which results in reduced vehicle delay, shorter queues, and decreased travel times. [28].

All adaptive control systems are driven by a similar conceptual process[42]:

1. Collecting data in real-time from sensor systems to identify traffic conditions.
2. Evaluating alternative signal timing strategies on a model of traffic behavior.
3. Implementing the "best" strategy according to some performance metric.
4. Repeating steps 1,2,3 again and again.

Each adaptive system is distinguished by how it uses different components or approaches to these four key steps in the control of the traffic system[42].

The search methodology itself is not typically important for signal timing, but how the traffic engineer is able to constrain or influence the search of alternative timing strategies is critical to the success of adaptive system deployment. For example[28]:

• The minimum and maximum phase lengths.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

• Which phase sequences are allowed or disallowed.

• How rapidly or slowly the system allows timing parameter changes.

Other parameters, specific to individual adaptive systems, are key elements to guide the adaptive system in searching appropriate and effective signal timing strategies[28].

E. Data Collecton

With the ever growing traffic demand and rising challenge of controlling it, our networks are equipped with different sensors to measure the actual traffic state. This data is essential for evaluating the performance of transportation systems and for supporting the development of new approaches and technologies that address traffic problems.

F. Detection Technologies

Optimally, consideration of detector technology should involve the following three factors (Cost, Practicality, Accuracy) [41]:

G. Trafic Simulation (SUMO - Simulation of Urban Mobility Traffic Simulation)

One of the systems that is best studied using a computer simulation is a traffic network. It is more common to experiment with traffic networks in a computer simulated environment because experimenting with traffic in the real environment is not practical[5]. The following are well known and widely used traffic simulation packages (SUMO - Simulation of Urban Mobility [3], Quadstone Paramics [22], Treiber's Microsimulation of Road Traffic [30] , Aimsun, [31], Trafficware SimTraffic, [29], CORSIM TRAFVU) [15]

SUMO is an open source software gives users the right to use the software free of charge; a feature that is not very common in commercial software packages. However, open source projects are becoming more and more popular because they give their users the right to use, study and modify the program without any restriction.

Out of the six software packages (Sumo, Treiber's Microsimulation of Road Traffic, Quadstone Paramics Modler, Aimsun, Trafficware SimTraffic, CORSIM TRAFVU) , only the first two are free to use [3, 30], while the other 4 are paid [22, 32,29,15]. While the free packages could be investigated to their full potential, we could only study the demo versions of the four commercial packages. The demo versions had 30 days usage restrictions and/or feature restrictions.

Another feature of the free software packages (SUMO and Treiber's Micro-simulation of road traffic) is that their source codes are freely available for download and use. The difference between the two packages is that SUMO is actually an open source project that is being developed by two different institutions, while Treiber's Microsimulation is a personal software project whose source code has been made available. The source code is not available for study, modification or research for users of the other four commercial software packages (Paramics Modeller, Aimsun, SimTraffic and CORSIM). One of the most popular features of open source projects is that they can be further modified by other programmers, as already mentioned previously. This feature supports the potential for parallelizing simulaion models and packages to explore high end computer systems.

VI. Fuzzy Adaptive Traffic Signal System

The mainly preferred feature in a traffic controller at an intersection is that it should be adaptive to any changes in the traffic demand. In case of the traffic controllers that are usually used, the relative durations of the green phases are determined by computer programming based on the traffic pattern at an intersection. But these traffic controllers are not adaptive because the settings can only be altered manually or by computer commands sent by the traffic control center. This problem is solved by using Fuzzy Adaptive Traffic Signal System (FATSS), which is capable of signaling adaptively at an intersection. The objective of this section is to propose a design methodology for modeling fuzzy controllers.

The main role of FATSS is to optimize the green phase duration using the fuzzy engine which uses the traffic demand variables as input variables and membership functions. Then, the fuzzy engine initiates the rule base that depends on input variables and according to outputs generated by fuzzy engine. The simulation program SUMO will change the phase durations to be suitable for new demand state. The general structure of the fuzzy traffic Signal system UML(Unified Modeling Language) Activity diagram will have the structure shown in Fig. 2

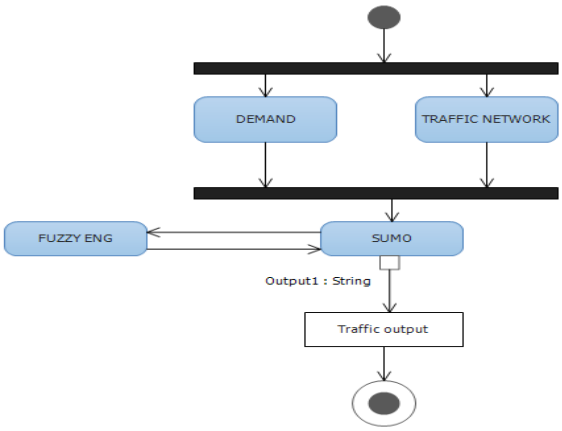


Figure 2. UML Activity Diagram of FATSS

A. FATSS inputs overview

The proposed system is based on multiple inputs which are necessary to run SUMO simulator like traffic network description and demand.

1. Traffic Network

SUMO uses an own road network description. Although being readable (xml) by human beings, at a coarse scale, SUMO road networks are directed graphs. "Nodes" represent intersections/junctions, and "edges" roads/streets. Then is the use of NETCONVERT [3] tool to build the network file.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

- Node Descriptions

Inside the node files, normally having the extension ".nod.xml" ,every node is described in a single line which looks like this:

<node id="<STRING>" x="<FLOAT>" y="<FLOAT>" [type="<TYPE>"]/>

The straight brackets ('[' and ']') indicate that the parameter is optional. Each of these attributes has a certain meaning and value range explained in table I.

Table I. Node Attributes Description

Attribute Name	Value Type	Description
id	id (string)	The name of the node
x	Float	The x-position of the node on the plane in meters
y	Float	The y-position of the node on the plane in meters
type	enum ("priority", "traffic_light", "right_before_left", "unregulated")	An optional type for the node

The following types are possible. Any other string is counted as an error and will yield in a program stop:

- **priority:** Vehicles on a low-priority edge have to wait until vehicles on a high-priority edge have passed the junction.
- **traffic_light:** The junction is controlled by a traffic light.
- **right_before_left:** Vehicles will let vehicles coming from their right side pass.
- **unregulated:** The junction is completely unregulated - all vehicles may pass without braking; this may result in additional incidents.

A complete file should like the xml code 1, which is the node file "fuzzy_node.nod.xml"

XML code 1: node description

```
<nodes>
 <node id="5" x="0.0" y="0.0" type="traffic_light" />
 <node id="2" x="-500.0" y="0.0" />
 <node id="3" x="0.0" y="-500.0" />
 <node id="4" x="+500.0" y="0.0" />
 <node id="1" x="0.0" y="+500.0" />
</nodes>
```

Only the first node named "5", which is the node in the center of the network, is a traffic light controlled intersection. All other nodes are uncontrolled. You may also notice, that each of both ends of a street needs an according node. This is not really necessary as you may see soon, but it eases the understanding of the concept: every edge (street/road) is a connection between two nodes (junctions).

- Edge Descriptions

The edge is a basic part of a road network, like a normal street which connects two nodes. Within the edges file, each description of a single edge looks like this:

<edge id="*<STRING>*" from="*<NODE_ID>*" to="*<NODE_ID>*" [type="*<STRING>*"] [numLanes="*<INT>*"] [speed="*<FLOAT>*"] [priority="*<UINT>*"] [length="*<FLOAT>*"] [shape="*<2D_POINT>*[*<2D_POINT>*]*"] [spreadType="center"] [allow="*<VEHICLE_CLASS>*[*<VEHICLE_CLASS>*]*"] [disallow="*<VEHICLE_CLASS>*[*<VEHICLE_CLASS>*]*"]/>.

The beginning and the end nodes are defined using their IDs (from="*<NODE_ID>*" to="*<NODE_ID>*"). Each edge is unidirectional and begins at the "from" node and ends at the "to" node. If a name of one of the nodes can not be dereference, because they have not been defined within the nodes file, an error is generated. For each edge, some further attributes should be supplied, such as the number of lanes the edge has (numLanes), the maximum speed allowed on the edge speed, and the priority may be defined optionally.

The length of this edge will be computed as the distance between the starting and the end point. As an edge may have a more complicated geometry, you may supply the edge's shape within the shape attribute. If the length of the edge is not given otherwise, the distances of the shape elements will be summed.

Table II lists an edge's attributes. The priority plays a role during the computation of the way-giving rules of a node. Normally, the allowed speed on the edge and the edge's number of lanes are used to compute which edge has a greater priority on a junction. Using the priority attribute, you may increase the priority of the edge making more lanes yielding in it or making vehicles coming from this edge into the junction, not waiting.

- **Traffic Light Program**

Usually, NETCONVERT produces traffic lights and programs for intersections during the generation process of the networks. But these computed programs differ quite often from those found in real world. To supply the simulation with real traffic light programs, it is possible to load extra programs. In addition, SUMO/SUMO-GUI allows to load definition which describe when and how a set of traffic lights can switch from one program to another. The user can load new definitions for traffic lights as a part of additional files.

As soon as loaded, the last program will be used. Switching between programs is possible via Weekly Switch Automatism WAUTs and/or TraCI. Also, one can switch between them using the GUI context menu. Table III shows attributes/elements of traffic light signal (TLS). Each phase is defined using the attributes in table IV.

Table II: Edge's Attributes Description

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

Attribute Name	Value Type	Description
Id	id (string)	The name of the edge
From	referenced node id	The name of a node within the nodes-file the edge shall start at
To	referenced node id	The name of a node within the nodes-file the edge shall end at
Type	referenced type id	The name of a type within the SUMO edge type file
numLanes	int	The number of lanes of the edge; must be an integer value
Speed	Float	The maximum speed allowed on the edge in m/s; must be a floating point number (see also "Using Edges' maximum Speed Definitions in km/h")
Priority	int	The priority of the edge
Length	Float	The length of the edge in meter
Shape	List of positions; each position is encoded in x,y (do not separate the numbers with a space!) in meters	The start and end node are omitted from the shape definition; an example: <edge id="e1" from="0" to="1" shape="0,0 0,100"/> describes an edge that after starting at node 0, first visits position 0,0 than goes one hundred meters to the right before finally reaching the position of node 1
spreadType	enum ("right", "center")	The description of how to spread the lanes; "center" spreads lanes to both directions of the shape, any other value will be interpreted as "right"
Allow	list of vehicle classes	Explicitly allows the given vehicle classes (not given will be not allowed). Vehicle classes must be separated using ' '.
Disallow	list of vehicle classes	Explicitly disallows the given vehicle classes (not given will be allowed). Vehicle classes must be separated using ' '.

Table III: TLS attributes/elements.

Attribute Name	Value Type	Description
id@tlLogic	id (string)	The id of the traffic light
type	enum (static, actuated, agentbased)	The type of the traffic light
programID	id (string)	The id of the traffic light program; Please note that "off" is reserved, see below.
offset	int	The initial time offset of the program

Table IV : Tls Attributes Description

Attribute Name	Value Type	Description
duration@phase	time (int)	The duration of the phase
state@phase	list of signal states	The traffic light states for this phase, see below

Each character within a phases' state describes the state of one signal of the traffic light. A single lane may contain several signals - for example one for vehicles turning left, one for vehicles which move straight. This means that a signal does not control lanes, but links - each connecting a lane which is incoming into a junction and one which is outgoing from this junction. Table V explain signal colors.

Table V: TLS signal colors.

Character	Description
R	'red light' for a signal - vehicles must stop
y	'amber (yellow) light' for a signal - vehicles will start to decelerate if far away from the junction, otherwise they pass
g	'green light' for a signal, no priority - vehicles may pass the junction if no vehicle uses a higher priorized for stream, otherwise they decelerate for letting it pass
G	'green light' for a signal, priority - vehicles may pass the junction

After having defined a TLS program as above, it can be loaded as an additional file; of course, a single additional file may contain several programs. It is possible to load several programs for a single TLS into the simulation. The program loaded as last will be used (unless not defined differently using a WAUT description). All sub keys of the additional programs must differ if they describe the same TLS. It is also possible to load a program which switches the TLS off by giving the programID the value "off", shown in xml code bellow.

```
<tlLogic id="0" type="static" programID="off"/>
```

B. *SUMO-GUI Simulation*

After completing the input files for SUMO-GUI ,which are described in last sections (Node File, Edges File, Traffic Light Program) ,now the simulation is ready to run. Fig. 3, shows the SUMO-GUI window

Figure 3. : SUMO-GUI window

C. *On-Line Simulation Information Retrieval*

TraCI is the short term for "**Tra**ffic **C**ontrol **I**nterface", see Fig. 4. It provides the access to a running simulation, and allows to retrieve values of simulated objects and to manipulate their behaviors "on-line". TraCI uses a TCP based client/server architecture to provide access to SUMO. Thus, SUMO acts as a server that is started with additional

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

command line options(remote-port) then SUMO will listen to for incoming connections.When started with the remote-port option, SUMO only prepares the simulation and waits for an external application(Traffic Fuzzy Engine), that takes over the control. After starting SUMO, a client connects to SUMO by setting up a TCP connection to the appointed SUMO port.

The client application sends commands to SUMO to control the simulation run, to influence single vehicle's behavior or to ask for environmental details. SUMO answers with a *Status*-response to each command and additional results that depend on the given command. The client is responsible for shutting down the connection using the close command. The system will use TraCI to retrieve input parameters for traffic fuzzy engine. Table VI presents an example for information that can be retrieved using TraCI commands. The information retrieved in table VI will be summarized to derive input parameters for traffic fuzzy engine.

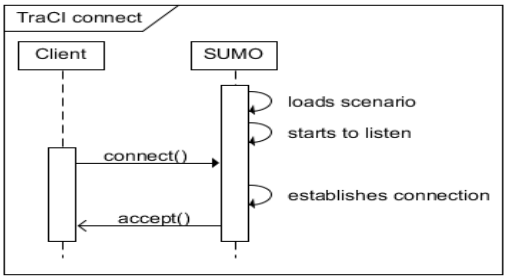


Figure 4. TraCI establishing a connection to SUMO

Table VI: Retrievable Induction Loop Variables

Variable	ValueType	Description
id list (0x00)	stringList	Returns a list of ids of all induction loops within the scenario (the given Induction Loop ID is ignored)
position (0x42)	double	Returns the position of the induction loop at its lane, counted from the lane's begin, in meters.
lane ID (0x51)	string	Returns the ID of the lane the induction loop is placed at.
count (0x01)	int	Returns the number of induction loops within the scenario (the given Induction Loop ID is ignored)
last step vehicle number (0x10)	integer	Returns the number of vehicles that were on the named induction loop within the last simulation step [#];.
last step mean speed (0x11)	double	Returns the mean speed of vehicles that were on the named induction loop within the last simulation step [m/s]
last step vehicle ids (0x12)	stringList	Returns the list of ids of vehicles that were on the named induction loop in the last simulation step
last step occupancy (0x13)	double	Returns the percentage of time the detector was occupied by a vehicle [%]
last step mean vehicle length (0x15)	double	The mean length of vehicles which were on the detector in the last step [m]
last step's time since last detection (0x16)	double	The time since last detection [s]
last step's vehicle data (0x17)	complex	A complex structure containing several information about vehicles which passed the detector

D. Input Parameters For Traffic Fuzzy Engine

The previous section describes the method to retrieve information from running simulation (On-line). Fig. 5 shows two parameters (queue length and waiting time) that can be derived from retrieved information to become input parameters for traffic Fuzzy engine.

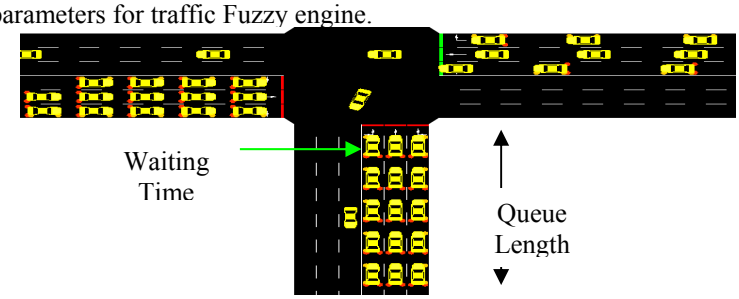


Figure 5. Simulation State Variable

1. Queue Length

The queue length value mean number of vehicles stopped in a lane behind the stop line at a traffic signal waiting for green phase at time *t* [27]. or the sum of arriving vehicles during the red light time period and the remaining vehicles at the end of previous green light period. During the red interval, the queue of vehicles waiting at the intersection begins to increase. The queue reaches its maximum length at the end of the red interval. When the signal changes to green, the queue begins to clear as vehicles depart from the intersection at the saturation flow rate . For a given time, the difference between the arrival pattern and the service pattern is the queue length.

2. Waiting Time

Waiting time of a single vehicle t_w is basically the time from its arrival to the intersection till the beginning of the green light phase for the vehicle's lane. If an arriving vehicle has green light, then waiting time equals zero [17].The user can summarize the waiting time as total waiting time or average waiting time for one lane or group of lanes for one direction or for all directions.

3. Arrival Flow Rate (Veh/Sec)

The mean of a statistical distribution of vehicles arriving at a point or uniform segment of a lane or roadway.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

E. Traffic Fuzzy Engine (TFZENG)

While SUMO simulation is running, the Traffic fuzzy engine can connect as client to SUMO simulation and read the necessary parameters to start the engine which starts to read input parameters like Queue Length (QL) , build membership function and create linguistic variables for it. Fig. 6 explains the UML component diagram for SOMU and Traffic fuzzy engine.

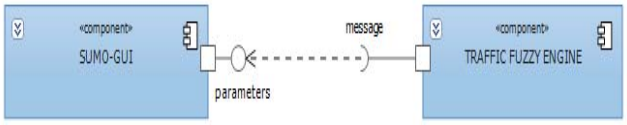


Figure 6. UML Component Diagram for SOMU and TFE.

Because the inputs are real numbers (crisp values) ,so the first step in TFZENG is to represent the input parameter as linguistic variable and build the membership functions according to specific ranges using **Fu**zzy **L**inguistic **V**ariable (FZLV) class which represents the first class in TFZENG class library. The next section will explain each class in TFZENG and its effort in engine.

1. Traffic Fuzzy Engine (TFZENG) Classes Description

The TFZENG includes 4 classes which corporate to receive input parameters from simulation and process it to give the crisp value as output returned to control the simulation. **Fu**zzy **L**inguistic **V**ariable (FZLV) is the first class that manipulates the input parameters. This class is responsible for building the linguistic variables for each one of input parameters and it dominates FZMF class. So the Fuzzy Membership Function (FZMF) is a member in FZLV. Each input parameter is defined as an instance of FZLV and because AFC receives real numbers from simulation, it must be converted to a linguistic variable.

The FZLV class is responsible for fuzzification process which converts the numeric value received from simulation to linguistic variables through triangular fuzzifier. At the same time, it will add membership functions with the possibility of changing values during simulation; a feature that provides to the user the possibility of tuning fuzzy engine Online. Each membership function has 4 values for triangular and trapezoidal which represent the values of the function for each input parameter from simulation more than one membership functions.

All membership functions belonging to traffic fuzzy engine belong to a special class FZMF. It is the class which is responsible for creating membership functions where each membership functions is an instance of the class FZMF. After the configuration of input linguistic variables with their membership functions, the user will configure the output linguistic variable and its membership functions which will be used later in the defuzzification process. Defuzzification class responsible for converting the output linguistic variable to crisp values (green times). The defuzzification process is done using the most widely used centroid method. After the defuzzification process, the traffic fuzzy engine will build TLS (traffic light signal) table relying on new times resulting from Traffic fuzzy engine and sends the table to change simulation signals and repeat previous operations per cycle.

2. Traffic Fuzzy Engine (TFZENG)Classes Implementation

All TFZENG classes are created by C# and the following sections will explain the implementation of each class

- **Fuzzy Linguistic Variable (FZLV)**

This class is responsible for creating the linguistic variable instances for each input and output parameters. The algorithm 1 describes the working of FZLV class.

Step 4 of algorithm 1 explains that if the current FZLV instance does not have a membership function, the algorithm will jump to another class (FZMF class) which is a member of FZLV class so the new instance of FZMF will be a member of FZLV current instance.

Algorithm 1: FZLV Class Implementation Algorithm

Input: Name(string),Value.
Output: FZLV instance (Fuzzified Linguistic Variable).
Begin
  Step 1: Compare name with all existing instance names.
  If found return error "naming instance"
  Step 2: Create FZLV instance Name.
  Step 3: Set LV value.
  Step 4: Does current LV have Membership Function
    Yes: Fuzzify the LV value step 5.
    No : Create new FZMF instance for current LV.
  Step 5: ***FOR*** each membership function in current LV ***Do***
    If range1 current LV between range1 and range2
    return (value – range1) / (range2 – range1);
    else if value between range2 and range3
      return 1;
    else if value between range3 and range4
      return (range4 - Value) / (range4 – range3);
    else return 0;
End

- **Fuzzy Membership Function (FZMF)**

The Fuzzy Membership Function is responsible for creating a membership function set for current FZLV instance.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

So all functions of this class are called by current FZLV instance. Algorithm 2 describes the work of FZMF.

Algorithm 2: FZMF Class Implementation Algorithm

Input: MF name , Range1, Range2, Range3, Range4.
Output: FZMF instance.
Begin
Step 1: Compare MF name with all existing instances name.
If found return error "naming instance"
Step 2:
A = Range3 – Range2;
B = Range4 - Range1;
C = Range2 – Range1;
S=((2 * A * C) + (A * A) + (C * B) + (A * B)
+ (B * B)) / (3 * (A + B)) + Range1
Step 3:
M = S – Range1;
N = Range4 – Range1;
O=(value * (N + (N - (M * value)))) / 2
return O;
End

In steps 2 and 3 of algorithm 2, the FZMF calculates the centroid value for current membership function ranges and return it to the current FZLV instance.

3. Fuzzy Rules (FZR)

Fuzzy Rules are the class that is responsible for creating and manipulating the fuzzy rule base which is used by fuzzy inference system. The main function in this class is to manipulate IF-THEN strings. So this class will receive fuzzy rules from user and then validates them with current FZLV and FZMF instances. Algorithm 3 describes the main functions of FZR class.

Algorithm 3: FZR Class Implementation Algorithm

Input: IF-THEN string.
Output: FZR instance.
Begin
Step 1: Is the Structure of IF-THEN string complete
No: error " IF-THEN structure error"
Step 2: for each FZMF instance
Search antecedent in FZMF name
Not Found :"error antecedent name"
Step 3:
for each FZMF instance
Search consequent in FZMF name
Not Found :"error consequent name"
Step 4: return.
End

Algorithm 3 concerns with validation of rule statements. Firstly, it validates statement structure to assure that IF and THEN parts are found. Then step 2, assures that antecedent name (IF condition part) is found in FZMF instances. Finally step 3 assures that consequent name (THEN part) is found in FZMF instances. Fig. 7 shows the GUI for adding, removing and editing the fuzzy rules. Each variable (input/output) represented by ComboBox its value item collections belong to FZMF instances for a specific variable. The AFC reads it automatically from FZMF existing instances. All Fuzzy rules will be activated by AFC program after user rules activation command button "Active". And because a fuzzy rule can have multiple antecedents, so the user can use And/Or CmboBox control to add and/or between antecedent variables. The fuzzy operator (AND or OR) is used to obtain a single number that represents the result of the antecedent evaluation. If the user uses this Fuzzy Rules Builder during simulation running all edited (tuned) rules will be active for the next cycle. So the new rules will affect the phases time of the next cycle (not the current cycle).

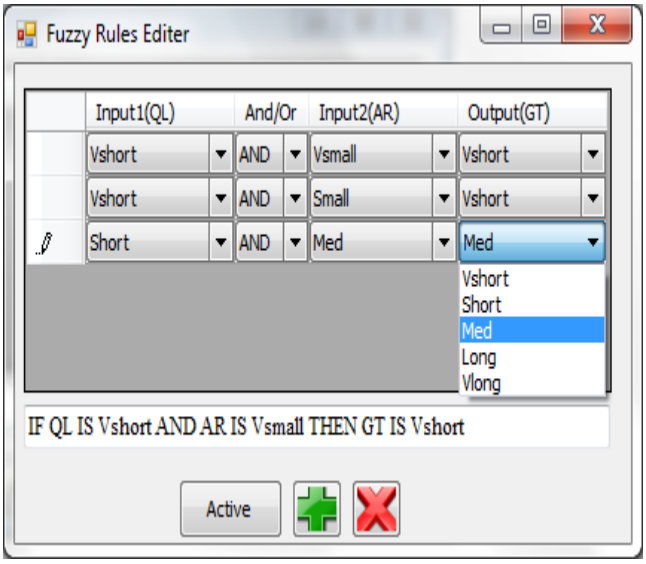


Figure 7. Fuzzy Rules Builder GUI.

- **Traffic Fuzzy Engine (TFZENG)**

Traffic Fuzzy Engine class is the main class in the fuzzy engine and it is responsible for the defuzzification process. All other classes in the engine are members in this class. So, TFZENG class can inherit any instance of the classes. Algorithm 4 describes the TFZENG class work. The TFZENG class algorithm 4 explains that the engine depend on Center of Gravity Method (COG) to calculate the output value of fuzzy engine. Algorithm 5 shows the overall operation step to calculate the output value from input value using all fuzzy engine classes.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

Algorithm 4: TFZENG Class Implementation Algorithm

Input: FZLV instance name.
Output: Defuzzification crisp value.
Begin
NUM=0;DOM=0
Step 1: ***For*** each FZMF in FZLV instance name ***Do***
Calculate FZMF.value (algorithm 42)
NUM=NUM+ FZMF.S * FZMF.O
DOM=DOM+FZMF.O
Step 2: COG=NUM/DOM
Step 3: return COG.
End

Algorithm 4.5: All Fuzzy Classes Implementation Algorithm

Input: input crisp values.
Begin
Step 1: ***For*** each input and output value ***Do***
Create FZLV instance (algorithm 1).

Step 2: ***For*** each FZLV instance ***Do***
Create FZMF(algorithm 2).

Step 3: Create FZR instances(algorithm 3)

Step 4:Defuzzify (algorithm 4).
End

F. AFC implementation

Section (A.1 Traffic network) describes the classes used by AFC program to control the simulation. This section will describe the implementation of AFC program and the way to control the SUMO simulation. The AFC program begins with loading configuration file which determines the traffic network that will be simulated. The configuration file is a SUMO specific XML file that contains a description and information about simulation needed files. The configuration file is responsible for specifying the TraCI remote port to connect to SUMO simulation. After loading the configuration file, the simulation traffic network needs a demand now to start the simulation. The demand can be chosen by AFC by using load demand window (Fig. 8). The user can choose any demand file to start the simulation. Fig. 9 shows the overall diagram of AFC with SUMO.

Now the simulation is ready to run by one click on "start simulation" button . The user also can choose the simulation output file or the simulation will start with default. Indeed, the simulation will load network and demand, then pauses, waiting for user controlling commands which user can manipulate by window form, which shows the number of steps that the user can jump in simulation or specify the input parameters which the user want to send to the Traffic Fuzzy Engine. Also the window form can display a specific simulation output according to Checked Variable (i.e. Queue Length, Arrival Rate and Waiting Time) at the same window. Fig. 10 shows an example of retrieved information from running simulation which represents simulation time until step 250 stopped, checking only Queue length and Waiting Time parameters, and retrieving only Queue length and Waiting Time parameters.

Depending on the current parameter, the user can build the linguistic variable and membership functions by click "Edit" button for each parameter. The membership function will depend on retrieved values. For instance the maximum Queue length retrieved from simulation will be the max range value for membership functions. Figure 11 shows the user interface for Build or Tune the membership function and it provide the following operations:

- Adding new membership function to current input parameter.
- Deleting existed member function from current input parameter.
- Drawing membership functions.
- Saving membership functions with values to XML file.
- Loading membership functions with values from Exist XML file.
- Editing specified membership function values (tuning).

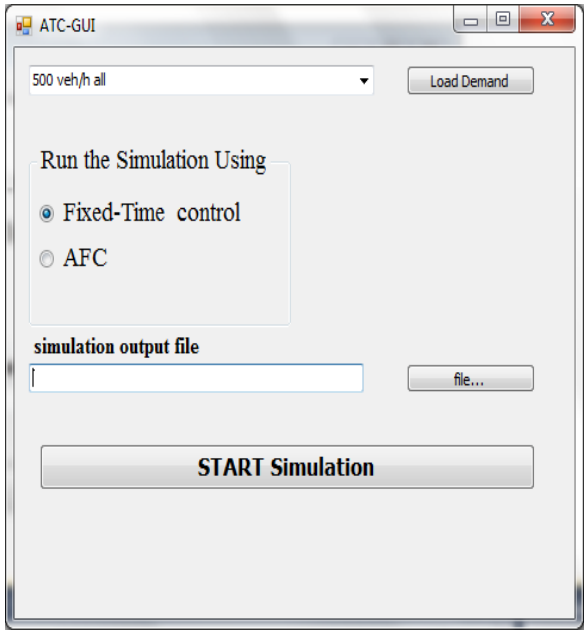


Figure 8. AFC Load Demand Window

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

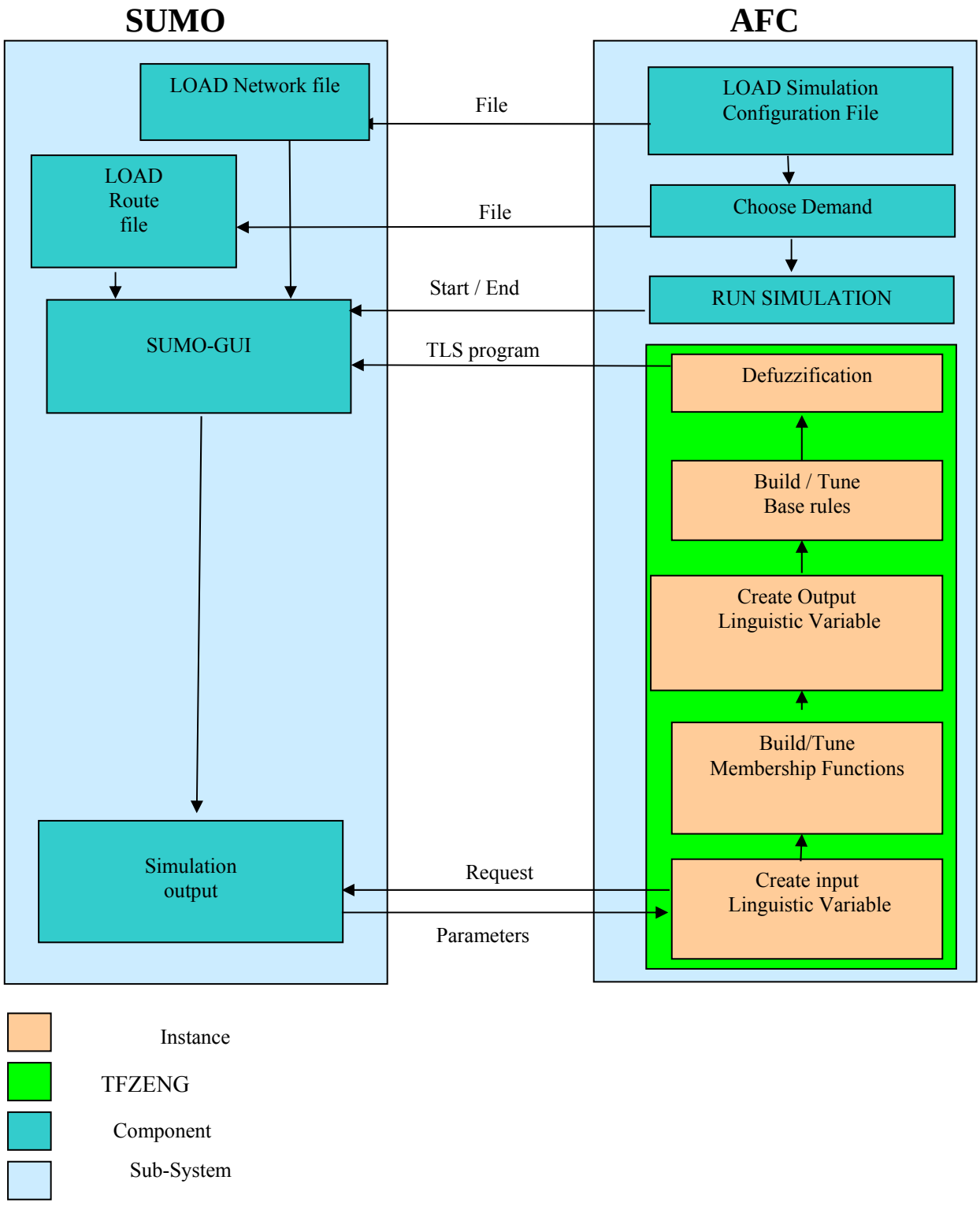


Figure 9 : Overall diagram of AFC with SUMO

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

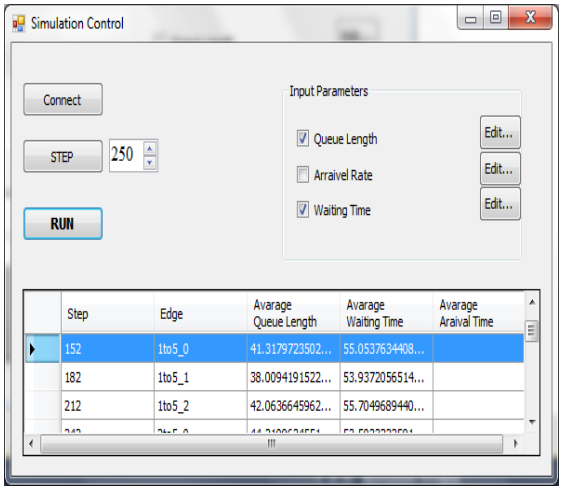


Figure 10 : AFC Simulation Control Window (retrieved information)

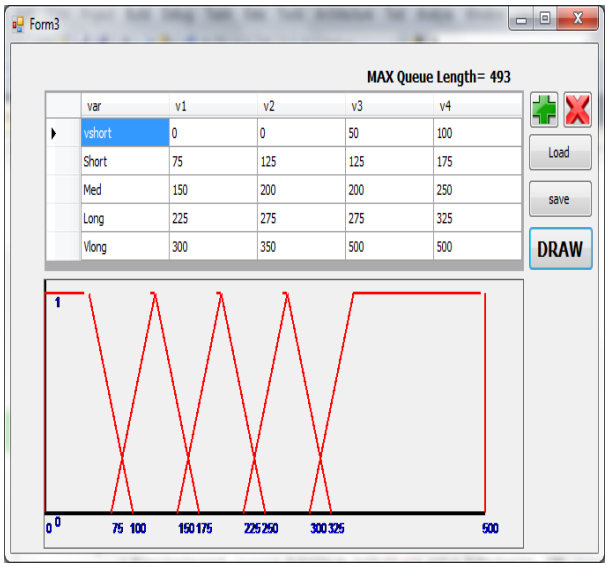


Figure 11 : AFC membership functions edit/Tune

VII. CASE STUDY AND RESULTS

A. *Case Study Description*

The first requirement for starting the simulation for study case is the traffic network(intersection). Returning to section 5.1.1, the network will describe the roads and nodes for simulation case. Fig. 12 shows the study case traffic intersection network. The intersection contains four approaches and each approach contains six lanes (Three reaching the intersection and three leaving the intersection).
Each lane contains two inductive loop detectors one at the head of lane queue and one at the upstream of lane , which are responsible for reading lane queue status and sending information to AFC program.

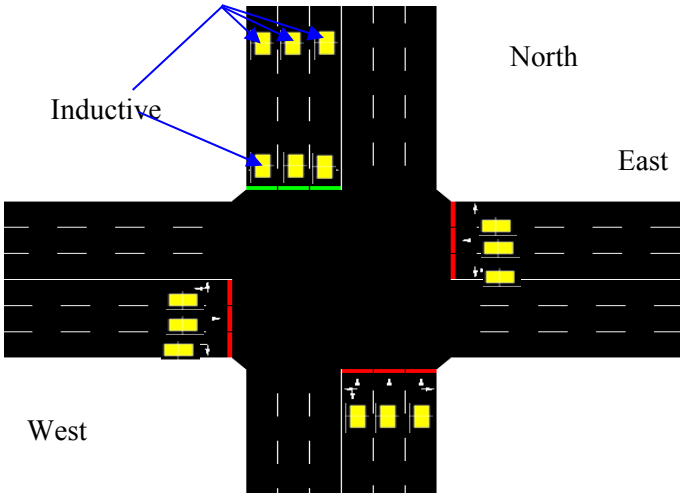


Figure 12. Case Study Traffic Light Intersection

It is usually easy to identify by observing the lengths of queues at the intersection . Sensors should be installed in multiple lanes at locations where the lane 's volume changes with the time of the day, Traffic volumes should be measured on each lane of an intersection approach. Average waiting time per cycle needs to be computed for each lane and sent to AFC as input parameters. A timer starts when a vehicle enters the detection zone of the queue sensor and resets to zero when the vehicle exits the zone of detection. If the system counts a predetermined number of seconds, the queue of vehicles waiting at the red signal lane extends upstream to the queue sensor, a vehicle will be located over the loop longer than the selected delay time. The inductive loop must be long enough to span the distance between standing vehicles. Concomitantly, it must be shorter than the shortest gap in moving traffic so that the breaks between moving vehicles will cause the delay timer to reset. So the distance between the first inductive loop and the second depends on case study traffic properties and behavior.

All data retrieved from SUMO simulation are saved to a database designed for AFC program. The database will ease the work with data to retrieve the specific information and summarize the information to be useful for Fuzzy engine. Because the SUMO is a Microscopic traffic flow simulation, it retrieves information about each car. So, AFC must converts this information to macroscopic parameters(like Volume). Algorithm 6 explains the work of AFC with Database to retrieve the important parameters.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

Algorithm 6: AFC DATA MANIPULATION

Input: data record(retrieved by Traci command).
Output: average waiting time.
Begin
Step 1: ***For*** each vehicle in Red signal lanes ***Do***
Vehicle.wait_time= Now - Vehicle.arrival
Wait=Wait+Vehicle.wait_time.

Step 2: ***For*** each vehicle in Green signal lanes ***Do***
Vehicle.wait_time= Vehicle.departure - Vehicle.arrival
Wait=Wait+Vehicle.wait_time.

Step 3: Avarage waiting time= wait/vehicle number

End

B. Demand Generation

The second step to make simulation work is to create a demand. So, the AFC program can create any demand by describing the volume of a given time interval for each intersection direction . All demands will be created before simulation starts and saved to xml routes file. The demand generation program will generate cars and generates routes for those cars.

C. Simulation Running

As soon as the demand file is created, traffic light simulation can start. Firstly TLS program is fixed-time for the first cycle only. The starting phases time is setup by GUI form window in Fig. 13 which enables the user to setup the times for first simulation phase. And then AFC will generate the times for all simulation phases.

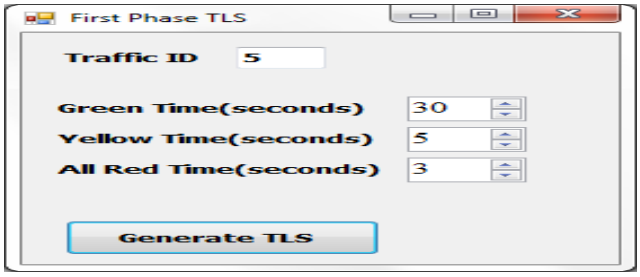


Figure 13. First Phase TLS Times.

D. Reading Variables

Two variables will be used in case study, namely QL and AR, SQL statement will be used to summarize the retrieved information from SUMO for each direction approach. All retrieved information will be stored to DataGridView control, then it can be manipulate as data Re cordSet to select the QL and AR variables.

E. Creating Linguistic Variables and Ranges

Now it is time to specify the ranges of case study linguistic variables. Noticing that QL has five linguistic values (VShort, Short, Medium, Long and VLong. AR variable also has five ranges linguistic values(VSmall, Small, Med, Large and VLarge). Table VII Explains the linguistic variables and their ranges. In practice, all linguistic variables, linguistic values and their ranges are usually chosen by the domain expert.

Table VII: Input Linguistic Variables and Ranges

Linguistic variable	Linguistic value	Ranges	
		From	To
QL	VShort	0	100
	Short	75	175
	Medium	150	250
	Long	225	325
	VLong	300	500
AR	VSmall	0	20
	Small	10	40
	Med	30	60
	Large	50	80
	VLarge	70	100

F. Determining Fuzzy Sets

Fuzzy sets can have a variety of shapes. However, a triangle or a trapezoid can often provide an adequate representation of the expert knowledge, and at the same time significantly simplifies the process of computation. Fig.14 and 15 show the fuzzy sets for QL and AR variables used in the case study respectively. As noticed, one of the key points here is to maintain sufficient overlap in adjacent fuzzy sets for the fuzzy system to respond smoothly.

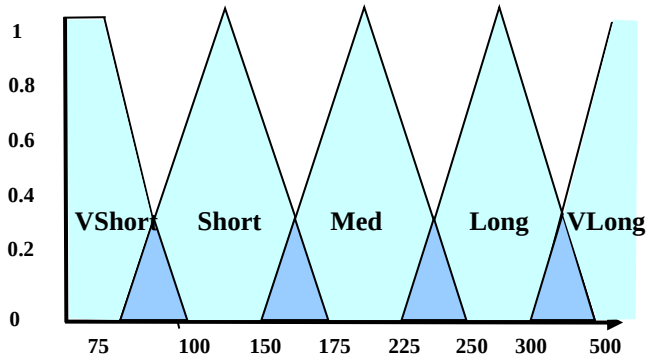


Figure 14. QL Fuzzy Sets

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

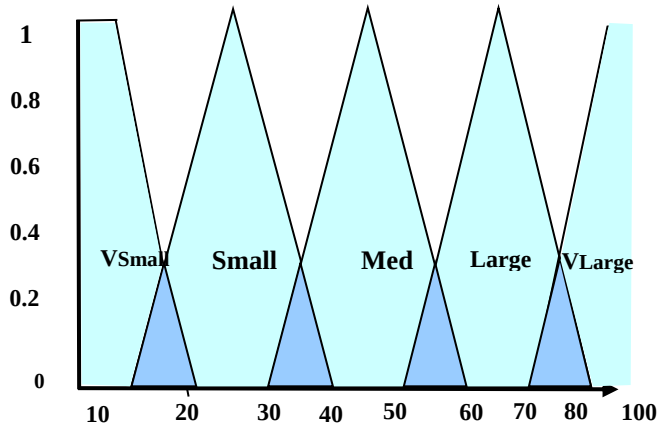


Figure 15 : AR Fuzzy Sets

G. *Constructing Fuzzy Rules*

The next step in the case study is to obtain fuzzy rules. To accomplish this task, the user might ask the expert to describe how the problem can be solved using the fuzzy linguistic variables defined previously. Required knowledge also can be collected from other sources such as Manuals and books. The two input variable fuzzy sets may enable us to derive 25 rules that represent complex relationships between two variables used in AFC. Table VIII contains these rules.

Table XIII: Fuzzy Rules Matrix

QL / AR	VShort	Short	Medium	Long	VLong
VSmall	VSHORT	VSHORT	SHORT	MED	LONG
Small	VSHORT	SHORT	MED	LONG	VLONG
Med	SHORT	SHORT	MED	LONG	VLONG
Large	SHORT	SHORT	MED	LONG	VLONG
VLarge	MED	MED	LONG	VLONG	VLONG

There are two inputs and one output variable in the case study. It is often convenient to represent fuzzy rules in a matrix form. A two-by-one system (two inputs and one output) is depicted as an M × N matrix of input variables. The linguistic values of one input variable form the horizontal axis and the linguistic values of the other input variable form the vertical axis. At the intersection of a row and a column lies the linguistic value of the output variable. From table VIII it is Clear that the output linguistic variable has fuzzy sets(VSHORT, SHORT, MED, LONG and VLONG) and fig. 16 shows the GT fuzzy sets. Table IX explains the GT output Linguistic Variable and its Ranges.

Table IX : GT Linguistic Variable and Ranges

Linguistic variable	Linguistic value
	VSHORT
	SHORT
	MEDIUM
	LONG
	VLONG

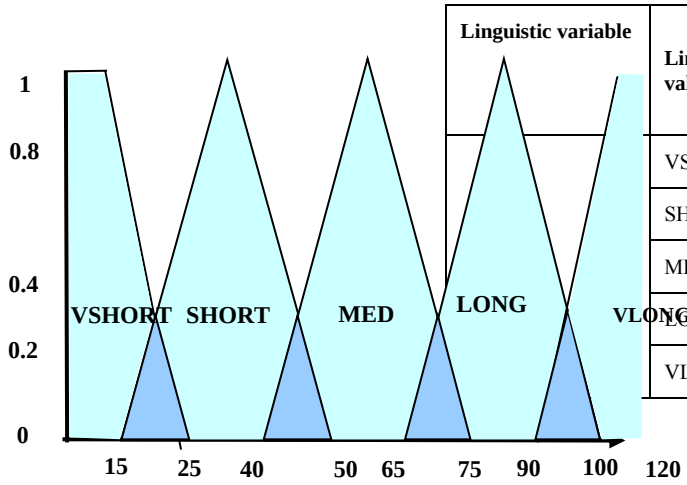


Figure16: GT Fuzzy Sets

H. *First Phase Optimization*

After all variables are read , linguistic variables are created, fuzzy sets and ranges are setup, simulation will begin with first cycle and before the cycle ends, the AFC will read simulation lane parameters and begin optimization.

The first variable to read is QL. AFC reads variable at simulation step 130 and converts it to linguistic instance. Fig. 17 shows the fuzzification of QL value=154 and its corresponding values in membership functions.

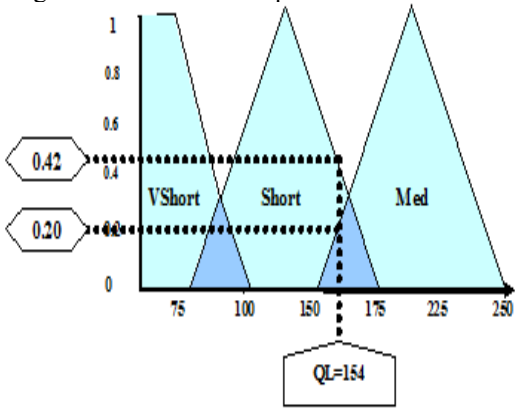
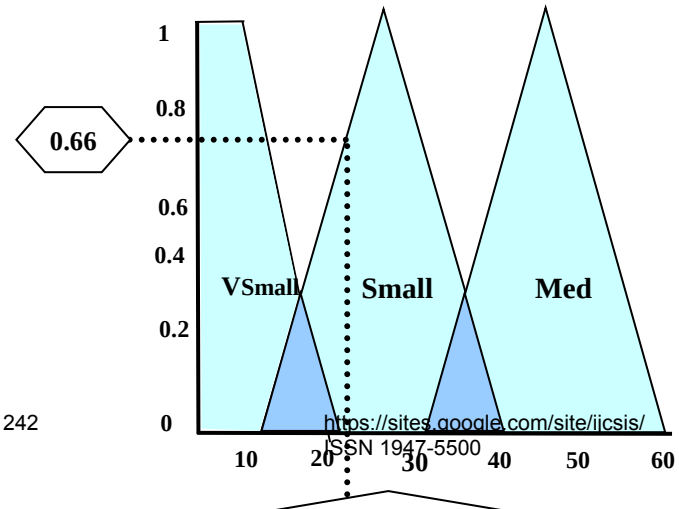


Figure 17 QL Fuzzification.

At the same simulation step 130 ,AR variable will be read and fuzzified by AFC. Fig. 18 shows AR variable value=22 and its corresponding value in membership functions.



https://sites.google.com/site/ijcsis/
ISSN 1947-5500

Figure 18: AR Fuzzification

After the fuzzification process for two input variables, rule evaluation process begins by AFC depending on previously built fuzzy rules. Fig. 19 explain that the two input values will belong to two fuzzy rules:

1) IF QL IS Short AND AR IS Small THEN GT IS SHORT
2) IF QL IS MED AND AR IS Small THEN GT IS MED

Each fuzzy rule will be evaluated separately by AFC, Fig. 19 shows the operation of rule evaluation for fuzzy rule1(19.a) and fuzzy rule2(19.b).

After the rules evaluation process done in fig. 19, aggregation process is needed. Aggregation is the process of unification of the outputs of all rules. In other words, membership functions of all rule consequents will be taken and cliped them into a single fuzzy set. Thus, the input of the aggregation process is the list of clipped consequent membership functions, and the output is one fuzzy set for each output variable. Fig.20 shows how the output of each rule is aggregated into a single fuzzy set for the overall fuzzy output.

The last step in the fuzzy inference process is defuzzification. Fuzziness helps us to evaluate the rules, but the final output of a fuzzy system has to be a crisp number. The input for the defuzzification process is the aggregate output fuzzy set and the output is a single number. There are several defuzzification methods, but probably the most popular one is the centroid technique. It finds the point where a vertical line would slice the aggregate set into two equal masses. In a computerized program, the COG is calculated over a continuum of points in the aggregate output membership function and this calculation may be slow for real time adaptive systems, but to speed up the process, a reasonable estimate can be obtained by calculating it over a sample of points. Fig. 21 shows the Defuzzifying of GT variable fuzzy set by a sample of points. For GT variable fuzzy set sample of points in fig. 21. The calculated COG value is

$$COG = (25+30+35+40+45) * 0.42 + (50+55+60+65+70)*0.20 / (0.42 * 5 + 0.20 * 5)$$

$$COG \approx 43$$

Thus, the result of defuzzification, crisp output GT, is 43. It means the next phase Green Time for current approach will be changed from 30 to 43 seconds. The retrieved crisp variables and defuzzification process crisp values for current phase optimization can be monitored by users using GUI special form shown in Fig. 22. Noticing that GT value =38 retrieved by AFC differs from GT value=43 calculated in example (fig. 21). That is because COG value retrieved by AFC is calculated for all points in GT output fuzzy sets. So it is more accurate. All the optimization operations explained for first phase will be repeated for all next phases for all approaches until simulation end time

I. Demand Flows

In order to evaluate the effectiveness of the fuzzy logic system, the case study carried out two types of flow, namely

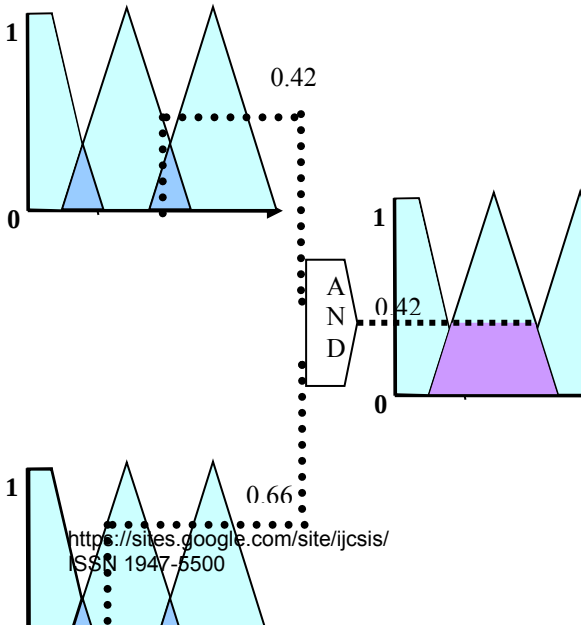
- case 1 (A,B,C,D) traffic flow (vph) is steady for all directions
- case 2 (A,B,C,D) traffic flow (vph) differs for each direction

CASE 1

A simulation run is made for 60 min periods with different traffic flow (vph) for each case (A,B,C,D) to produce the output average delay of vehicles. The results for case 1 show that generally the proposed fuzzy logic system and the fixed time controller produce little difference (best rate Case 1-C =1.7%) in results. See table X that shows the simulation results for case 1.

Table X: Case 1 Results

Case	Average Delay (sec)		IMPROVEMENT (%)
	FIXED TIME SYSTEM	FUZZY LOGIC SYSTEM	
Case 1-A 1500 vehs for each direction (N, E, S, W)	30.8	30.3	**1.6**
Case 1-B 1200 vehs for each direction (N, E, S, W)	27.3	26.9	**1.5**
Case 1-C 1000 vehs for each direction (N, E, S, W)	17.8	17.5	**1.7**
Case 1-D 500 vehs for each direction (N, E, S, W)	**12.9**	**12.7**	**1.6**



https://sites.google.com/site/ijcsis/
ISSN 1947-5500

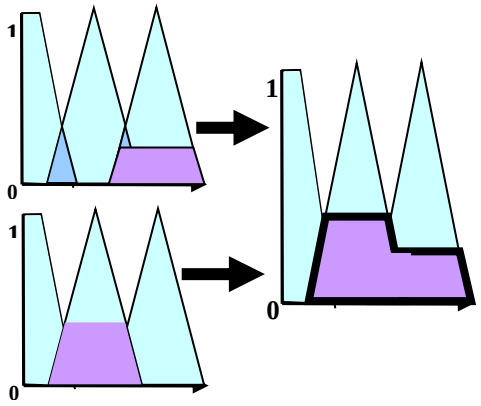


Figure 20: Aggregation of the rule outputs

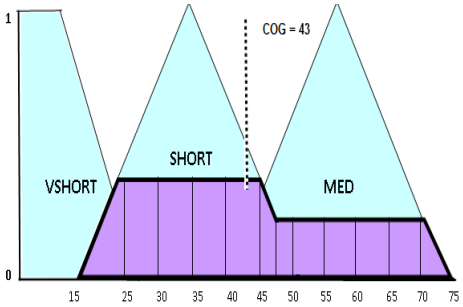


Figure 21: Defuzzifying of GT by a sample of points

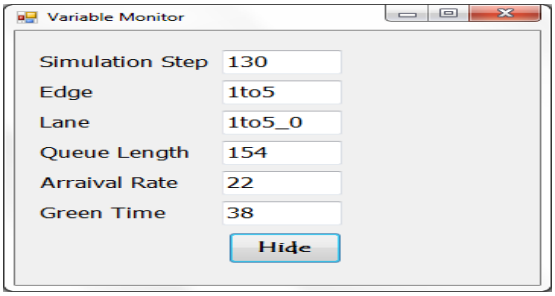


Figure 22: Variables Monitor GUI.

Figure 23 gives a graphical representation of the average waiting time of the cars, proposed fuzzy logic system and the fixed time controller produce little difference results.

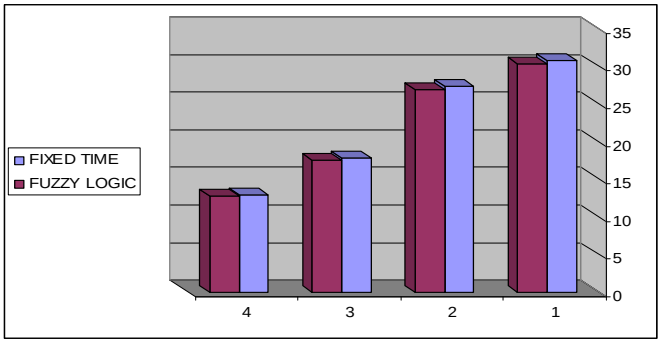


Figure 23: Performance Chart Case 1 (Waiting Time).

CASE 2

A simulation run is made for 60 min periods with different traffic flow (vph) for each case (A,B,C,D) to produce the output average delay of vehicles. This case has unbalanced traffic for each direction. Some times one direction has a heavy traffic and another has a light traffic. The results for case 2 indicate that the proposed fuzzy logic system performs much better (best rate Case 2-A =20.04 %) than the fixed time controller. See table XI which shows the simulation report for case two. Figure 24 gives a graphical representation of the average waiting time of the cars, proposed fuzzy logic system performs much better than the fixed time controller.

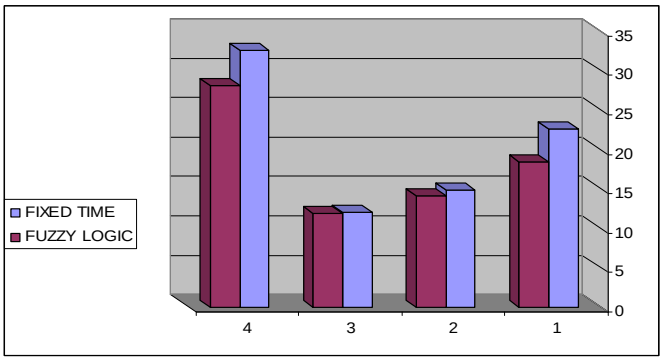


Figure 24: Performance Chart Case 2 (Waiting Time).

For Case 2 , over all time reduction can be calculated by multiplying average waiting time of one vehicle by the number of cars per hour, then it can be supposed that each 1 hour idle car consumes as average 2.5 liter of petrol [7], and each liter of petrol contains 2640 grams of CO_2 [7]. Table XII shows the amount of reductions in Time, Fuel and CO_2 .

Table XII: Case 2 time, fuel and CO2 reduction results.

Case	Average Delay (sec)		Time reduction (seconds)	Fuel amount (liter)	CO_2 kg
	FIXED TIME SYSTEM	FUZZY LOGIC SYSTEM			
Case 2-A N=1000,E=2000, S=1500,W=1000	22.5	18.4	22550	15.65	**42.3**
Case 2-B N=500,E=1000, S=750,W=500	14.8	14.1	1925	/	/
Case 2-C N=300,E=500, S=700,W=400	12.0	11.8	380	/	/
Case 2-D N=1500,E=1000, S=1200,W=700	**32.5**	**28.1**	**19360**	**13.44**	**35.5**

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

Case 2-A example:

For case 2-A, the over all reduction in time, fuel and emission reduction can be calculated for one hour:
22.5-18.4=4.1 sec average waiting time reduction
4.1 sec * 5500 vehicles=22550 sec≈ 6.26 hour waiting time reduction for all 5500 car
6.26 h* 2.5 liter = 15.65 liter fuel reduction
15.65 * 2640=41316 grams≈ 42.3 Kg CO2 reduction

For fuel reduction ,supposing there are 4 rush hours every day, so 365 will have 1460 rush hours.
1460 rush Hour * 15.65 **liter per hour=22849 liter per year**
22849 liter per year * 450 Iraqi Dinar =10282050 I.D. per year, The amount of money approximately 10 million I.D. only for one traffic light.

There are about 150 traffic light intersections in Baghdad only. so in simple calculation it is 1.5 Billion I.D. ≈ 1.2 million dollar per year. And these numbers will be increased for big cities like Istanbul which has about 1200 traffic intersections. Finally, the problem does not concern reducing the waiting time for drivers or reducing the cost for fuel. It rather concerns with the environmental factors because reduction of fuel consumption leads to reduction of emissions which affect the environment in a great way. When fuel is burned, it releases CO2 which is a greenhouse gas into the air and contributes to the greenhouse effect. The greenhouse effect is when these greenhouse gases(such as Co2) trap energy from the sun ; just like in a greenhouse, and raise the temperature of the earth; thus causing glaciers to melt and ocean currents to change, and eventually resulting in the end of the planet.

7 Conclusions

The intersect between traffic control and fuzzy logic can be rewarding, by understanding the fuzzy logic strengths to improve the design of high performance traffic control systems. The conclusions which have been accessed from AFC are Fuzzy system is suitable for traffic control for two major reasons

- fuzzy logic is well suited for controlling a process or system that is too nonlinear or too poorly understood to use conventional control designs.
- fuzzy logic allows control technicians to implement control tactics used by human operator. Human operators formulate their control tactics based on imprecise information described in linguistic terms. No processing of quantitative information is involved.

The performance of AFC is affected by the configuration of the membership functions of input/output variables and the rule base., Reducing waiting time between 2% to 20% has the following benefits, as it, increases throughput on the network, reduce dangerous vehicle emissions, which affect human health and environment, and reduce fuel consumption which affects the improving economy.
Open source - cost free – projects (like SUMO) are becoming more and more popular because they give their users the right to use, study and modify the program without any restriction or cost. **At last** Using SUMO reduces the efforts, time and cost that spent by researchers to develop a traffic simulation environment.

The visions about the future works As obviously, this paper are focuses on Fuzzy Logic only, for a future proposal to design a hybrid system that mixes between Fuzzy Logic and other common artificial techniques(like Neuro-fuzzy) to obtain the optimization, or making a comparison between these Fuzzy Logic and other artificial techniques

References

1. Alper Aksaç · Erkam Uzun · Tansel Özyer,A real time traffic simulator utilizing an adaptive fuzzy inference mechanism by tuning fuzzy parameters, Springer Science+Business Media, LLC 2011.
2. C. P. Pappis, and E. H. Mamdani. A Fuzzy Logic Controller for a Traffic Junction. IEEE Transactions on Systems, Man, and Cybernetics, 1977.
3. Centre for Applied Informatics (ZAIK) and the Institute of Transport Research at the German Aerospace Centre: Simulation of urban mobility sumo website. Available at http://sumo.sourceforge.net/, 2009.
4. Choi, W., Yoon, H., Kim, K., Chung I., Lee S., A traffic light controlling FLC considering the traffic congestion. In Pal, N. and Sugeno, M., editors, Advances in Soft Computing — AFSS 2002, International Conference on Fuzzy Systems 2002 ,pages 69–75.
5. G. Kotusevski and K.A. Hawick, A Review of Traffic Simulation Software, Computer Science, Institute of Information and Mathematical Sciences, Massey University, Albany, NS 102-904, Auckland, New Zealand 2009.
6. Hoogendoorn S, Hoogendoorn-Lanser S, Schuurman H. Fuzzy perspectives in traffic engineering, Workshop on Intelligent Traffic Management Models, Delft, 1999.
7. Information Guide No.2 "idle reduction" ,United State Environmental Protection Agency (EPA), 2010.
8. KIM Jong-wan. A fuzzy logic control simulator for adaptive traffic management. FUZZ-IEEE Proceedings, 1997(3), pp.1519-1524.
9. Lawrence A Klein, Sensor Technologies and Data Requirements for ITS, ISBN 158053077X, Artech House Publisher, Boston, London, Jan 1, 2001.
10. Lee, J. H., LEE, K. M. AND LEEKWANG, H. , Fuzzy controller for intersection group. Int. IEEE/IAS Conference on Industrial Automation and Control, Taipei, Taiwan 1995. pp. 376-382.
11. Lee, J., Lee, K., Seong, K., Kim, C., and Lee-Kwang, H. , Traffic control of intersection group based on fuzzy logic. In Proceedings of the 6th International Fuzzy Systems Association World Congress 1995 , pages 465–468.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

12. Levinson, D. , The value of advanced traveler information systems for route choice. Transportation Research Part C: Emerging Technologies 2003.
13. Liu Zhiyong, Zhu Jin, Li Ziupin, Yin Zhengqi; A Multi-phase Fuzzy Control Method Used for Single Intersection. Information and Control, 1999,28(6): pp. 453-458.
14. LIU Zhiyong, WU JINPEI, LI XIUPING; Hierarchical Fuzzy Control for Urban Traffic Trunk Roads. Journal of Highway and Transportation Research and Development, 1997, 14(3): pp.17-23.
15. McTrans Moving Technology: Thesis: Traffic software integrated system website. Available at http://mctrans.ce.u.edu/featured/tsis/, 2009.
16. Md. Shabiul Islam, Bhuyan M. S., Azim M. A., Teng L. K., Othman M. ; Hardware Implementation of Traffic Controller using Fuzzy Expert System, International Symposium on Evolving Fuzzy Systems, September, 2006.
17. Mladen Antunović, Hrvoje Glavaš, FUZZY LOGIC APPROACH FOR TRAFFIC SIGNALS CONTROL OF AN ISOLATED INTERSECTION, Faculty of Electrical Engineering Osijek 2005.
18. Mohamed B. Trabia, et al. A Two-stage Fuzzy Logic Controller for Traffic Signals. Transportation Research Part C, USA, 1999(7), pp. 353-367.
19. Nicholas J. Garber,Lester A. Hoel, Traffic and Highway Engineering FOURTH EDITION 2009.
20. Niittymäki J., Fuzzy Traffic Signal Control: Principles and Applications, Ph.D thesis, Helsinki University of Technology, Finland, 2002.
21. NIITTYMAKI, J. AND PURSULA, M., Signal control using fuzzy logic. Fuzzy Sets and System 2000. 116, pp. 11-22.
22. Quadstone Paramics: Quadstone paramics website. Available at http://www.paramics-online.com/, 2009.
23. Sayers T M, Bell M G H, Mieden T, et al. Improving the traffic responsiveness of signal controllers using fuzzy logic. IEE Colloquium on Urban Congestion Management, 1995.
24. Sayers T M, Bell M G H, Mieden T, et al. Traffic responsive signal control using fuzzy logic—a practical modular approach.. In: Proceedings of the 4th European Congress on Intelligent Techniques and Soft Computing, 1996, 2159–2163.
25. T. K. Ho. Fuzzy Logic Traffic Control at a Road Junction with Time-Varying Flow Rates. IEE Electronic Letters,1996, 17(32): pp. 1625-1626.
26. The Manual on Uniform Traffic Control Devices (MUTCD), U.S. Department of Transportation, Federal Highway Administration, 2000.
27. Traffic Control Systems Handbook, Office of Transportation Management, Federal Highway Administration, 2005.
28. Traffic Signal Timing Manual, U.S. Department of Transportation, Federal Highway Administration, 2008.
29. Trafficware: Trafficware website. Available at http://www.trafficware.com/, 2009.
30. Treiber, M.: Microsimulation of road traffic applet. Available at http://www.traffic-simulation.de/, 2009.
31. TSS - Traffic Simulation Systems: Aimsun website. Available at http://www.aimsun.com/site/, 2009.
32. Xu Dongling, et al. A Fuzzy Controller of Traffic Systems and Its Neural Network Implementation. Information and Control, 1992, 21(2): pp. 74-78.
33. Xu Dongling, et al. A Method for Real Time Traffic Fuzzy Control of a Single Intersection. Information and Control, 1997, 26(3): pp. 227-233.
34. Xu Jianmin, et al. A New Fuzzy Control Method for Isolated Intersection Traffic Management. Journal of South China University of Technology (Natural Science), 2000.
35. Ying Bai, Hanqi Zhuang and DaliWang, Advanced fuzzy logic technologies in industrial applications. - (Advances in industrial control), Springer-Verlag London Limited 2006.
36. Zade And Dandekar, Simulation of Adaptive Traffic Signal Controller in MATLAB Simulink Based On Fuzzy Inference System, International Journal of Computer Applications, National Conference on Innovative Paradigms in Engineering & Technology NCIPET-2012.
37. Shao Chun,"Adaptive Control Strategy For Isolated Intersection And Traffic Network", University of Akron, May, 2009.
38. Wiering Marco, Jelle van Veenen,Jilles Vreeken,Arne Koopman, Intelligent Traffic Light Control, April 2003.
39. Taha M., Ibrahim L., Traffic Simulation System Based on Fuzy Logic, Procredia Computer Science, Complex Adaptive System Puplication , Conference Organied by Missouri University of Science and Technology, 2012.
40. Taha M, Traffic Traffic Simulation System Based on Adaptive Fuzzy control, PhD Thesis University of Mosul, College of Computer Sciences and Math. , 2013.
41. Michael S.,Randy M.l, Development of a Phase-by-Phase, Arrival-Based, Delay-Optimized Adaptive Traffic Signal Control Methodology with Metaheuristic Search, University of Texas at Austin,2006.
42. Traffic Signal Timing Manual, U.S. Department of Transportation, Federal Highway Administration, 2008.