CS4001 - Review of an Intelligent Traffic Lights Control System

Maxime Mulamba Ke Tchomba [18341916]
Xander De Jaegere [18341938]

December 9, 2018

1 Introduction

In this report we study the practical application of Fuzzy Logic in the design of an intelligent traffic lights control system as described by (Khalid 1996). The system aims to improve flows of traffic by adjusting the activation periods of lights colour at intersections depending on the load of cars currently in circulation.

2 Description of product/service

2.1 Design

The system considers a four-way intersection, with traffic coming from the four possible directions. For ease of understanding, only the West to East lane and the North to South lane are considered because the same reasoning holds for any other combination of crossing lanes.

As illustrated on Figure 1, each traffic light is associated with two electromagnetic sensors. The first one, located under the traffic light itself, detects cars actually passing the traffic light while the second one, put at a predetermined distance D [1] away from the traffic light detects cars

[1]this distance is chosen depending on the traffic flow pattern in the area

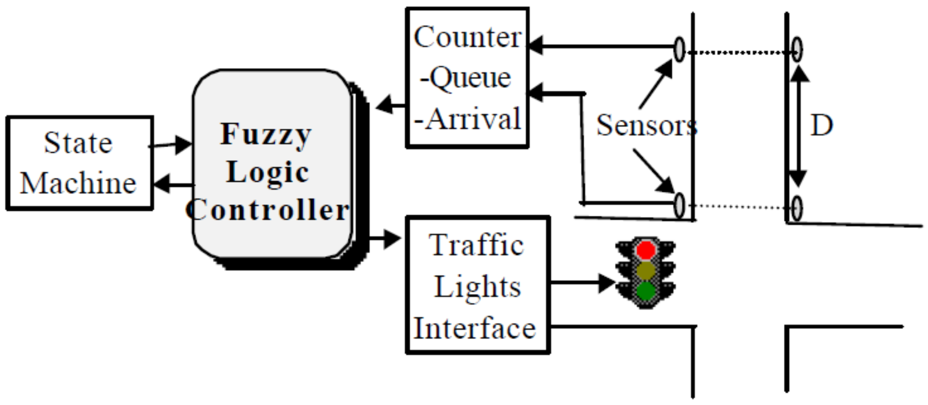

Figure 1: Structure of the traffic light control system (from Khalid 1996)

arriving to the intersection. A Fuzzy Logic Controller takes input from all sensors and can infer from them the amount of traffic arriving from a specific direction as well as the amount of traffic currently waiting at a specific traffic light by computing the difference in count of vehicles between its two sensors.

The controller outputs an extension time for the green light state, bounded between a minimum and a maximum value[2].

Each possible light configuration for the intersection is a state. The State Machine iterates through each state in a cycle with respect to instructions it receives from the Fuzzy Logic Controller.

2.2 Assumptions

Following (Khalid 1996) indications, the solution provided by this controller system works under the following assumptions:

1. The Fuzzy Logic Controller is able to distinguish traffic of the North-to-South lane from the traffic of the West-to-East lane.
2. If traffic is going down the North-to-South lane, traffic from the West-to-East lane has to stop (and vice-versa).
3. Left and right turns are ignored.

2.3 Linguistic variables and Fuzzy Base Rules

If the traffic is moving on the first lane while waiting on the second one, the authors of (Khalid 1996) define the `Arrival` and the `Queue` respectively as the quantity of vehicles moving and the quantity of vehicle waiting. Each of them can have four different levels of intensity.

The `Extension` is defined as the additional amount of time the current green light has to stays on. This variable is the consequent and also has four linguistic values while `Arrival` and `Queue` are the antecedents.

Arrival		**Queue**		**Extension**	
Almost None	AN	Very Small	VS	Zero	Z
Few	F	Small	S	Short	SO
Many	MY	Medium	M	Mid long	ML
Too Many	TMY	Large	L	Long	LO

Table 1: Linguistic variables and their linguistic values (with their corresponding shorthand notation).

The set of all linguistics values and variables are displayed on the Table 1. These linguistics variable enables us to write down the rules governing the system in the form :

```
IF there are MANY cars on the arrival side
AND the queue size is MEDIUM
THEN extend the green light for a SHORT duration
```

		Arrival			
		AN	**F**	**MY**	**TMY**
Queue	**VS**	Z	SO	ML	LO
	S	Z	SO	ML	ML
	M	Z	Z	SO	ML
	L	Z	Z	Z	SO

Table 2: Fuzzy Relationship table, using shorthand notation provided in Table 1.

The authors of (Khalid 1996) provide an more concise way to write the 16 rules by representing them in a matrix-like form, as displayed in the table 2.

3 Fuzzification, Composition, De-fuzzification

3.1 Fuzzification

For our model we have two inputs, the amount of cars waiting and the amount of cars arriving. These get fuzzified in correspondance with the membershipfunctions in Figure 2. So for example if we input 6 cars in queue and 12 in arrivals, we would have the following labels arrivals.MY, arrivals.TMY, queue.S, and queue.M. These linguistic variables are then used for the inference and composition.

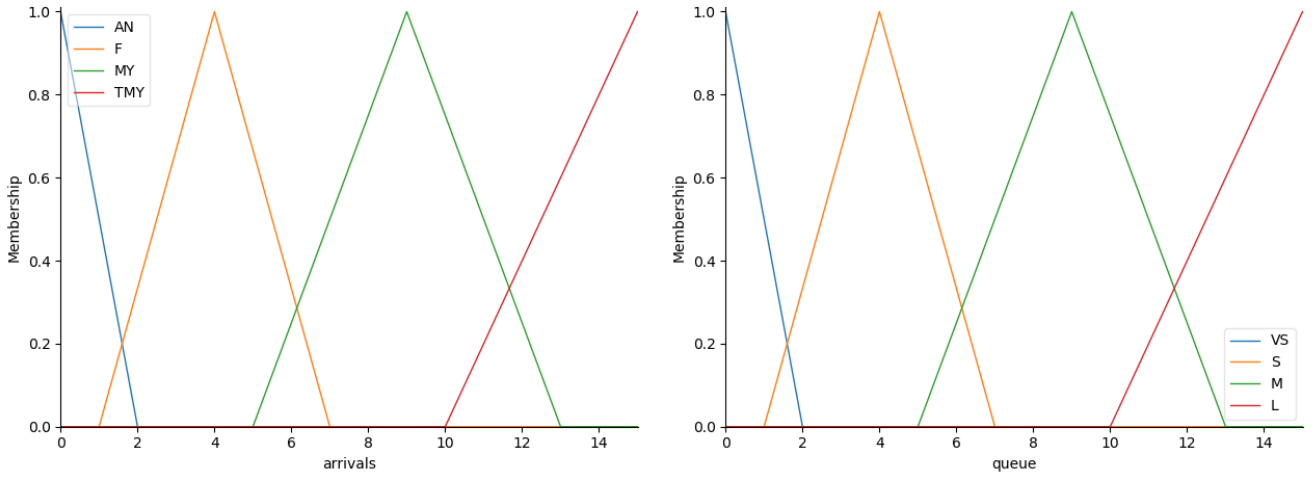

Figure 2: Fuzzy sets

3.2 Composition inference

As we have more than one rule firing with this example we can't immediately defuzzify our output. We first need to check which rules fire and what their alpha-level cut is. The rules we need to check are the rules described in 2. The rules that fire are:

[2]which again will depend on the spatial configuration of the area

```
IF A.MY and Q.S THEN E.ML
IF A.MY and Q.M THEN E.SO
IF A.TMY and Q.S THEN E.ML
IF A.TMY and Q.M then E.ML
A = arrivals
Q = queue
E = extension
```

So the alpha levels with 6 cars in queue and 12 arriving are 0.25 for E.SO and 0.33.. for E.ML. With this we're able to defuzzify our aggregated output function.

3.3 De-fuzzification

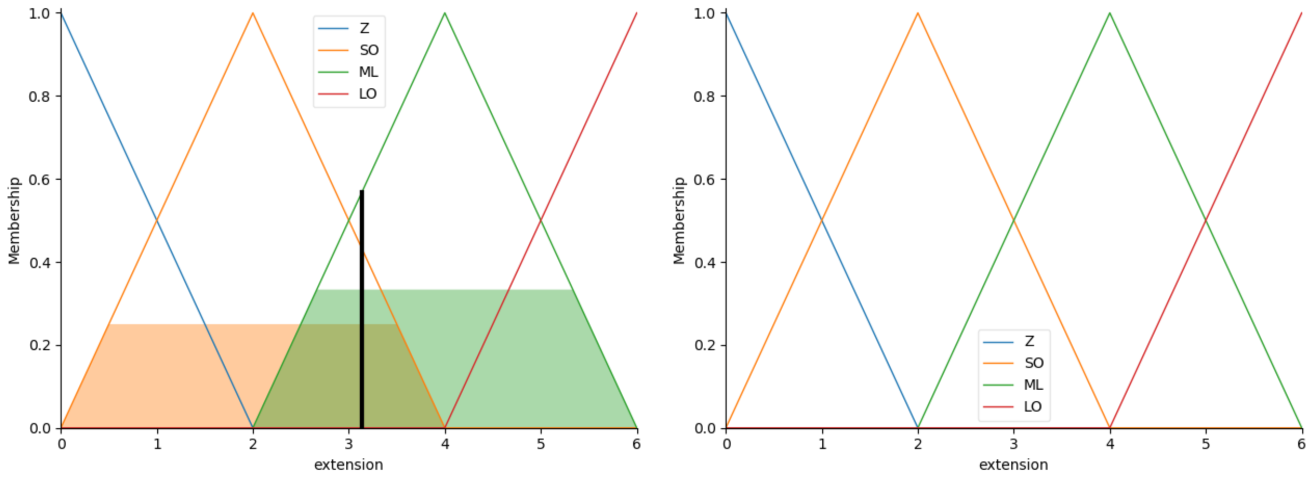

Figure 3: Defuzzification of 3 cars arriving and 1 in queue

Figure 3 represents the fuzzy sets of extension and the aggregated function we calculated from the composition and inference section. To get a crisp value we use the centroid method, calculating the center of gravity (COG) under the charts area and the value we receive for our example is 3.137 seconds.

4 Simulation

4.1 Code Structure

We implemented a simulation of the model in `python`, relying on the `Scikit-Fuzzy` library developed by (Warner et al. 2017) for the fuzzy logic part. The code is organized as follows:

trafficLightFuzzyController.py contains all the fuzzy logic related code.

models.py contains the models of traffic light, states and controller used in the simulation.

road.py defines classes used to represent roads and vehicles.

simulation.py is the main simulation script. More information is available on the provided README.md file.

4.2 Description

After implementing this fuzzy controller into the system it is important to verify that this improves the flow of traffic in the intersection. To test this we built a simulation that models a simplified situation.

Each lane has the capacity for 50 cars and the queue sensor can count up to 15 cars between itself and the sensor at the beginning of the intersection. Each step of the simulation, there is a 50 % chance that a new car appears in the North-to-South lane, against a 20 % chance for the West-to-East lane. During a step if a car has the possibility to move forward it will always advance. So the minimum wait time for each car is 50 seconds.

The simulation ends when there have been more than 50 cars in each lane or when the simulation has computed more than 400 steps. After running the simulation 50 times both with the fixed controller and the fuzzy controller the average total wait time for the fuzzy controller was 16% shorter (it took 84% of the time that the fixed controller took).

Controller type	**5 simulations**	**50 simulations**	**500 simulations**
fixed controller	1117 seconds	1078 seconds	1086 seconds
fuzzy controller	960 seconds	906 seconds	907 seconds

Table 3: Results of simulation

5 Conclusion

From the results we can conclude that a fuzzy logic controller for traffic lights can optimize traffic flow on the intersection which leads to more efficient time use for every commuter and visitor. To implement this system there does need to be enough statistical data available to define the membership functions and to set the sensors at the appropriate points. This takes more effort than setting up a fixed time controller.

References

Khalid, Marzuki (1996). "Intelligent Traffic Lights Control by Fuzzy Logic". In: *Malaysian Journal of Computer Science* 9.2, pp. 29–35. URL: `https://ejournal.um.edu.my/index.php/MJCS/article/view/2995`.

Warner, Josh et al. (2017). *JDWarner/scikit-fuzzy: Scikit-Fuzzy 0.3.1*. DOI: `10.5281/zenodo.1002946`. URL: `https://doi.org/10.5281/zenodo.1002946`.