Learning Traffic Light Control Policies

Avram Robinson-Mosher
Christopher Egner

1. Introduction

With the growth of modern cities and the reliance of many of their populations on personal automobiles for the primary mode of transport, finding improved means to control the flux of vehicles has grown increasingly important. There are substantial benefits to be derived from improved traffic flow. For many commuters, reclaiming part of their day would enhance their quality of life and less congestion would engender fewer accidents, saving lives. Furthermore, time spent traveling to and from work is not time spent doing work. In fact, most people are essentially constrained to perform only the task of driving as they commute. Goods must be transported and service providers must travel to their clients. Clearly, traffic delays impinge on productivity and economic efficiency. There is also the concern of pollution as cars are generally less efficient in stop-and-go than in smooth flowing traffic. Longer commutes also mean longer running times and entail more greenhouse gases.

By improving the policies that control traffic lights, traffic flow can be improved to mitigate these problems and it can be done for considerably less cost than other infrastructural improvements such as increasing the capacity and number of roadways or adding public transit systems.

Currently, there are efforts to create reasonable control policies, but most are ad hoc and constitute manual tuning. The result is that drivers notice policies that are clearly suboptimal. We will show that, by applying machine learning techniques, we hope to derive policies that are, at least, locally optimal and are, in the general case, better than manual tuning. These policies would yield a net improvement in the efficiency of traffic systems while maintaining fairness. Since engineers are no longer hand-tuning policies, automated policy search could also yield a reduction in cost of system design.

2. Methods

We attempted to learn traffic light control policies for the same road map (see Figure 3.1) using three different approaches. The first approach is traffic balancing. Essentially, it attempts to balance the green time of a direction at the light with the relative amount of traffic arriving in that direction at the intersection. The second approach uses simulated annealing and a cost heuristic to derive the traffic light control policies in a reinforcement learning context. The final approach is to recast the problem as a Markov Decision Process (MDP) and use policy iteration to find policies. All three methods rely on a simulator to generate the features of a function to determine the quality of a solution.

2.1. Simulation

Since there is no readily apparent closed-form function that predicts traffic flow on an arbitrary map, we resort to simulation. By simulating traffic flow and extracting salient features, we derive a basis on which to compare policies.

2.1.1. Modeled Phenomena

The focus of our work is to apply and analyse the success of various machine learning techniques for learning traffic light control polices. The simulation of traffic flow given a map, speed limits, vehicle features, driver patterns, *et cetera*, is incidental to our work and hence deriving a realistic and validated simulation is simply beyond our scope. To mitigate this disadvantage, we designed our learning tools so that any simulation capable to producing the metrics we require can be attached and thus we are not strictly dependent on one particular simulator.

We use a simulator largely developed by Samantha Chui, TJ Hsiang, and Jennifer Shen at Stanford. This simulation is correct to a first order approximation. It models roads and intersections, controlled by traffic lights. Roads are single lanes with speed limits. Cars accelerate to speed limits, never exceeding them, and decelerate to avoid collisions and to comply with traffic lights. Drivers choose the shortest distance between their starting location and their destination. Accidents, merging, multilane roads, turn lights, varying speed limits, and driver aggressiveness are not modeled.

2.1.2. Metrics

We have two intuitive criteria for determining the quality of a policy. The first is that a

policy for individual traffic lights should, in aggregate, maximise the number of cars that are able to travel from their point of departure to their destination within the course of the simulation. This is the efficiency criterion. The second is that the policy should be fair. Clearly, optimising for flow alone could cause starvation or near-starvation as minor streets intersect major thoroughfares. White it is entirely appropriate that major roadways should take precedence, extreme wait times should be penalised. This is the fairness criterion.

These two criteria motivate five simulation metric to measure the quality of an ensemble of individual light policies, given a road map:

- time required to travel key routes
- average speed of all cars
- throughput at each intersection
- disparity in wait-time distributions for different directions are each intersection, and
- average time vehicles spent accelerating and decelerating while under the control of a light.

The first two metrics are global in the sense that their values are for a simulation run and not for individual lights. They primarily address the efficiency criterion. The last three are local, meaning that the policy for each light is evaluated individually against them. The first local metric again addresses primarily the efficiency criterion while the second local metric addresses the fairness criterion. The final metric specifically targets stop-and-go oscillation which increases pollution and hinders the efficient, smooth flow of traffic. It also discourages policies which change the light very frequently, which may be unsafe.

2.2. Throughput Balancing

Throughput balancing is an algorithm for locally maximising the expectation that a vehicle arriving at a light will not have to stop. It assumes fixed-length periods for traffic lights and adjusts the fractional amount of time for each direction to be the fractional amount of traffic arriving for each direction. In other words, if 65% of the traffic flows north-south through and intersection, then the light should be green for north-south 65% of the time. This algorithm greedily attempts to increase traffic flow at each individual light in the hopes that this will maximise the flow for all lights. If we ignore the effects of acceleration and yellow lights and let x be both the fraction of traffic arriving at an intersection for either of the two roads and also the fractional amount of time for which the light is green for this road, we have the expectation that any car arriving at the intersection will be able to proceed $g(x)$ as

$$g(x) = x^2 + (1-x)^2$$

$$\frac{dg(x)}{dx} = 4x - 2$$

$$\arg\min_x g(x) = \frac{1}{2}, \min_x g(x) = \frac{1}{2}.$$

This implies that, if we ignore situations where neither direction at a light is green, then the expectation that any car arriving in either direction will be able to continue through the light is at least 50%. Since , in the real world, the period it takes for a light to completely cycle is large compared to time spent switching directions (i.e. in the yellow-red and red-red phases), the above bound reasonably approximates real world conditions.

2.3. Simulated Annealing

Simulated annealing is a non-deterministic search technique. Parameters are altered and the new solution is evaluated. If the alteration is an improvement, it is accepted with a probability given by the sigmoid function

$$p(x) = \frac{1}{1 + e^{tx}}$$

where x is the quality of the solution according to some fitness function and t is the "temperature" of the annealing process. Temperature decreases linearly during rounds. As the

process cools, changes leading to improved fitness values are more and more likely to be accepted. For our fitness function, we used each metric individually. A probability that the change in that metric should be accepted is derived. A random number is then generated from a uniform distribution and a vote is cast by that metric for acceptance of the change. A simple majority vote decides whether or not the change will be accepted.

Under our model the parameters for each light are the length of time it spends green for each direction. Policies were randomly initialised and the annealing process continued for approximately 1100 rounds. Annealing was then repeated several times in an effort to mitigate the effects of random seeds and local optima.

2.4. Markov Decision Process

The Markov Decision Process, or MDP, formalism is attractive for traffic light control for two reasons. First, the Markov assumption, that the next state of traffic only depends on the current state, is reasonable: vehicles having already left the intersection generally have little effect on local conditions. Second, there are algorithms for determining locally optimal policies once the problem is recast in the formalism.

First intuition may lead one to model the entire traffic system globally with a state of the system being the state of each traffic light (two variables, one for each direction, each taking on three values, green, yellow, and red). However, since the number of states is exponential in the number of variables, this quickly becomes intractable. For example, a simple grid pattern with four roads in each direction has 16 traffic lights. This yields $3^{16 \cdot 2} \approx 10^{15}$ states. More concretely, many large east coast American cities have blocks between $1/16$ and $1/8$ of a mile long. A 10-by-10 grid, representing about one square mile, would have $3^{100 \cdot 2} \approx 10^{95}$ states, orders of magnitude more than the number of atoms in the universe. More complex traffic systems, such as those with turn lights, explode to intractability even faster.

Therefore, we reduce the problem from a global scope to a local scope. We treat each traffic light as its own MDP. Its state is defined as its configuration (assignment of green, yellow, red to its two directions) and the configurations of the four adjacent lights. We also introduce a temporal variable to the state to represent the time spent in the current configuration of the five lights. This model is linear in the number in traffic lights and so the problem becomes tractable. The actions available in any state are to transition the light to green in a certain direction if it is not already. Since the adjacent lights may change their configurations external to the decision of the central light's policy, state transitions become probabilistic under this model.

There is an intuition to back this local model. A traffic light is concerned with traffic that comes from four other sources. There is generally a correlation between that light's configuration and the amount of traffic it sends, i.e. that more traffic arrives when it is green in the direction pointing toward the central light. This traffic takes a certain amount of time to propagate and hence a model that observes the four adjacent lights and tracks the time for which they have been in their current configuration is reasonable.

Once we have the model in place, we then use the policy iteration algorithm determine a locally optimal policy for each light. The policy for the system is the combined policy for all the lights.

3. Results and Conclusions

Applied the three different models of the problem to a traffic simulator to discover if we could improve policies over time where improvement is measured by the metrics outlined in section 2.1.2. Each model has its flaws as does the simulator.

Our algorithms competed on the same map (Figure 3.1). This map was chosen because it based a common grid pattern with varying distances between intersections. Traffic loads were increased for certain streets. If our algorithms were performed well, we expected to see policies for lights where heavily traveled streets intersected lightly traveled ones to favor the heavily traveled ones. We also expected to see improvements in our metrics since the system should ameliorate as

each light learns and improves the way it processes its traffic.

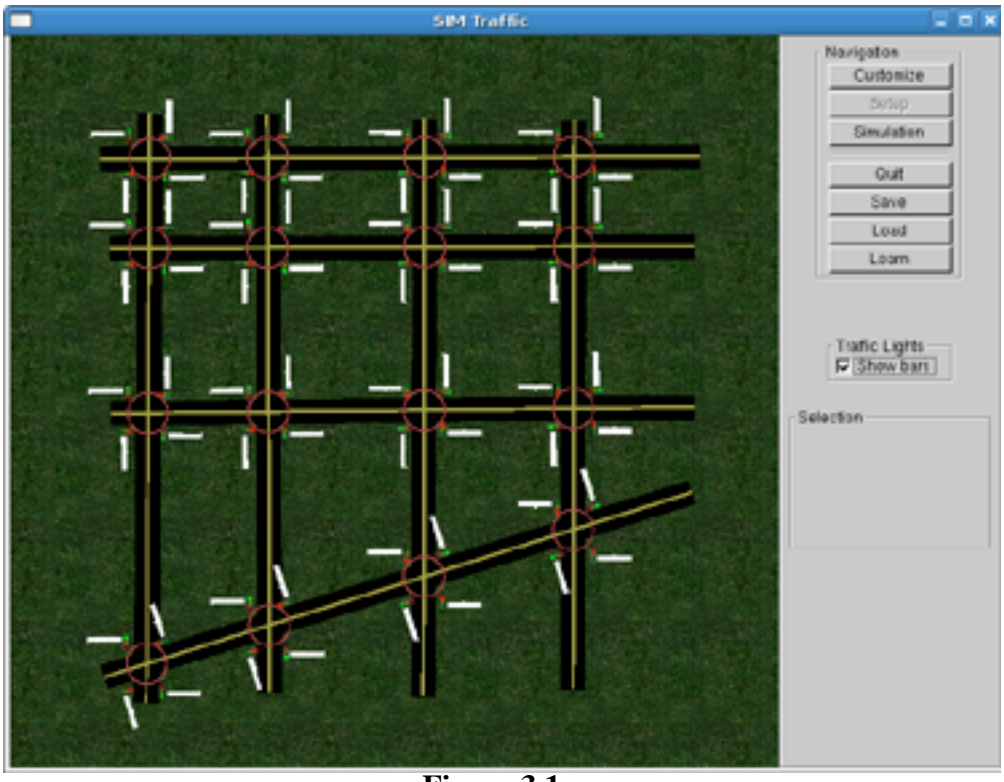

Figure 3.1

Throughput balancing did not produce useful policies. There are two primary reasons for this. First, the assumption of fixed-time periods for light cycles is likely far too strong. Also, the hypothesis class for light policies does not permit offsets between lights, i.e. all lights start their cycle's at the same time. Consequently, these policies assume a fixed order, that is the algorithm will never find a policy where a light starts inverts its cycle to be north-south then east-west if it started in east-west followed by north-south. Given the algorithm's inability to achieve improvement and the significant limitations of the class of policies, we believe that the modeling assumptions induce sizable modeling error.

Tests resulting from simulated annealing also showed no improvement over time though they did reveal significant obstacles produced by the simulator. We found that, if we allowed the random number generator seed to vary between runs, we saw significantly different results. For example, without changing the policies of any lights, one run moved about 220 combined vehicles through the 16 intersections of the map in ten minutes of simulated time while another moved 60. This instability is shown in Figure 3.2. The baseline throughput for each round of simulated annealing is shown as are the vote cast based the throughput metric and the final decision to accept the change in the parameters of a light or not. If a change was accepted at an iteration, a point appears in the upper row and if it was rejected, in the lower row. Therefore, we would expect to see the throughput metric stay approximately the same in the next round if the final decision was the reject the

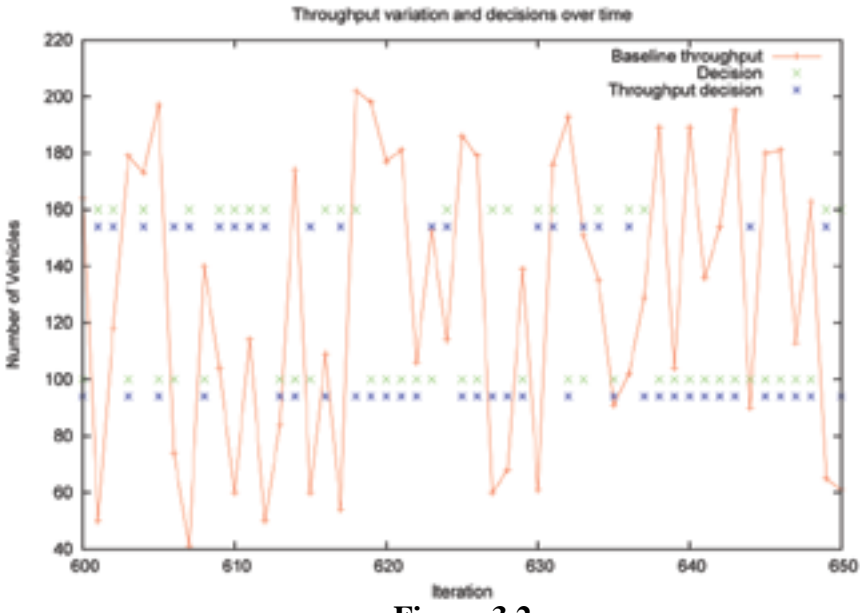

Figure 3.2

change. This is not what we observed.

The Markov Decision Process also did not perform as well as hoped. It is possible that modeling the decision of an adjacent light to change as a random variable allows too much information to be lost. Further, we found the establishing the reward for a state to be difficult given the instability of simulation conditions over varied random seeds.

In the end, we have doubts that the search space for policies is amenable to machine learning models performing optimisation. While local optima are a nontrivial problem, this issue is dwarfed by the problem of instability in the simulator. Since we do not have a simulator that was subjected to a rigourous verification and validation, it is not possible for use to say whether the high variance over random seeds was an effect of the simulator itself or rather that the nature of traffic is such that the fine details are of primary importance. Thorpe (Thorpe 97) notes that small changes in traffic light policies can lead to large changes in traffic congestion. While he sights this as a reason to seek improvement through automated learning, we are leery that small changes in policy can lead to large, often unpredictable changes in traffic congestion, making this search space very difficult for machine learning techniques.

4. References

Chui, S., T. Hsiang, J. Shen. Traffic Sim. Stanford University, 2004.

Puterman, M. Markov Decision Processes: Descete Stochastic Dynamic Programming. John Wiley & Sons, 1994

Thorpe, T. Vehicle Traffic Light Contro Using SARSA. Mathers Thesis, Department of Computer Science, Colorado State University, 1997.