Simulating a Traffic Intersection in Python

John Knight

The University of Texas Rio Grande Valley

STAT-4345 Introduction to Simulation

Prof. Tamer Oraby

29 April, 2023

Abstract

This study used a model written in Python to simulate a four-way traffic intersection with each road having a straight lane and a left-turn lane, at varying levels of traffic. Several probability distributions were utilized to reflect the unpredictable nature of human drivers, including a Poisson process by which cars appeared at each road on the intersection, a normal distribution of random variables for attributes such as each driver's preferred acceleration rate and top speed, and an exponential distribution of drivers' reaction time when the traffic light changed from red to green. The study found that traffic flow has a nonlinear relationship with traffic volume, reaching a saturation point beyond which more cars barely increase the rate at which cars traverse the junction. At low levels of traffic, shorter green light cycles were found to be optimal, while at higher levels of traffic, longer green light cycles with shorter green arrow cycles were preferred.

Introduction

Most drivers around the world are familiar with the frustration of being stuck in traffic. Not only is traffic a nuisance, it is also estimated to cost the U.S. economy $179 billion per year (Clark & Smith, 2019). Traffic is often congested around an intersection controlled by traffic lights, and all drivers can surely recall times when the settings of the light appear to have been suboptimal. For example, perhaps they have been on a very quiet road, and stuck at a red light with no other cars to be seen. Or conversely, they might be at a very busy intersection and the light changes too quickly from red to green and back to red, such that they do not get through on the first cycle.

So the question arises, what are the optimal settings for a traffic light to cycle between its different states? De Schutter and De Moor (1998) constructed a model to optimize traffic light switching schemes using a simple green, amber, red sequence. However, in urban areas of the United States, many intersections include an additional state of “green arrow” that allows cars to turn left. This project used a simulation of such an intersection to answer the following research questions:

1. How does the mean time taken to traverse an intersection change as traffic volume increases?

2. What are the optimal settings of green light and green arrow that minimize the time taken by cars to traverse the intersection?

3. How does this optimal setting change at low, medium, and high levels of traffic?

Methods

A simulation was written in Python to replicate the behavior of cars at an intersection similar to that shown in Figure 1. This was based on roads in the United States, where vehicles drive on the right. Cars approaching the intersection were able to turn left or right, or continue straight (this intention was randomly assigned to each car when it was generated). Cars turning left used the designated turning lane, while cars going straight or right used the other lane. An additional stipulation was that cars turning left could do so when there was a green arrow, or a green light with a sufficient gap in the oncoming traffic. Similarly, cars turning right could do so on a green light, or on a red light with a sufficient gap in traffic coming from the left or from the oncoming turning lane.

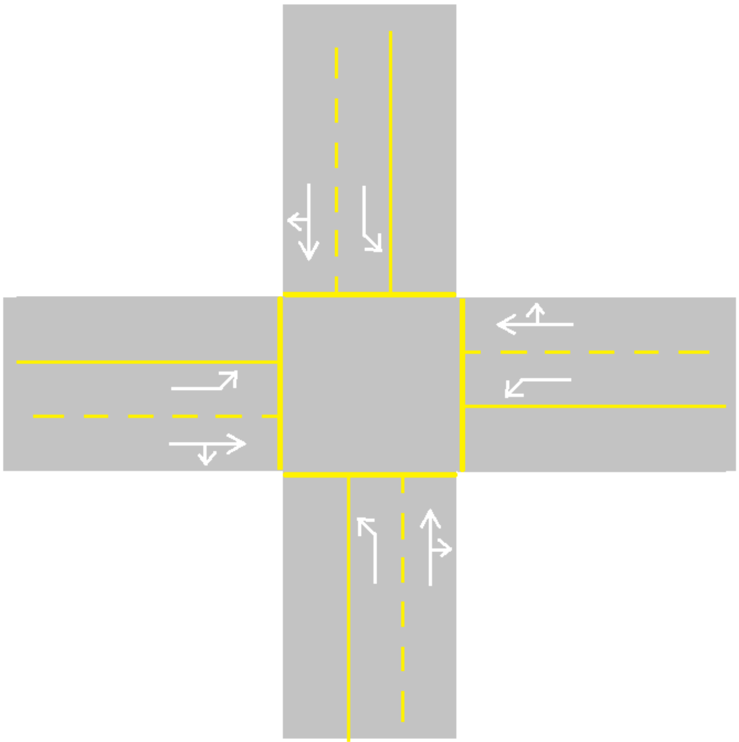

Figure 1. Four-way intersection with designated left-turn lanes.

Each of the four roadways was designated by the direction the traffic is heading: North, South, East, or West. Cars appeared as a Poisson process with rate λ at a point 100 meters short of the traffic light on each of the four roads. To reflect a real-life traffic system, cars did not all behave identically. Upon generation, each car had its own preferred acceleration rate, top speed, and safe distance from the car in front, all of which were normally distributed random variables with a designated mean and standard deviation. Table 1 shows the default values for these variables as well as other variables that could be altered in any future development of the model.

Table 1. Default values and descriptions of the global variables in the model.

Variable	Default Value	Description
default_start	100	The starting distance from the traffic light of each newly-generated car (in meters).
L	10	L=lambda. The rate (per minute) of the Poisson process by which cars appear on each of the four roads.
acc_mean	1.3	Mean acceleration rate of cars (in m/s^2).
acc_sd	0.1	Standard deviation of acceleration rate.
ts_mean	20	Mean top speed of cars (in m/s).
ts_sd	1.33	Standard deviation of top speed.
safe_d_mean	1.5	Minimum safe distance from car in front (in number of seconds it would take to hit car in front at current speed).
safe_d_sd	0.2	Standard deviation of safe distance.
chunk	0.1	Number of seconds that elapse in each cycle of the simulation.

car_length	6.5	Length of cars (in meters), interpreted as 4.5m for the car plus an additional 2m minimum gap between cars.
green_t	30	The number of seconds traffic lights are on green.
green_arrow_t	10	The number of seconds traffic lights are on the green turning arrow.
change_lag	2	The number of seconds between changes, when all lights are on red.
react	0.5	The mean reaction time (in seconds) for drivers starting to move from a stationary position.
react1	1	The mean reaction time for the first driver at the intersection.
turn_props	[0.1,0.8,0.1]	The proportion of drivers turning left, going straight, and turning right, respectively.
safe_turning_gap	3	The number of seconds' gap between oncoming cars required for a car to turn left on a green light, or right on a red light.

Each iteration of the simulation proceeded for the amount of time designated by the variable **chunk** (default 0.1 seconds). The program cycled through each of the four roads, first generating cars, then moving cars, and finally changing the traffic light state if appropriate. As cars were generated they were automatically set to their preferred top speed, unless they were too close to the car in front in which case they were reduced to an appropriate speed and moved farther back if necessary (in the case of a long traffic jam there was no limit to how far a car could theoretically be moved back).

The movement of a car depended on its current speed, the location and speed of the car in front, the current state of the traffic light, and the car's intention to either go straight or turn left or right. Firstly, any car that was stationary but now needed to move (for instance when a light turned from red to green) had its own unique reaction time. This was generated as an exponentially distributed random variable. To simulate real life, the first car at the traffic light had a higher mean reaction time than all other cars (since the driver was more likely to be caught looking at his or her phone, perhaps). Regardless of their desired movement, all cars had to maintain a safe distance from the car in front, and this limited their speed and acceleration. Cars that were restricted by a red light decelerated at an appropriate rate (using the 3rd equation of motion: $v^2 = u^2 + 2as$) to reach a speed of zero at the light. To mimic the role of an amber warning light on a real-life traffic signal, drivers were deemed to be able to predict the timing of a red light change, and so they would begin to decelerate on a green light if it was calculated that they would not be able to reach the traffic light before it turned red.

A car that was turning left could proceed through the light if the current state was "green arrow", or if the current state was green and it was calculated that there was a gap equivalent to the variable **safe_turning_gap** in the oncoming lane (for example, the oncoming lane for the South traffic turning left would be the North traffic going straight). Similarly, cars wishing to turn right on a red light would need the same gap from cars coming from their left (for example, South cars turning right would need a gap in the East cars going straight) or from cars turning left in the oncoming road (for South cars turning right, these would be North cars turning left).

Once all cars had been moved, the number of seconds remaining until a light change was checked. When this reached zero, the state of the light changed to the next phase of the cycle. As in real life, this included a designated lag period where all lights were on red, without which there would be a high possibility of an accident if a car even slightly overran a red light.

Cars were counted as they passed through the intersection and the system was simulated for a total of 65 minutes for each set of parameters. This included a ‘burn-in’ period of 5 minutes to allow the distribution of traffic to settle into its natural state, given the intersection always began with no cars. For each simulation, parameters calculated were the mean amount of time cars took to traverse the intersection, as well as the number of cars that passed each light per minute. Additionally, the mean squared time of each car was recorded, since this might account for an uneven distribution of utility in how long it takes to get through a traffic light – it may not be desirable to simply minimize the mean time, if it meant some drivers were stuck at the light for an inordinately long period.

The first simulations run were with the default light settings of 30 seconds green and 10 seconds green arrow, for all lambda values from 1 to 20. Thereafter, simulations were run at low ($\lambda = 5$), medium ($\lambda = 10$), and high ($\lambda = 15$) levels of traffic, each with varying combinations of green light (15, 30, 45, and 60) and green arrow (5, 10, and 15).

Results

The mean time to traverse the intersection (Figure 2) rose slowly from $\lambda = 1$ to 23.11 seconds at $\lambda = 10$, and then rose much more sharply as traffic increased. Also, the distribution of cars per minute passing each light (Figure 3) increased in a linear fashion up to $\lambda = 10$ and then almost flattened at higher values.

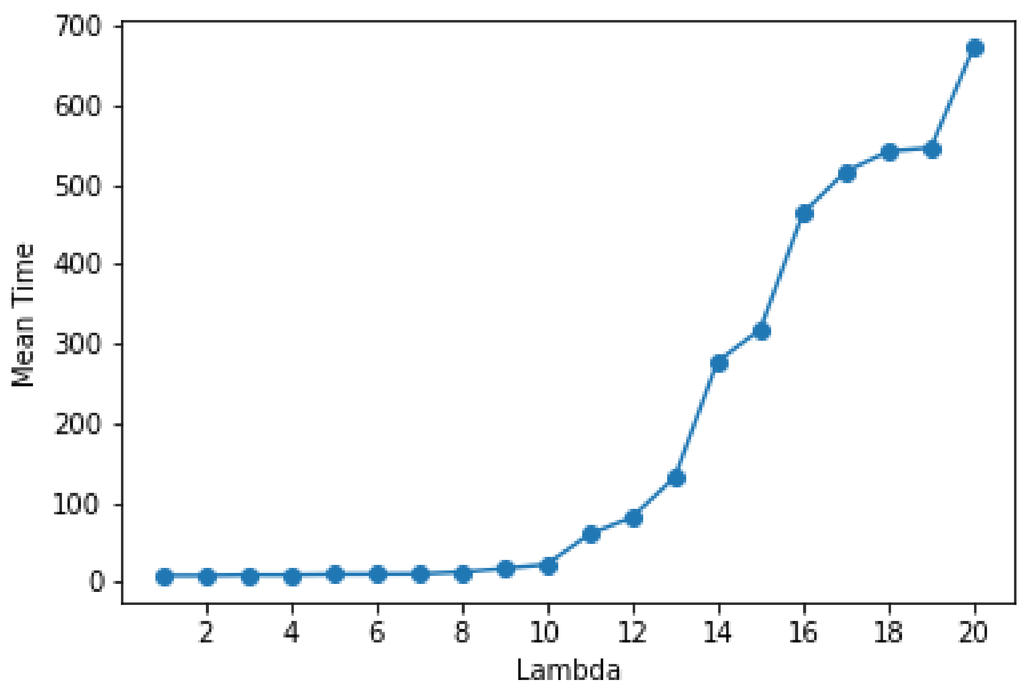

Figure 2. Mean time (in seconds) for a car to traverse intersection for different values of λ from 1 to 20, using default traffic light settings of 30 seconds green, 10 seconds green arrow.

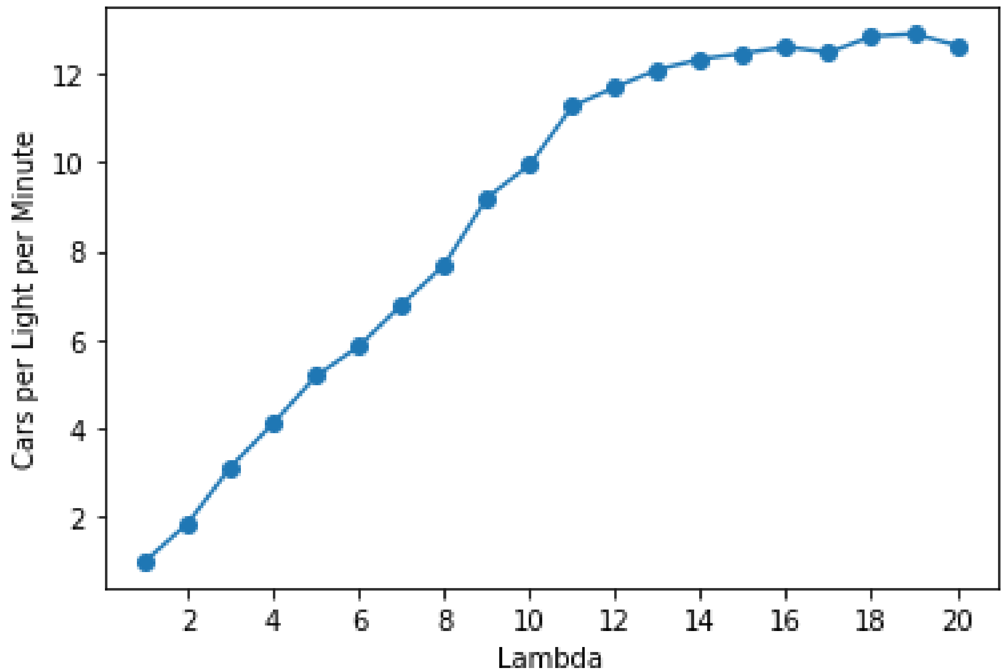

Figure 3. Number of cars that traversed each light per minute for different values of λ from 1 to 20, using default traffic light settings of 30 seconds green, 10 seconds green arrow.

At a low level of traffic (Tables 2 and 3) the optimum setting was 15 seconds green, 15 seconds green arrow. At a medium level of traffic (Tables 4 and 5) the optimum setting was 45 seconds green, 10 seconds green arrow. At a high level of traffic the optimum setting for mean time (Table 6) was 30 seconds green, 5 seconds green arrow. However, when measuring by mean time squared (Table 7) it was 60 seconds green, 5 seconds green arrow.

Table 2. Mean time (in seconds) for car to traverse intersection with $\lambda = 5$.

Green / Green Arrow	5	10	15
15	8.66	8.89	8.5
30	9.97	9.47	9.5
45	11.32	9.77	9.7
60	11.79	10.38	10.29

Table 3. Mean squared time (in seconds) for car to traverse intersection with $\lambda = 5$.

Green / Green Arrow	5	10	15
15	91.46	98.17	88.62
30	141.24	122.12	126.27
45	210.13	149.77	145.81
60	275.21	201.70	189.14

Table 4. Mean time (in seconds) for car to traverse intersection with $\lambda = 10$.

Green / Green Arrow	5	10	15
15	23.17	18.81	21.85
30	19.74	23.11	19.89
45	22.04	18.56	19.43
60	22.99	24.31	19.78

Table 5. Mean squared time (in seconds) for car to traverse intersection with $\lambda = 10$.

Green / Green Arrow	5	10	15
15	984.37	624.85	886.82
30	606.28	868.23	636.80
45	795.77	531.34	605.67
60	844.18	925.11	663.03

Table 6. Mean time (in seconds) for car to traverse intersection with $\lambda = 15$.

Green / Green Arrow	5	10	15
15	313.97	346.30	315.61
30	293.04	340.96	309.11
45	347.79	326.14	377.65
60	300.80	306.61	329.89

Table 7. Mean squared time (in seconds) for car to traverse intersection with $\lambda = 15$.

Green / Green Arrow	5	10	15
15	149018	172902	158069
30	132745	179791	154380
45	175017	150585	211370
60	131506	140836	175249

Discussion

From Figures 2 and 3 it is apparent that the relationship between volume of traffic and the time it takes to traverse the intersection is not linear. Rather, it appears that there is a saturation point beyond which traffic quickly grinds to a halt. In this simulation the saturation point is around $\lambda = 10$ (equivalent to a mean of 10 cars per road per minute).

At low levels of traffic, the results suggest that a short period of time is optimal for the green light. This is perhaps intuitive, as it prevents cars from being sat at a red light while there is no traffic crossing the perpendicular road. However, it does not follow that the same is true for the green arrow, since the lowest mean time was recorded with the green arrow showing for 15 seconds. This could be due to the fact that cars take time to decelerate, react, and accelerate when a light changes, and so more frequent changes add more of this wasted time.

At a medium level of traffic, a more balanced setting of 45 seconds green and 10 seconds green arrow was optimal. It is interesting that for the highest level of traffic, whether Table 6 or Table 7 is used, the optimum setting had the lowest number of seconds for the green arrow, meaning it was more valuable to devote time to the traffic moving straight or turning right than just the cars turning left. Although, drivers of the cars turning left would no doubt disagree.

Conclusion

This simple simulation model provides insights into the relationship between traffic levels and optimum traffic light settings, and offers a platform for more complex analysis of traffic flow at an intersection. There are many areas in which this study could be further developed. Variance reduction techniques like antithetic variables could possibly be used as well as longer simulations, to reduce variance. An interesting twist would be to use different values of λ on each road, and see how the optimal settings align with those. Of course, there are also many different types of intersection with varying numbers of lanes and turning lanes, with other rules such as no turning on a red light at some junctions. Indeed, there is an almost limitless number of combinations one could think of, and it underlines the difficulty and importance of the role of city planners. Recent advances include smart traffic lights that use sensors, cameras, and even artificial intelligence to make real-time decisions that improve traffic efficiency. It is an example of statistics being used to improve the lives of all citizens, whether they realize it or not.

References

Clark, B., & Smith, J. (2019, December 23). How traffic jams cost the US economy billions of dollars a year. *CNBC*. https://www.cnbc.com/2019/12/24/traffic-jams-how-they-form-and-end-up-costing-the-us-economy-billions.html.

De Schutter, B., & De Moor, B. (1998). Optimal traffic light control for a single intersection. *European Journal of Control*, *4*(3), 260-276.

Appendix

Link to Python code: https://github.com/capybara2/traffic_project/blob/main/traffic2.py