ELMOPP: An Application of Graph Theory and Machine Learning to Traffic Light Coordination

Fareed Sheriff

July 28, 2020

Abstract

Traffic light management is a broad subject with various papers published that put forth algorithms to efficiently manage traffic using traffic lights. Two such algorithms are the OAF (oldest arrival first) [5] and ITLC (intelligent traffic light controller) [1] algorithms. However, a major flaw in many traffic light algorithms is that they do not consider future traffic flow and therefore cannot mitigate traffic in such a way as to reduce future traffic in the present. The Edge Load Management and Optimization through Pseudo-flow Prediction (ELMOPP) algorithm aims to solve problems detailed in previous algorithms — through machine learning with nested long short-term memory (NLSTM) [7] modules and graph theory, the algorithm attempts to predict the near future using past data and traffic patterns to inform its real-time decisions and better mitigate traffic by predicting future traffic flow based on past flow and using those predictions to both maximize present traffic flow and decrease future traffic congestion. Furthermore, while ITLC and OAF require the use of GPS transponders; and GPS, speed sensors, and radio, respectively, ELMOPP only uses traffic light camera footage, something that is almost always readily available in contrast to GPS and speed sensors. ELMOPP was tested against the ITLC and OAF traffic management algorithms using a simulation modeled after the one presented in [1], a single-intersection simulation, and the collected data supports the conclusion that ELMOPP statistically significantly outperforms both algorithms in throughput rate, a measure of how many vehicles are able to exit inroads every second.

Introduction

Traffic lights used specifically for road systems have been around since the 19th century with the installation of a gas-lit traffic light in London. This traffic light was a single light that controlled horse-drawn carriage traffic and was prone to explosions. This was soon followed by the first electric light, installed in Ohio in 1914; this was also a standalone traffic light. The first network of traffic lights was implemented in Salt Lake City, Utah in 1917 as a collection of six traffic lights controlled through a manual switch. The purpose of traffic lights was to control traffic to prevent jams and decrease the risk of accidents. This is a heavy task to conduct manually because it requires that humans decide the optimal or even just an adequate configuration of traffic light timings to minimize traffic congestion, which becomes increasingly more difficult to manage as the number of traffic lights increase [3].

As a result, the synchronization of traffic lights has largely been relegated to computer systems that take in real-time data and attempt to optimally coordinate traffic lights to reduce traffic congestion. One of the most commonly-used systems for this task is the Sydney Coordinated Adaptive Traffic System (SCATS), which is a combination of myriad technologies to produce real-time metrics used in tracking and managing traffic created around 1970 and used around the world. These include specialized controllers and in-road sensors. Due to the sheer volume of data and equipment necessary to set up and run the system, SCATS, though it has been implemented for thousands of traffic lights, is extremely costly and requires physical modifications to existing road systems to function at full capacity [8]. Although SCATS has shown convincing results for its efficacy, it is simply far too costly and time-consuming for many areas to use. Furthermore, SCATS does not attempt to mitigate traffic congestion using future predictions, which necessarily bottlenecks the system's ability to mitigate traffic. As a result, a low-cost traffic light coordination system that takes into account predicted future traffic is necessary.

Another algorithm that requires significantly less resources with promising results was detailed in *An Intelligent Traffic Light Scheduling Algorithm Through VANETs* [1]. This algorithm, however, requires that vehicles have GPS transponders, not unrealistic in this day and age but nevertheless problematic especially if some drivers do not use a GPS or turn it off for personal reasons. Another problem, not currently considered to be of paramount importance, is the lack of attention paid to future vehicle movement. It is a valid and logically sound argument to assume that traffic diverted away from areas that will see high densities of traffic in the near future will result in better traffic flow as the lower the density, the lower the average wait time per car at intersections because the greater the density of cars on a road, the slower the car will travel. Very few traffic light coordination system have as of yet accounted for this important factor.

Algorithm

The algorithm this paper presents, Edge Load Management and Optimization through Pseudo-flow Prediction (ELMOPP), aims to solve the problems detailed in previous algorithms and systems — through machine learning with nested long short-term memory modules (LSTM), the algorithm attempts to predict the near future using past data and traffic patterns to inform its real-time decisions and better mitigate traffic. The algorithm is also intended to easily scale to larger road systems. Furthermore, the algorithm only requires use of nearly ubiquitous traffic cameras to obtain all of the information it needs as not only are traffic cameras common on traffic lights, but if there were no traffic camera on a traffic light it would cost very little to install one. Because of this, the algorithm and associated system cost very little to install and implement, far less than most other traffic management systems, which require access to far more equipment and computing power.

It seeks to improve traffic flow by treating intersections as a set of four possible traffic light configurations and choosing the best short-term goal at each intersection in conjunction with all of the vertex's neighbours, which it then applies to every vertex in the road network. In this way, it reaches a local-global optimum because each vertex's decision is based on the states of its neighborhood, which allows the algorithm to approximate a global optimum.

A road system may be modeled as the induced directed graph $G = (V, E)$ of the road network where V is the vertex set of the digraph representing all intersections of the systems while E is the directed edge set contain all directed roads connecting each intersection.

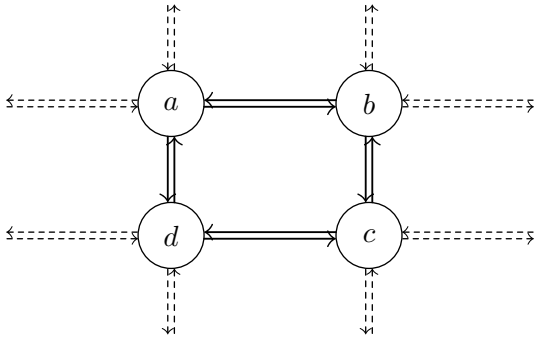


Figure 1: A sample induced digraph G_α of a road system

Note that the dashed edges connected to each vertex represent roads that are only connected to one intersection/vertex; as a result, these directed edges are termed "pseudo-diedges" as they do not fit the traditional definition of a directed edge but they are roads nonetheless. Pseudo-diedges are not included as part of the induced digraph of a road system and are drawn as a formality: they are not directly considered in calculations and are only indirectly considered through in-flow predictions of directed edges to which these pseudo-diedges form a path. Also note that the "edges" referenced prior to this point were considered edges that mapped to a single road. From this point onward, edges will be considered in both the context of roads and lanes within those roads. In this way, an edge may contain multiple edges termed subedges that map to road lanes.

The adjacency matrix of the induced digraph contains all vertex-vertex connections. This paper follows the row-tail, column-head adjacency matrix convention, where each row represents vertices marking the start of a diedge and each column represents vertices marking the head of a diedge. For example, the adjacency matrix A_α of the graph in Figure 1 is

$$A_\alpha = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

while the vertex set V_α is $\{a, b, c, d\}$ and the directed edge set E_α is $\{(a, b), (b, a), (a, d), (d, a), (b, c), (c, b), (c, d), (d, c)\}$. However, because each road has four possible directions it can take (forward, left + U-turn, and right), every edge is represented as a 3-vector containing the partitioning of each road into its lanes so that every component of the 3-vector is a subedge, where the first component represents the number of vehicles in the left lane/s, the second component represents the number of vehicles in the middle lane/s, and the third component represents the number of vehicles in the right lane/s. This turns the adjacency matrix of the induced digraph into an adjacency tensor of order three, hereon represented as a matrix of vectors. Therefore, a possible adjacency tensor of G_α could be

$$A_\alpha = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

The elements of A_α are shown as row vectors only for comprehension; the transposition of these vectors has no effect on meaning.

Likewise, define a capacity tensor $C^{i\times j\times k}$ that maps a maximum vehicle capacity to every edge in the adjacency tensor and a variable quantity tensor $Q^{i\times j\times k}$ that contains the number of vehicles on every road lane at time t. It follows that the load L of G equals the Hadamard quotient of the quantity and the capacity[1].

$$L = Q \oslash C \tag{1}$$

As an example, the quantity, capacity, and load tensors for G_α could be

$$Q_\alpha = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & 0 & 0 \\ 7 & 7 & 7 \end{bmatrix} & \begin{bmatrix} 0 & 2 & 3 \\ 0 & 0 & 0 \\ 2 & 1 & 3 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 1 \\ 0 & 0 & 0 \\ 3 & 2 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 6 & 4 \\ 0 & 0 & 0 \\ 4 & 5 & 4 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

$$C_\alpha = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 7 \\ 0 & 0 & 0 \\ 9 & 8 & 7 \end{bmatrix} & \begin{bmatrix} 0 & 4 & 4 \\ 0 & 0 & 0 \\ 3 & 5 & 3 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 3 & 2 & 4 \\ 0 & 0 & 0 \\ 4 & 5 & 5 \end{bmatrix} & \begin{bmatrix} 3 & 9 & 6 \\ 0 & 0 & 0 \\ 7 & 8 & 9 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

$$L_\alpha = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 3/7 \\ 0 & 0 & 0 \\ 7/9 & 7/8 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 1/2 & 3/4 \\ 0 & 0 & 0 \\ 2/3 & 1/5 & 1 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 2/3 & 0 & 1/4 \\ 0 & 0 & 0 \\ 3/4 & 2/5 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 2/3 & 2/3 \\ 0 & 0 & 0 \\ 4/7 & 5/8 & 4/9 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

When a traffic light is active, it is often the case that another traffic light at the same intersection could also be active. For example, at a four-way intersection the left lane green light may be active for two antiparallel inroads. More specifically, the only two possible non-intersecting configurations for a pair of antiparallel inroads to an intersection are both left lanes active and both middle and right lanes active. This may be seen in US road system traffic light coordination. A traffic light configuration is defined to be a set of subedges directed toward a common vertex so that none of the subedges' traffic streams intersect. The

[1]The load of an edge is artificially set to 0 when the capacity of an edge is 0 as it is not possible for any vehicles to travel on that edge. In setting the edge load to 0 when the capacity is 0, the urgency of that edge becomes 0, preventing so-called "dead" edges from having any effect on urgency.

activation of a traffic light configuration is defined to be the activation of each traffic light corresponding to its associated subedge in a configuration. There are eight such configurations for a four-way intersection: two for two antiparallel inroads, two for the other pair of antiparallel inroads, and four for each inroad. The configuration set for a vertex v is symbolized $C(v)$ and contains all valid configurations for a vertex while a specific configuration on v is symbolized C_v, where $C_{v_n} \in C(v)$.[2]

Define the urgency of subedge e_n as a function of the load of the subedge and the time T since the last activation of a subedge so that the urgency of subedge e_n, $U(e_n)$, is defined to be

$$U(e_n) = \frac{L(e_n)}{e^{\frac{T(e_n)}{t_{max}}-1}} = L(e_n)e^{1-\frac{T(e_n)}{t_{max}}} \tag{2}$$

and the urgency of orientation C_{v_n}, $U(C_{v_n})$, is defined to be

$$U(C_{v_n}) = \sum_{e_m \in C_{v_n}} U(e_m) \tag{3}$$

where $L(e_n)$ is the load of subedge e_n, $T(e_n)$ is the time passed since e_n was last activated, and t_{max} is the maximum legal activation time for a traffic light. The urgency of an edge configuration is defined as the sum of the urgencies of all elements of the edge's configuration set. Then, the algorithm simply chooses the configuration with the highest urgency for vertex v, C_{v+}

$$C_{v+} = \max_{C_v \in C(v)} U(C_v) \tag{4}$$

This configuration is held until

$$\max_{C_v \in C(v) \setminus C_{v+}} U(C_v) \geq U(C_{v+}) \tag{5}$$

for

$$t \in (t_{min}, t_{max}) \tag{6}$$

otherwise,

$$t = \underset{p \in \{t_{min}, t_{max}\}}{\arg\min} |p - t| \tag{7}$$

where t_{min} is the minimum legal activation time for a traffic light. This is so that the configuration is activated until the urgency directly before the configuration was activated equals the urgency of another configuration directly before it was activated, while keeping in mind minimum and maximum green light length rules. The algorithm is conducted in real-time, so there is no need for a model of unload speed; rather, real-time data is used and the rules above are used when monitoring load. In fact, because the algorithm is conducted in real-time, the modelling of out-flow may be safely ignored in its entirety as no modelling is required when real-time data is available. As a result, it suffices to only consider edge & subedge in-flow in the algorithm.

The previously-detailed algorithm is a naïve algorithm similar to those of various systems used to coordinate traffic lights; however, these algorithms fail to consider future traffic and how to best prevent traffic density from increasing through future predictions. As may be obvious, the intersection-road induced graph fails to consider the number of vehicles at places like department stores or office buildings because

1. taking those places into account would result in an overly complex graph and overall system and
2. violating the conservation of graph flow allows for generalized predictions to be made by relegating the entrances and exits of vehicles into and from buildings to negative and positive edge flow.

[2]Note that, although four-road intersections are most often mentioned in this paper, intersections of any possible number occur in the real world. Four-road intersections are used as examples here simply because they are common and it is easy for audiences to understand the ELMOPP algorithm through easily-relatable examples. However, the idea of a configuration is valid for all types of intersections. All that needs to be done is define the configuration set accordingly.

Therefore, rather than calling the movement of vehicles between edges flow, which it inherently is not due to the lack of conservation of graph flow, flow shall hereon be considered to be pseudo-flow, a construct defined to be equivalent to flow in all respects save for obeying the conservation of flow. It is then possible to create a vector each of whose elements correspond to the flow of a certain edge over time. Furthermore, the closer in time the future flow of an edge is to the current time of an edge, the greater the effect the future flow will play in the configuration decision at an intersection. Furthermore, the effect of future edge flows may be modelled using a bounded sine function whose maximum occurs at the current time and minimum at the maximum configuration activation time. The cumulative urgency U_c of a given path may be expressed as the convolution integral transform between current and future urgencies and a negatively-sloped line whose area from the origin to the x-intercept equals 1, effectively modeling a triangle with legs on the x- and y-axes and base t_{max}. With a bit of elementary geometry, it is shown that the height of the triangle must be

$$bh = ht_{max} = 2 \tag{8}$$

$$h = \frac{2}{t_{max}} \tag{9}$$

Therefore, the equation of the line is

$$y = h - \frac{h}{b}x = 2\left(\frac{1}{t_{max}} - \frac{x}{t_{max}^2}\right) \tag{10}$$

The cumulative urgency convolution then becomes

$$U_c(e_n) = (\triangle * U)(t) = 2\int_0^{t_{max}} U_t(e_n)\left(\frac{1}{t_{max}} - \frac{x}{t_{max}^2}\right)dt \tag{11}$$

where

$$U_0(e_n) = U(e_n) \tag{12}$$

and

$$U_t(e_n) = L(e_{n_t})e^{1-\frac{T(e_n)}{t_{max}}} \tag{13}$$

where e_{n_t} is the subedge e_n at time t.

Future flow is predicted using a recurrent neural network (RNN) due to the fact that the current in-flow of a directed edge both is roughly periodic in nature and is a product of previous in-flows. For example, it is to be expected that certain roads will see above-average in-flows during rush hour. Similarly, in-flows cannot stay high for long periods of time if the in-flows of other roads have been low because the low in-flows of other roads means that the road with high in-flow is getting saturated and is approaching capacity. The RNN used is, to decrease computation times, simplify calculations, and take into account the inherently discrete nature of traffic light timing, a single triply-nested long short-term memory (NLSTM) unit [7]. The reason behind the use of a triply-nested LSTM rather than a conventional LSTM is that the time-steps used in the model occur on the order of minutes, but patterns in traffic in-flow often occur on the order of days, months, or even years. One oft-noted problem with vanilla LSTMs is that although they are competent at recognizing patterns occurring over hundreds of time-steps, their long-term memory fail to remember patterns spanning thousands or more time-steps [4]. Nested LSTMs present a solution to this problem by replacing the long-term memory vector with another LSTM, thus increasing the long-term memory span of the nested LSTM. The reason behind using a triply-nested LSTM is because the span over which patterns occur ranges from minutes to years. Because there are around $365 * 24 * 60 * 60 = 32,850,000$ minutes or time-steps in a year, which is close to $\log_{500} 32,850,000 \approx 2.7784 \approx 3$ magnitudes greater than a single LSTM can handle, it is necessary to have at least three levels of LSTMs to properly account for patterns spanning long ranges of time. Due to the discreteness of both the RNN and the configuration activations, the cumulative urgency integral must be transformed into a discrete convolution summation

$$U_c(e_n) = (\triangle * U)[t] = 2\sum_{t=0}^{t_{max}}\left[U_t(e_n)\left(\frac{1}{t_{max}} - \frac{x}{t_{max}^2}\right)\right] \tag{14}$$

The cumulative urgency is substituted for the naïve urgency when relegating intersection configuration so that C_{v+} becomes

$$C_{v+} = \max_{C_v \in C(v)} U_c(C_v) \tag{15}$$

Simulation & Data Analysis

A simulation will be conducted to test the detailed algorithm. This simulation will be of the same form as the simulation detailed in [1] so as to facilitate direct comparison between the two algorithms. The simulation graph G_S is the induced digraph of the simple 4-star, meaning its adjacency tensor is of shape $5 \times 5 \times 3$.

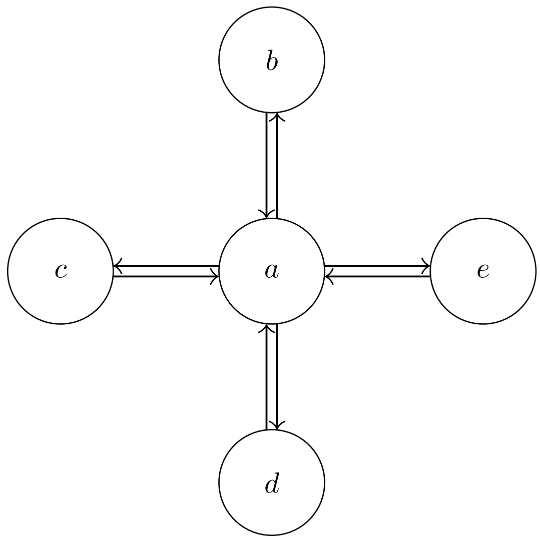


Figure 2: The induced digraph G_S of the simple 4-star, used to simulate the algorithm

$$A_S = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

Note that because the simulation detailed in [1] only considers traffic inbound to vertex a, the algorithm will not account for outgoing traffic originating from a; in other words, the loads of all outgoing edges from a are artificially set to a constant 0. Just like the simulation detailed in [1], this simulation considers the length of each time-step to be a second; however, unlike the simulation, this simulation does not randomly generate per-second inflows because road inflow in real life is not random. The simulation time of the provided simulation is 2000 seconds/time-steps while the total number of vehicles is 200 to 1,000, with a set of thirty trials conducted. Therefore, a periodic in-flow function will be enacted to better simulate real-world traffic inflow. This in-flow function is a 4-dimensional autonomous hyperchaotic system modelled off the Lorenz attractor as described in *On the dynamics of new 4D Lorenz-type chaos systems* [10]. That system of differential equations (provided below) exhibits chaotic behaviour, which models the real world.

$$(x, y, z, w) \coloneqq \begin{cases} \frac{dx}{dt} = a(y - x) - ew, \\ \frac{dy}{dt} = xz - hy, \\ \frac{dz}{dt} = b - xy - cz, \\ \frac{dw}{dt} = ky - dw \end{cases} \tag{16}$$

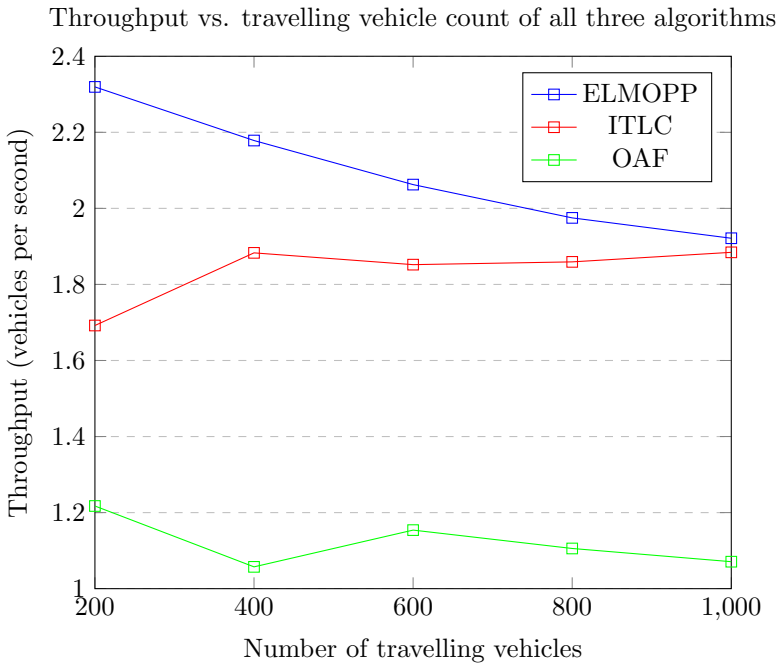
where "a, b, c, d, e, h are positive parameters of system" [10].

To conform to the simulation used, the hyperchaotic system will be normalized by dividing the values generated at each timestep by the integral of the system over the interval $[0, 2000]$, then will be scaled by a random integer over the interval $[200, 1000]$. This results in chaotic inflow with, as described in the simulation detailed in [1], a total number of incoming vehicles between 200 and 1,000. The capacity of each road was

not given in the original paper's simulation description; therefore, the capacity of every inroad to a is set to be a constant $1,000$. Note that the value of the capacity itself may be arbitrarily positive because all that matters is that the capacities themselves are equal across all edges, as only capacity ratios are considered as can be seen from the urgency formula. The simulation previously described also contains other information, such as the area over which the simulation is conducted, variables that are extraneous to this simulation as ELMOPP does not require these variables.

The LSTM used in the simulation will not be a triply-nested LSTM, but will instead be a vanilla LSTM because the hyperchaotic system will be timewise scaled down to produce patterns over time intervals ranging from minutes to hours due simply to the length of the simulation detailed in [1] (around 33 minutes over 2,000 timesteps). As a result, the RNN used will not need to be able to handle large time scales. More details on the simulation are provided in the appendix.

The results from the simulation were collected and compared against the results from [1], as shown in the plot below. The data collected were plotted against the results of the ITLC and OAF algorithms.



Furthermore, an independent samples t-test was conducted for each pair of algorithms. The mean of the ELMOPP algorithm is 2.09133 and the standard deviation 0.158824. The mean of the ITLC algorithm is 1.83414 and the standard deviation is 0.080730. Finally, the mean of the OAF algorithm is 1.12108 with a pooled standard deviation of 0.063498. Thirty trials were conducted for each data point provided for every algorithm. Three independent samples t-tests were conducted for every pair of algorithms. It was assumed that the data distributions and the variances for each algorithm were different when conducting the tests, supported by the calculated standard deviations. The null hypothesis was that the means of the distributions were equal while the alternate hypothesis was that the greater mean was actually greater than the lesser mean. Therefore, the t-test was one-sided. Finally, the degrees of freedom were set to 298 for each test as there were 150 data points for each dataset and the level of significance was set to 0.05. The results of the t-tests are shown below.

Test	**Calculated t-value**	**Table t-value**	**Significance**
t-test: ELMOPP v. ITLC	17.6799	1.6500	Significant
t-test: OAF v. ELMOPP	69.4727	1.6500	Significant
t-test: ITLC v. OAF	85.0275	1.6500	Significant

At the 95% confidence level, it is seen that the alternate hypothesis is supported for each pair of al-

gorithms. This, coupled with the algorithm's means in order from greatest to least as ELMOPP, ITLC, and OAF, supports the hypothesis that ELMOPP exhibits a higher throughput than both ITLC and OAF. Surprisingly, the calculated t-value for the ITLC v. OAF test is the greatest of all calculated t-values. This is because, even though the difference of means for this test is smaller than the difference of means for the ELMOPP v. OAF, the variance for the ITLC dataset is also smaller than the variance for the ELMOPP dataset.

Conclusion

This paper outlined a novel traffic management algorithm named ELMOPP, which used graph theory and nested LSTMs to predict and attempt to better manage and forecast traffic. In contrast to the ITLC algorithm, which assumes vehicles have access to a wireless transceiver and GPS tool as well that vehicles are able to communicate with the traffic lights themselves by sending and receiving traffic-related data, the ELMOPP algorithm assumes only that current traffic flow is correlated with past traffic flow and that traffic cameras are able to see incoming traffic and correctly determine the number of cars at an intersection, the latter of which is not only possible but has been proven a reality with the advent of image recognition and classification technology. All in all, the data supports the conclusion that the ELMOPP algorithm statistically significantly outperformed both the ITLC and OAF algorithms for traffic management simulation and did so with fewer available metrics. Note, however, that this paper only addressed traffic management performance in the context of a simulation; no physical, real-world testing occurred, something that would fall under the banner of further research.

Applications & Further Research

Applications of the novel ELMOPP algorithm are varied in scope. As a traffic management algorithm built to keep traffic flow in road systems as high as possible, ELMOPP could be used on practically any road system. However, ELMOPP would be especially useful in places that do not have access to GPS systems or aren't able to pay for extreme renovations to road systems as would be required by systems like SCATS. This includes places such as cities with low funds or towns with minimal extra resources. In fact, ELMOPP could be applied to any road system that needs a traffic management system quickly as it is straightforward to implement and requires practically no physical modifications to existing road systems. The observed increase in throughput shown by ELMOPP in comparison with ITLC and OAF is beneficial for another reason: environmental impact. Increased traffic congestion has been shown to correlate with lower ambient air quality and has even been linked to trends in increasing mortality. Hazards associated with traffic congestion have been shown to be related to travel time and rush hour length, among others [11]. As ELMOPP seems to show a higher throughput than ITLC and OAF, it is safe to say that travel time will be reduced as more vehicles at any given moment are travelling to their destination for ELMOPP than for either of the other two algorithms. Furthermore, ELMOPP was specifically created to predict and mitigate future traffic, something that has immediately-obvious implications for rush hour traffic. This further supports the application of ELMOPP to decreasing environmental impact and positively impacting people's health.

Further research on the subject of traffic management algorithms would likely involve different methods of approaching and describing the problem. ITLC used algebra and basic combinatorics to create what was a very loose description of road systems; ELMOPP used graph theory and linear algebra to describe and optimize traffic. There are other, potentially more scalable, methods of describing traffic flow. Further research by the researcher would likely be focused on the avenues of other methods of predicting traffic flow, such as through vanilla artificial neural networks (ANNs) or graph convolutional networks (GCNs) [6]. One specific venue for further research could be real-world testing by implementing ELMOPP for a physical road system and comparing its performance to other traffic management algorithms.

References

[1] M. Bani Younes and A. Boukerche. "An Intelligent Traffic Light scheduling algorithm through VANETs". In: *39th Annual IEEE Conference on Local Computer Networks Workshops*. 2014, pp. 637–642.

[2] Ennio Cascetta. "Transportation Systems Engineering: Theory and Methods". In: 2014. Chap. 3: Traffic Stream Models. ISBN: 9781475768732.

[3] Larry Clark. *Traffic signals: A brief history*. 2019. URL: https://magazine.wsu.edu/web-extra/traffic-signals-a-brief-history/.

[4] K. Greff et al. "LSTM: A Search Space Odyssey". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10 (2017), pp. 2222–2232.

[5] V. Hemakumar and H. Nazini. "Optimized traffic signal control system at traffic intersections using VANET". In: *IET Chennai Fourth International Conference on Sustainable Energy and Intelligent Systems (SEISCON 2013)*. 2013, pp. 305–312.

[6] B. Jiang et al. "Semi-Supervised Learning With Graph Learning-Convolutional Networks". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 11305–11312.

[7] Joel Ruben Antony Moniz and David Krueger. "Nested LSTMs". In: *Proceedings of the Ninth Asian Conference on Machine Learning*. Ed. by Min-Ling Zhang and Yung-Kyun Noh. Vol. 77. Proceedings of Machine Learning Research. PMLR, Nov. 2017, pp. 530–544. URL: http://proceedings.mlr.press/v77/moniz17a.html.

[8] New South Wales Government. *SCATS: Sydney Coordinated Adaptive Traffic System*. 2011. URL: https://www.qtcts.com.au/media/512152-RTA532_SCATS_A4_Product_Brochure_07.pdf.

[9] Axel Wolfermann, Wael Alhajyaseen, and Hideki Nakamura. "MODELING SPEED PROFILES OF TURNING VEHICLES AT SIGNALIZED INTERSECTIONS". In: Jan. 2011.

[10] Guangyun Zhang et al. "On the dynamics of new 4D Lorenz-type chaos systems". In: *Advances in Difference Equations* 2017.1 (Aug. 2017). DOI: 10.1186/s13662-017-1280-5. URL: https://doi.org/10.1186/s13662-017-1280-5.

[11] Kai Zhang and Stuart Batterman. "Air pollution and health risks due to vehicle traffic". In: *Science of The Total Environment* 450-451 (Apr. 2013), pp. 307–316. DOI: 10.1016/j.scitotenv.2013.01.074. URL: https://doi.org/10.1016/j.scitotenv.2013.01.074.

Appendix

Hyperchaotic System Inflow Simulations

The hyperchaotic system governing inflow dynamics for the simulation was

$$(x, y, z, w) \coloneqq \begin{cases} \frac{dx}{dt} = a(y - x) - ew, \\ \frac{dy}{dt} = xz - hy, \\ \frac{dz}{dt} = b - xy - cz, \\ \frac{dw}{dt} = ky - dw \end{cases} \tag{17}$$

with the specific attractor a = 5, b = 20, c = 1, d = 0.1, e = 20.6, h = 1, and k = 0.1. As stated by the paper, the largest Lyapunov exponent of the attractor above is 0.24. As a result, the Lyapunov time of the system is $1/0.24 \approx 4.167$. Because each system time-step is 0.01, the Lyapunov time expressed in time-steps is $4.167/0.01 \approx 417$, which amounts to $417/60 \approx 6.95$ minutes, far greater than any reasonable maximum traffic light activation time. As a result, the cumulative urgency formula is not expected to significantly diverge from the true chaotic distribution as the Lyapunov time is far greater than any expected result.

The system was solved by using the Runge-Kutta family's Euler method with a step size of 0.01 over $[0, 2000]$ by treating the system as the derivative of a vector-valued function.

Euler's method allows for discrete approximations of differential equations. It states that for function y and its derivative y',

$$\mathbf{h}_{n+1} = \mathbf{h}_n + \mathbf{h}'(t_n)\Delta_n \tag{18}$$

This may be extended to vector-valued functions, such as the previously-described hyperchaotic system [10], where

$$\mathbf{h} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}, \qquad \mathbf{h}' = \begin{bmatrix} a(y - x) - ew \\ xz - hy \\ b - xy - cz \\ ky - dw \end{bmatrix} \tag{19}$$

and $\Delta_n = 0.01$.

The system was normalized by dividing the system by the L1 norm of its integral over $[0, 2000]$, then scaling it up by a factor of 800. This effectively modeled chaotic vehicle inflow so that the total inflow over the 2,000 timesteps was 800 vehicles. The integral of the solution was calculated by summing all of the data points calculated through Euler.

Because the step size was 0.01 and the stiffness ratio (ignoring the eigenvalue of 0) was $-7.56/0.23 \approx -32.8695$, which is not significantly less than -1, it is safe to say that the system is non-stiff for the estimation technique used.

LSTM Prediction Model

The LSTM prediction model trained on 10,000 data points. The data was generated by choosing a random real vector $v \in \mathrm{R}^4$ so that v is a vector randomly sampled from the 4-dimensional hypercube whose boundaries are defined by the fourfold Cartesian product $\{\{0, 1\} \times \{0, 1\} \times \{0, 1\} \times \{0, 1\}\}$. The LSTM's training data was split 80-20 as traditionally split when training and testing data for machine learning models. The first 8,000 data points were used only training while the last 2,000 points were used for simulation. Every data point part of the last 2,000 was trained pointwise on the LSTM as a single-sample batch.

Traffic Intersection Simulation

Very few details were given on the ns2 map used in [1]; as a result, some assumptions were made about the simulation. The turning speed was set at a constant 10 meters per second, chosen from the thorough analysis done in [9]. Furthermore, the outflow rate (q) was calculated using a numerical Greenshield model [2] to be

$$q = kv_f \left(1 - \frac{k}{k_j}\right) \tag{20}$$

where k is the density, v_f is the free-flow speed, and k_j is the jam density. The capacity of each street was set to 1,000 vehicles. Each inroad was initialized to a load of 200 vehicles with the total vehicle inflow over the 2,000-second testing interval being 800, with a resulting 4,000 vehicles total and an average of 1,000 vehicles per inroad, only accomplished if all 4,000 vehicles flow out over the 2,000-second interval. The capacity for each inroad was set to 1,000 with a jam density of 1.