University of Nottingham

School of Computer

G53IDS

**Evaluating the Efficiency of a Traffic Signal Control Agent Using Reinforcement Machine Learning**

April 23rd, 2018

Author:

4262267 Thomas Redhead (psytr1)

Supervised by:

Dr Brian Logan

Associate Professor

[*bsl@cs.nott.ac.uk*](mailto:bsl@cs.nott.ac.uk)

1. Abstract

Contents

# 1 Introduction

This project is an experiment into the application of a reinforcement machine learning algorithm on a custom-made simulator. It utilises a C++ framework provided by Jason Atkin of the University of Nottingham Computer Science department and relies on feedback of the proposed algorithm by Mike Hu of Nottingham Council Traffic Signal Management.

The aim of this project is to develop a model of a set of traffic lights and evaluate how implementing an algorithm using deep reinforcement learning to alter the traffic signal control will affect the average length of journey time for vehicles travelling through the set crossroads.

The key objectives are:

1. To investigate traffic light sequences around the world that apply to junctions of different shapes and sizes.
2. To investigate the feasibility of utilising an already existing traffic simulator or whether it would be more appropriate to implement self-built simulator. If an existing simulator does not include the desired libraries or procedures then a self-built one might be necessary.
3. To construct a grid model representing an abstract pattern of roads and junctions. This model will depict variations of junctions and road types to present the effect a machine learning algorithm could have on the system as a whole. The grid model will include agents representing vehicles and will assign distance values to vertices if suitable.
4. To design an algorithm using reinforcement learning that will dynamically adjust to the conditions of traffic or the type of junction present. This algorithm will take the existing state of a single junction sequence and make increments to improve the wait times at that junction. It will accomplish this by continuingly using trial and error and measuring the results against the previous state. Learning from past experiences the algorithm will know not to try a certain path if that has been proven to be inappropriate. I expect that although the wait times will decrease at this location they could drastically increase elsewhere due to build up.
5. Apply the algorithm to every junction, making alterations based on junction type, and combine the results to try and obtain and overall better average wait time. The algorithm will require fine tuning to find it’s optimal result. As this system is being tested in an abstract simulator it is not expected to have to deal with lots of categorical data. However, it may be necessary to implement an already existing algorithm to handle any conversion type errors that occur. In this instance the Catboost algorithm by Yandex |1| could be appropriate as it automatically deals with category related problems thus permitting more focus on the actual fine tuning. A potential issue exists where the algorithm improves Junction A which directly deteriorates Junction B. Junction B then tries to improve itself yet regresses Junction A and so a mathematical tug-of-war occurs for optimisation. A possible fix would be to calculate the overall effectiveness of the two junctions at any one point and quantify this in a variable. This variable will be influenced by the priority of the junctions, e.g. one is on busier road, and a potential number of other factors which will contribute to the overall value. This means that although the improvement of Junction A deteriorates the quality of Junction B, Junction B won’t resist as the global efficiency has increased.
6. Evaluate the feasibility of the algorithm being applied to the real world by contacting Nottingham City Council – Highways and Energy Infrastructure |2|. The method of this would be to enquire if they are aware of any such system already in place and evaluate the potential of the new system. It is expected that they will provide feedback which can be actioned and developed upon.
7. If approval is met, then proceed with publishing findings. However if the feedback is negative then try to improve the system to meet the requirements advised. For example if the suggested fault was in the lack of consideration in the algorithm regarding weather or road works then endeavour to implement these variables into the program.

# 2 Motivation

The average person in the UK spends up to 8 minutes a day stuck in traffic |3|. That’s 2 days in a year and 130 days in the mean life expectancy |4|. It may not sound insignificant but that’s over a third of a year doing nothing productive in your life. Traffic can be stressful especially when already in a tough working environment. It is therefore essential that this time is somehow cut down and the total time spent in traffic reduced.

At crossroads in urban environments a fixed time light sequence is commonly implemented as it helps to ease congestion and distribute fair green light time to each avenue approaching the crossroads |5|. This can be a practical solution during rush hour but in times when the frequency of road traffic is heavily reduced this can be a burden upon drivers who are required to halt even though there is little traffic around. A significant positive of the current system is that it is predictable. A local citizen who is setting off on a journey would know that the time they’ll spend at a traffic light is going to be the same assuming similar traffic frequency. This comfort may need to be sacrificed to improve the wait efficiency for everyone at a crossroads.

Traffic light systems using machine learning have been suggested previously in various iterations and most of them try to develop an algorithm that optimises a single junction rather than a grid model |6|. They don’t consider the wider impact of altering the sequence at one junction, as this may have a knock-on effect that disrupts traffic at many other surrounding junctions and thus results in a more inefficient system overall. The program that is desired would be one that can operate in a wider-spanning model and determine the most optimal algorithm for that whole area.

If a system can be created which dynamically adjusts to the intensity of traffic, then perhaps a lot of issues regarding effectiveness can be eliminated. It may introduce fresh problems like individual wait time increasing however, as it may benefit others, there is good reason to believe it will be an overall improvement.

# 3 Related Work

There already exist projects and research in~~to~~ improving a traffic light system at a localised junction. An example would be a paper by W.Genders *et al |*6| that implements a deep reinforcement learning agent for traffic signal control. They accomplish this by providing the agent access to three states, current state, an action state and a reward state. The agent observes the current situation and variables then analyses the various paths that could be taken. Upon making a decision, the agent receives a reward which indicates the effectiveness of the decision made. With each iteration the agent will be pursuing a higher reward and so will be constantly trying to improve its efficiency. The best thing about this proposal is that it suggests a relatively simple method in making the agent more efficient. It is, however, somewhat localised and can’t be easily adapted to other classifications of junction nor does it consider the impact to its neighbours.

In another paper, titled Adaptive Traffic Signal Control: Deep Reinforcement Learning Algorithm with Experience Replay and Target Network, the objective is to develop a complex and comprehensive agent that optimises a much larger junction with more external input. Authored by J.Gao *et al |*7|, it proposes that a deep reinforcement learning algorithm can automatically extract all features useful for traffic signal controls and optimise the policy based on them. The most important thing about this project is in its complexity, where the solution is developed with a 4 lane crossroads taken into consideration. This makes the mathematics and the computing behind the agent a lot more complicated yet yields greater results when applied correctly. Similarly to the previous example though is the fact that this agent only attempts to improve the congestion at a single junction, there is no consideration of the wider impact such a policy might have.

The importance of my project is that it intends to look at a wider scope than most other traffic light agents. Where other projects are designed to improve the efficiency of traffic flow at one junction, my agent will optimise the traffic flow at all surrounding junctions as well. This means that one junction may suffer from worse traffic congestion if it results in the improvement of several others. The product of this agent could be a more realistic algorithm developed that could be applied universally and make tangible change to the efficiency of urban infrastructure.

# 4 Description

The desired outcome of my project is to deliver an agent based learning algorithm to an abstract simulation and see significant reduction in traffic congestion times. The premise is to apply the algorithm to a simulated junction and then implement it across a custom-built road grid. This will hopefully adapt to the simulated environment and improve itself until an optimised solution can be found across all junctions. It may be the case that some junctions have an increase in congestion times, but this can be acceptable if it results in the improvement of others.

# 5 Methodology

## 5.1 Overview

The process of development for this project is in a three-part schedule. The foundation of the project is the simulation of the traffic light sequence. This model will allow me to construct junctions and mobile vehicles that interact with the environment and obey a predefined law of traffic signals. The second sequence of this project will be the deep reinforcement learning algorithm that will initially be applied to a single junction. The third section of the project will be inclusive of adjustment and optimisation. This involves the implementation of the algorithm across multiple junctions and the subsequent fine tuning of the algorithm to create the best results in the model. This stage will also require external feedback to assess the feasibility of the proposed system and then to action that response. Instead of working on each section in parallel I have elected to implement them in sequence, this will allow me to concentrate on each section individually and prevent disruptions to the work of other tasks if one section goes awry or requires further adjustment.

## 5.2 Simulation

The key part of developing the model is to recognise that it is not the focus point of the project. The purpose of my project is not to build a functional traffic simulator but to develop an algorithm that can use input data to modify itself to improve the results of the system it acts upon. Therefore my priority when researching and analysing different techniques of building a traffic light simulation was in ease of use and effectiveness. My options when it came to building a model were to utilise an already existing simulation program or to construct my own from scratch. Although existing simulators would save me the effort of having to design my own program they are overcomplicated. A popular traffic simulation program, Sumo, provides a lot of the basic functions that are required for effectively creating my algorithm. However, the program is also accompanied by a lot of unnecessary additions, such as pedestrian and cyclist inclusion. These extensions would be useful if more emphasis was placed on the model yet as the model is acceptable and preferable when simple, I believe that choosing this existing program would be inefficient for building the necessary model. Thus I investigated the feasibility of creating my own program using a language of my choice. There will be multiple entities that interact with each other suggesting the use of an object oriented programming language. This language will also have to be adept at handling mathematical inputs and processes so will need to have good processing power. I viewed Python as a potential candidate for my program yet I found that its IDLE framework was clumsy to use and I was inexperienced in its use. Any time spent learning the mechanisms of Python and its coding conventions would have been time that could be used more productively elsewhere. Conclusively I decided that Python would be an unsuitable programming language. As an alternative I turned to C++. Although the programming required would be arguably more complex than what Python demanded, I was already familiar with the language, having utilised it as part of a second year project to design a game. In order to give myself and my peers a boost in this language, we were provided a framework by the module convenor, Jason Atkin, as well as imports of various useful libraries like SDL – a crucial C++ library for implementing graphics. Initially I endeavoured to implement the libraries myself and build my own framework. I quickly learned that this was a meticulous and long process and wasted many hours attempting what had already been done by my module convenor. Therefore I abandoned my efforts and sought a quicker route to acquiring a framework for my simulator via Jason. Permission was acquired for the use of his framework through a personal enquiry and email correspondence. As a result of this the programming language that I wished to use for the construction of my model had been appropriately selected and provided a platform from which to build a model suitable for my ambitions.

## 5.3 Algorithm Development

Due to the structure of my development cycle I have yet to begin comprehensive research of how I’m going to develop an algorithm. Despite this, I have taken note of the existing methods of algorithm production. The paper by W.Genders (Genders & Razavi, 2016) details their idea for including a state system within their algorithm that the agent can act upon. By introducing a reward to the agent depending on which direction was taken, the algorithm encourages the agent to constantly seek improvements and become more efficient. I am intent on implementing my own algorithm with inspiration from this process as it provides a solid foundation that can be easily adapted and built upon. In spite of this basic research there is much more that demands investigation, such as how the agent will choose a path without extensive trial and error, thus saving time and providing consistent efficiency. There will also need to be a study into how the algorithm determines the efficiency of all the incoming lanes collectively at the junction. It is not yet clear how the algorithm will decide if the small congestion reduction at two of the lanes is worth a larger increase in congestion of just one.

## 5.4 Algorithm Application

Similarly to the algorithm development, the algorithm application is yet to be analysed and meticulously planned. Once again this is due to the priority of the sections of the project. The first section, the construction of the model, is incomplete and until it becomes clear it makes planning the third section abstract conjecture and subject to much change. Outlining any design is made more difficult by the fact that I have located no existing examples of this stage being attempted before. Therefore the results of my plan become impossible to predict and prevent me from analysing alternative ideas.

# 6 Design

## 6.1 Simulation

When considering how the algorithm would be best applied to a set of traffic lights I concluded that a completely abstract model would be the most appropriate. This is due to the lack of unpredictable behaviour in an abstract scenario. There are no external factors like weather or roadworks to consider as the model is purely hypothetical. Whilst it won’t display the effectiveness of the algorithm in a real world setting, the model can demonstrate the knock-on effect that a revised traffic signal control mechanism might have on surrounding junctions and traffic. It also eliminates unique features in the roadway such as roundabouts and five or more lane junctions. Whilst these occur in the real world they would interfere with the development of the algorithm and increase the complexity of the model beyond feasibility. Therefore I designed a system that contained two four-arm crossroads with two perpendicular branch roads.

**NW**

**SW**

**S**

**SE**

**NE**

**N**

**NW**

**NS**

**NE**

**NN**

**SW**

**SS**

**SE**

**SN**

*Figure 6.1.A*

The primary advantage of the design proposed is in its simplicity. The roads that remain consistently at right angles to each provide a good foundation on which to build mathematical equations for the physics of any vehicles in motion. The duplicity of the two types of junction allows for repetition of coding thus streamlining the development of the model.



*Figure 6.1.B*

### 6.1.1 Block Objects

When designing this simulator I initially looked at how to create the background and infrastructure of the system. These provide constant templates and fixed locations for the adaptive objects to interact with.

At the most basic level the background its self is set to a shade of medium grey to reflect buildings or other such impassable terrain that may be found in real life. This is constant shade and will not be changed during run time.

At the next level is the creation of the roads. These will be comprised of black rectangles placed in the pattern shown in figure 6.1.A, with one road running vertically lengthways in the simulator to represent an arterial road. The other two roads will be set at equidistant locations, bisecting the arterial road and thus acting as branches onto the artery.

Through each of these road blocks is a thinner white block that represents the road markings one would find to distinguish the boundaries of each lane. The colour is set to white as this is the most commonly used line colour used from the standard road markings listed in The Highway Code |8|

### 6.1.2 Traffic Light Objects

An essential part of the simulator is the system of traffic lights. Each one is responsible for ensuring that traffic flow is regulated yet allowed to pass through without too much delay.

Similarly to the block objects, the traffic light objects are initialised as rectangles at the end of each lane. This means that when a vehicle reaches the end of its respective lane it must past through the traffic light object in order to progress. The colour of the light is initially set based on an internally attributed green variable which determines the whether the light is green or not. If this green variable is changed throughout the run time of the simulator then the colour of the traffic light will also change to reflect that.

In order to create a basic sequence for the traffic light within the simulator there is a simple algorithm constructed. Utilising a variable that increments with each time step, the algorithm acquires the modulus of this time variable by dividing by 400 and returning the remainder to a modulus variable. This results in the modulus variable constantly incrementing until 399 then resetting back to 0, thus creating a light cycle for each traffic light. The value of 400 was chosen because it is an easily manipulated number and will provide ease of use in implementation. In reality the number could be any sufficiently high yet this is the one I chose. Next in the sequence, in a conditional statement, the modulus variable is compared with a threshold value. If the modulus variable is less than this threshold value then the green variable will be allocated to be one of true or false, depending on the location of the traffic light object. If the modulus variable is greater than this threshold variable then the green variable is set to the opposite. Thus the altered green variable will determine the colour of the traffic light at each timestep.

### 6.1.3 Vehicle Objects

The vehicle objects are the catalyst that instigate change within the simulator. When they interact with surrounding objects they can demonstrate patterns that emerge ibn the system as well as show how changes to the traffic light sequence might impact the flow of traffic.

The vehicle object is designed to be easily identifiable and simple to track. Each one is set at a bright orange colour that stands out from the drab background and can’t get mixed up with the traffic light objects. Their size is slightly smaller than the width of a lane so that it is clear what section of the simulator they occupy. In addition, they are in a circular shape so as to reduce difficulties with identification when pressed together with other objects.

The idea for the vehicles is that they enter the simulator at different rates. So, in the example of the South vehicle object type, 30 vehicles may be spawned yet they will enter the simulator in a staggered pattern in an attempt to mimic reality. This staggering is also generated randomly to further emulate the potential randomness of normal life. Once in the simulator each object will be locked into a lane section – the parameters of which are determined manually. This means that if the top left x and y coordinates are within a lane area then they follow the movement rules for that area. So for example if the vehicle is in the South Left lane then they are instructed to change their coordinates at rate of 2 pixels per time step, thus creating a movement mechanic in the system. If the vehicle then migrates to a different lane area the instructions it receives may be different and thus the direction the vehicle moves in might change.

Upon entering a junction, the vehicle will shift into a void space where no lane area has control. In this void space the vehicle will randomly determine the direction it will take. Each time the vehicle reaches this position is generates a direction variable which is set to a random number between 1 and 100. Then, using conditional statements, the vehicle can determine a direction depending in what margin the direction variable lies. If the direction variable is less than 45 for example, then the South vehicle will shift its y position by -1. This will move it into the middle left lane as the shift has placed it within that lane area. Thus, new commands are issued to the vehicle and the movement cycle will commence again.

Upon making contact with a traffic light object the vehicle will assess what colour the light is set to be. If the traffic light colour is determined to be red, then the vehicle will enact the procedure required to halt its movement. Only once it detects the traffic light changing to green will it continue along its path.

### 6.1.4 Wait Times

To keep track of the changes made to the system and to act as a metric by which the effectiveness of the system is measured the average wait times of all vehicles will be recorded.

First of all, a universal variable will be added to each vehicle which increments with each time step. If the vehicle leaves the visible range of the simulator then this variable is reset to zero to reflect the vehicle having reached its destination. This variable is then extracted by the engine of the simulator and added to another variable that counts the total wait time of all vehicles of that specific vehicle type. This sum is then divided by the number of vehicles in the system of that type to produce an average waiting time which is then output in the display on the simulator screen. The output is done to provide a visual representation of the effects that the traffic light sequences might have on their respective vehicles. This process is repeated for every vehicle type.

Further to this, an average total wait for the entire system is generated using the same initial method yet getting the sum of the respective wait times for all vehicle types which is then divided by the total quantity of vehicles in the system. The value produced by this is then output into the simulator to be used as the metric of success of any sequence algorithms applied to the traffic lights. As the purpose of the project is to evaluate the potential improvement a reinforcement machine learning algorithm might have on the entire system, this output value is the most important of all.

Finally, to add a marker at which data can be consistently taken, a simple timer will be implemented that counts, in seconds, the time passed since the running of the simulator began.

## 6.2 Algorithm

Due to the impact that a machine learning algorithm would need to have on the system, I decided that it would be placed within the engine of the simulator. This is beneficial because it provides easier access to many of the fundamental and core members and methods that might be in the code. It is also the most efficient place for it to be written as it can directly impact all of the traffic light objects simultaneously.

### 6.2.1 Proportion Variable

As a preliminary to the algorithm the instructions that are given to the traffic light objects needed to be altered slightly. Instead of just setting a constant value that acts as a threshold for the modulus of the time, a dynamic variable needed to be implemented as this is the key to altering how the sequence of the light signals can change. Each traffic light objects extracts a proportion value from the engine depending at what junction they are located. This means that the proportion value can be altered and tinkered with from inside the engine and the changes made will pass on through automatically to the traffic light object.

### 6.2.2 Red Counting

The Red Counting function acts as an auxiliary part of the machine learning algorithm. This is a function that produces values required for the main algorithm to make suitable improvements.

It accomplishes this by first introducing red variables to every vehicle object for each potential traffic light they may encounter. Within the vehicle collision method these variables are set to be true respective to what traffic like they are stuck on red at. So if a South vehicle was waiting at the South South traffic light then the red variable for South South will be true. Once the light turns green and the vehicle is released the respective red variable is set to false.

Back in the Red Counting function within the simulator engine a red counter variable is initialised for each traffic light. The function then loops through every vehicle object in the system and at each one determines if it is in contact with the specified traffic light. For every one that is, the red counting variable for that traffic light is incremented by one so by the end of the loop the total number of vehicles stopped at the traffic light whilst it is on red is known.

### 6.2.3 Machine Learning Algorithm

The machine learning algorithm is applied to only one junction at a time and all changes made will only impact that specific junction. The result of this algorithm is hopefully a more optimised proportion value that can be provided to the traffic light objects at both this junction. This means that every time the machine learning algorithm runs it must return an integer proportion variable.

The premise of this algorithm is that a reward is provided to the algorithm if the results it produces move it in the direction of a more optimised solution. By comparing the total average wait time from the last timestep to the one produced most recently by the learning algorithm it is possible to detect whether there has been improvement, worsening or no change. If the total average wait time has been improved, then a reward is issued. The reward comes in the form of not changing the current proportion variable thus continually moving the ideal position of the modulus threshold in the same direction. The basic formula of the learning algorithm is to say that if the total average wait time is moving in the direction of a more optimised solution then continue with the same proportion value.

If a reward is not issued to the algorithm, then the algorithm is punished. It must attempt to generate a new proportion value until the average wait time begins to improve. It accomplishes this by using the variables for each traffic light produced by the red counting function. The red counting variables for both the traffic lights in the x axis are added together to produce a total weight pushing upon the lights in the x axis. The same is done with the traffic lights in the y axis. The total weights of the axes are compared with each other and the procedure following depends on the result.

If the combined weight on the y axis is greater than that on the x axis, then the difference between the two is calculated. A multiplier variable of 1 is then adjusted by subtracting the difference in weights multiplied by 0.01. The 0.01 value is decimal selected because it allows changes made to the proportion to be significant enough to be noticed but not too large that an optimal position may be overshot. The current proportion value is then multiplied by the multiplier variable thus creating a new proportion that will be slightly less than the one from the previous time step. Once passed to the traffic light objects this new proportion value will be used to then assign more green, light time to vehicles in the y plane and reduce the green light time to vehicles in the x plane.

If the combined weight on the x axis is greater than that on the y axis, then the difference between the two is calculated. A multiplier variable of 1 is then adjusted by adding the difference in weights multiplied by 0.01. The current proportion value is then multiplied by the multiplier variable thus creating a new proportion that will be slightly more than the one from the previous timestep. Once passed to the traffic light objects this new proportion value will be used to then assign more green light time to vehicles in the x plane and reduce the green light time to vehicles in the y plane.

# 7 Implementation

## 7.1 Simulation

### 7.1.1 Basic Objects

### 7.1.2 Traffic Light Objects

### 7.1.3 Vehicle Objects

### 7.1.4 Wait Times

## 7.2 Algorithm

### 7.2.1 Proportion Variable

### 7.2.2 Red Code

### 7.2.3 Learning Algorithm

# 9 Acknowledgements

The primary acknowledgment that needs to be made in this report is to Jason Atkin. My second year C++ module convenor permitted me to utilise his code framework to build upon and develop my model. Without this the programming task would have been significantly more difficult and would have wasted time that could have been used more productively in my important parts of the project.

# 10 Bibliography

|1| Yandex (2017). *Catboost.* Retrieved December 7, 2017, from <https://tech.yandex.com/catboost/>

|2| Members Directory (2017). *Nottingham City Council – Highways and Energy Infrastructure.* Retrieved October 20, 2017, from https://www.recc.org.uk/scheme/members/00063268-nottingham-city-council-highways-and-energy-infrastructure

|3| Confused.com (2017). *Drivers spend two days a year waiting at red lights.* Retrieved October 20, 2017, from https://www.confused.com/on-the-road/driving-law/drivers-spend-two-days-a-year-at-traffic-lights

|4| The World Bank (2017). *Life expectancy at birth, total (years).* Retrieved October 20, 2017, from https://data.worldbank.org/indicator/SP.DYN.LE00.IN

|5| Traffic Signal Design (2017). *How do Traffic Signals Work?.* Retrieved October 20, 2017, from http://www.traffic-signal-design.com/how\_do\_traffic\_signals\_work.htm

|6| Gao, J., Shen, Y., Liu, J., Ito, M., & Shiratori, N. (2017). *Adaptive Traffic Signal Control: Deep Reinforcement Learning Algorithm with Experience Replay and Target Network.* Nara: Graduate School of Information Science, Nara Institute of Science and Technology.

|7| Genders, W., & Razavi, S. (2016). *Using a Deep Reinforcement Learning Agent for Traffic Signal Control.* Ontario: Department of Civil Engineering, McMaster University.

|8| GOV.UK (2018). *The Highway Code: Road Markings.* Retrieved April 22, 2018, from https://www.gov.uk/guidance/the-highway-code/road-markings