CMPT 440 – Spring 2019: Traffic Light Project

Tom Mackinson
Due Date: 15/5/2019

1 Abstract

This project was an attempt to simulate the traffic lights at an intersection. By using a DFA I was able to simulate 2 sets of traffic lights, one going north/south, the other east/west. By simulating a standard intersection I was able to put the basics of a DFA to use in a real world example. This project involved creating a somewhat detailed state transition table, and then revising that document several times to add new types of lights and to fix some minor issues that were in early versions of the diagram. Overall this project proved an interesting example to look at with a DFA, and it provided a good way to put what I learned in this class to somewhat practical use.

2 Introduction

A traffic light is a very basic system, something that we see and interact with almost every day of our lives. Early in the semester when I was looking into example problems that could be simulated with a DFA I found that a traffic light would be a good candidate to work on. Traffic lights have a few very specific rules that they work on, there are only 3 possible colors, which can either be solid or flashing. Lights can never have the same color at the same time, unless they are both red. One additional rule I put in was that lights can not change colors if they are flashing already. To change the flashing color, they must go back to solid then cycle to the color requested.

Throughout the paper I will describe how my project works, and try to provide a description of how a real traffic light works, as this project is supposed to be simulating that, or at least providing a close approximation to how they work in the real world. Researching how traffic lights operate was interesting, and made me realize how it is possible to automate relatively simple systems with relatively simple solutions.

3 Detailed System Description

If this system were to be implemented in a real city or town, civil engineers would need to interact with it to set up the lights and ensure that they work. Once the system is setup, unless an error occurs somehow, they would never need to touch the system again. From then on the only people who would need to use it are drivers and pedestrians. Whenever someone approaches the light they would interact with it by just having it change colors.

The way this DFA works is by cycling through both sets of lights one at a time. Initially the north/south set is green, and the east/west set is red. The lights can either be set to flashing, or cycle to the next color, where north/south is yellow and east/west is red. If the lights cycle to yellow and red, they can again become flashing, or can cycle so they are both red. If both are red, the only possibility to make the east/west lights green, and the north/south lights red. These states are the reverse of the previous set, now the north/south lights stay red while the east/west lights cycle from green, to yellow, back to red. Just like before they can switch to flashing colors at any state, except if both are red. If the lights go to flashing, they cannot change colors unless they become solid first. These are a few simple rules that I put in placed based on how a real traffic light operates. Accepting states are any state where one light is green and one is red, or any state where both lights are flashing.

This system does not simulate when the light would change colors, as there is method for it to check how many cars are at a light, or which light should be green at a given time. That would probably be handled separately by a series of cameras, or sensors, which would detect when cars are waiting to go, and how many there are. That other system would then need to signal the light at an appropriate time to change colors. This would allow the light to function in a real environment, and would mean that the sensor system would be another user of my traffic light simulation then.

There are a few changes I would make to my project, if it were to implemented in the real world. Right now the project simply checks to see if the last state is an accepting state, and if it is it outputs that the project was correct. If the state is not an accepting state, then it says that the solution was not correct. This is a problem as there shouldn't be an accepting state for a traffic light, as it should never stop functioning in the real world. In order for this project to remain consistent with what else we have done in class, I decided to make any state where one light is green and the other is red, or any state where both lights are flashing, an accepting state. I would want to remove all accepting states, but still leave the error state as it is always possible that something can go wrong.

Another change I would make to my project is what is output when execution stops. As I said before, it outputs whether or not the state it ended at is an accepting state. If I remove all accepting states, then the output would need to change as well. I would instead output all of the values each light had during execution. This means that if one set of lights went from green, to yellow, to red, while the other set stayed red, the output would show the colors that each set had. This means there would be two columns, one to show the lights that changed from green, to yellow, to red, and another column for the lights that stayed red the whole time. This would be more useful for my project as it is now, and if this were to be adopted in the real world, it would allow a civil engineer to see what a light is displaying in real time. This would be much more useful than what it currently displays, however since the project currently works I would rather stick with what is currently implemented.

4 Requirements

The physical requirements of this system would be 4 traffic lights, one set for north/south, one set for east/west. In addition to this, there would need to be a set of sensors to detect when cars are waiting to go, as well as some sort of computer to interface with the traffic lights. This would allow a civil engineer or somebody to connect to the system, in case they need to for maintenance purposes. This would be easy enough to implement in a city or small town, and would just require a bit of planning before hand to figure out where it should be installed.

5 Literature Survey

As a traffic light is not a topic that has much research on it, there isn't much other research out there. Instead I will describe in more detail how a real traffic light works. According to HowStuffWorks.com, "When current first starts flowing in the coil, the coil wants to build up a magnetic field. While the field is building, the coil inhibits the flow of current. Once the field is built, then current can flow normally through the wire. When the switch gets opened, the magnetic field around the coil keeps current flowing in the coil until the field collapses... So... Let's say you take a coil of wire perhaps 5 feet in diameter, containing five or six loops of wire. You cut some grooves in a road and place the coil in the grooves. You attach an inductance meter to the coil and see what the inductance of the coil is. Now you park a car over the coil and check the inductance again. The inductance will be much larger because of the large steel object positioned in the loop's magnetic field." (How Stuff Works, Traffic Lights) This information would then be used by the light to see if the signal needs to change. Based on the number of cars and how long they have been waiting, the light can decide when to change colors, and so on.

One glaring issue with traffic lights today is their vulnerability to wireless attacks. While my project doesn't approach this weakness in anyway, it is present and generally simple to fix. Hackers from the university of Michigan were able to perform attacks on many traffic lights in their area, with proper permission first. They found that information passed between sensors, controllers, and the lights themselves were often unencrypted. Much of this hardware used default username and password settings, meaning gaining access and reprogramming them was very easy. Once access was gained to these devices, several attacks could occur, ranging from denial of service which sets all of the signals to red, to taking direct control of lights allowing someone to give themselves green signals at every intersection. (Ghena et al. "Green Lights Forever") While my system does not inherently provide security advantages, someone installing it for the first time would be informed that these weaknesses exist. Since hacking is viewed as a much greater threat today than it was over a decade ago, people installing this system would definetly take care to ensure that proper security measures, such as encryption and robust passwords, were in place.

6 User Manual

As described in the detailed system description section, this project works by cycling through the colors for each set of lights. It's also possible to set the lights to be flashing instead of solid, but it is not possible to change the color of flashing lights without making them solid again. For a better example of this, see the DFA State diagram below. This diagram doesn't list any error states as it is much easier to read without them drawn in. Assume that anything not explicitly stated in the diagram leads to an error.

In order to ensure proper use and prevent errors, I would ensure that the sensors communicate to some central system. This system could have either physical or virtual machines responsible for each intersection, and would be able to check the colors at each intersection to prevent any issues. These machines would also be responsible for calculating when the lights need to change colors, based on how many cars are waiting, and how long they have been waiting for. By separating each intersection out to different machines, it would make each intersection easy to keep track of, and would make it difficult for signals from different intersections to get confused. If all lights and signals in an area were monitored together it could lead to issues or confusion, if signals started changing based on the wrong sensors. It also means that adding new intersections into the system is even easier, as you would just need to spin up a new machine responsible for these lights and sensors. This reduces the likelihood of errors caused by lights changing color based on information from other sensors, which could cause problems if two lights were both set to green at the same time.

By assigning a machine or controller to each intersection, you could also ensure that each intersection is in a valid state according to the state diagram at any one time. This means that if both lights were to suddenly become green, the machine could see this as an error and override it, probably making both lights red until it can be reset or figure out what also happened. The controller could probably also be able to simulate changes before they are made live, in order to enforce the DFA diagram. If a change in colors would put the lights into an invalid state, then an error message could be sent and a human or some other system could intervene to ensure that everything continues operating smoothly. Preventing errors would be of the up-most importance since an error with traffic lights could put people's lives at stake.

7 Conclusion

This project was an attempt to use a DFA to simulate a traffic light. The goal was to put what we had learned in class into practical use, by simulating something that we all interact with in our daily lives. Traffic lights provide a valuable service to society, as they ensure that people are able to travel safely and smoothly. With only a few changes and some other systems in place, my simulation could almost be used as a real traffic light, as it provides the same function. Implementing this in the real world would require a few more systems, as well as some way to control the lights and ensure that they stay in valid states. With all of these things in effect, it could be possible for this system to work in the real world.

8 Bibliography

Contributors, HowStuffWorks.com. “How Does a Traffic Light Detect That a Car Has Pulled up and Is Waiting for the Light to Change?” HowStuffWorks, HowStuffWorks, 1 Apr. 2000, auto.howstuffworks.com/car-driving-safety/safety-regulatory-devices/how-does-a-traffic-light-detect-that-a-car-has-pulled-up-and-is-waiting-for-the-light-to-change.html

Ghena, Branden, et al. “Green Lights Forever: Analyzing the Security of Traffic Infrastructure.” USENIX, 28 July 2014, www.usenix.org/system/files/conference/woot14/woot14-ghena.pdf.

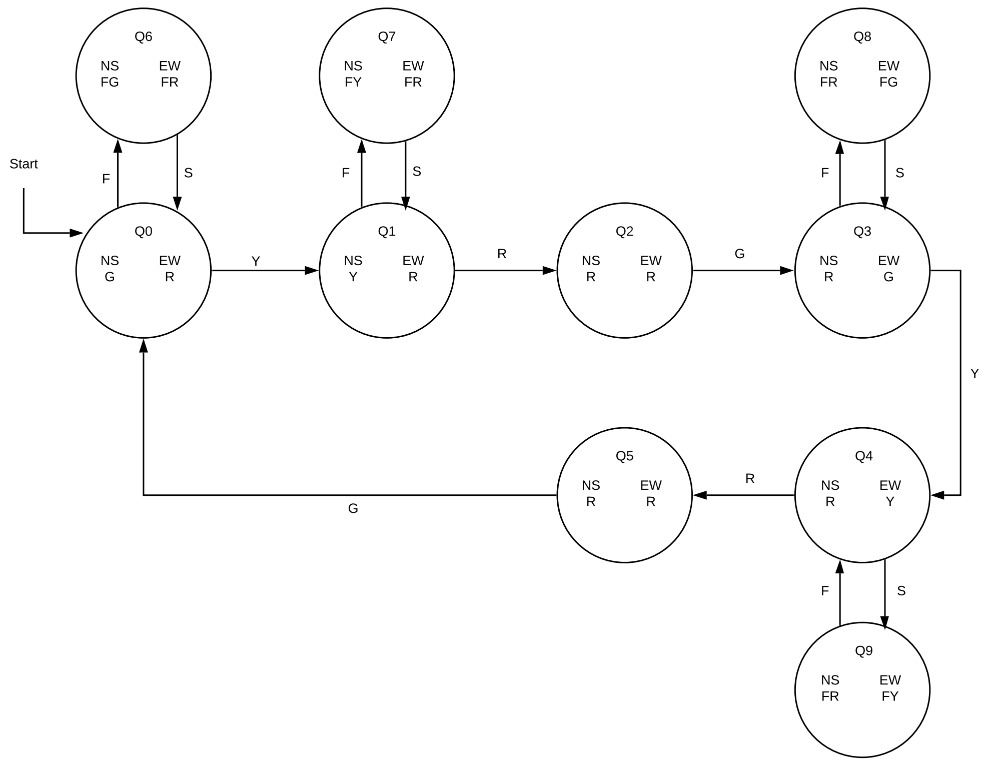

Figure 1: DFA State Diagram