A Stable Longest Queue First Signal Scheduling Algorithm for an Isolated Intersection

R. Wunderlich[1], I. Elhanany[2]
Electrical Engineering and Computer Science Department
University of Tennessee, Knoxville
Knoxville, TN 37996
e-mail: {[1]rwunderl,[2]itamar}@utk.edu

T. Urbanik II
Civil and Environmental Engineering Department
University of Tennessee, Knoxville
Knoxville, TN 37996
e-mail: turbanik@utk.edu

***Abstract*— There have been countless efforts directed toward efficiently controlling the flow of traffic through an intersection. This paper describes an algorithm designed for the signal control problem that employs concepts drawn from the field of computer networking. The novel method proposed utilizes a maximal weight matching algorithm to minimize the queue sizes at each approach, yielding significantly lower average vehicle delay through the intersection. Lyapunov function-based analysis is provided, deriving the conditions under which the system is guaranteed to be stable. The algorithm is compared to an optimized fixed time controller using the VISSIM traffic simulation environment. Simulation results clearly demonstrate the performance gain obtained when using the proposed scheme, particularly in the presence of the non-uniform traffic scenario proposed.**

I. Introduction

The first traffic signals were manually operated mechanical signs erected in the late nineteenth century [1]. The first coordinated lights appeared in the early twentieth century; the system consisted of three consecutive lights that could be traversed without stopping while driving at just twenty miles per hour [1]. These signals were usually operated by police officers, and were prone to mechanical failure. With the advent of modern computerized traffic signaling systems, and due also to the immense amount of traffic that now pulses through the streets that they control, new and more complex control methods are being proposed. All of these methods share a common goal: to maximize the traffic throughput at controlled intersections in the shortest amount of time while maintaining driver safety.

Without the ability to test the new and increasingly complex control techniques on live traffic flows (due to obvious safety concerns), it becomes necessary to use computers to simulate traffic flows in order to facilitate the light cycle testing and verification process. The standard method of signal timing has been the optimization of traffic cycles in off-line computations according to statistical measures of traffic flows under certain conditions such as morning traffic, rush hour, etc. Controllers programmed with several different cycles can choose the cycle most appropriate for the current traffic conditions. Moreover, many controllers have the ability to modify the light cycle depending on detection of vehicles, the time of day, as well as other factors.

With adavances in traffic detection technologies, adaptive signal control algorithms have been extensively studied. The adaptive signal control outperforms pretimed signal control both in perfect knowledge in vehicle arrivals and in imperfect knowledge of the currect traffic situation [2]. A novle simulation study has evaluated the preformances of real-time adaptive traffic signal control algorithms used in OPAC, PRODYN, ALLONS_D and COP in isolated intersection [3].

In recent work, the problem of scheduling traffic at an intersection has been addressed by structuring the problem as a Markov decision process (MDP) [4]. It has been shown that by using dynamic programming techniques, which aim to solve the Bellman equation given a stochastic model of the system, an optimal control strategy can be obtained [5]. However, in real life, a model of the system is not provided. Approximating a model yields limited results due to the nonstationarity and non-Markovian characteristics of vehicular traffic flows at intersections.

In this paper we present LQ-MWM – an algorithm for scheduling signals at an isolated intersection so as to maximize the traffic throughput while minimizing the average latency experienced by the traversing vehicles. In particular, we employ a queue size based maximum weight matching (MWM) framework, which has been drawn from the field of data packet switching. We derive the stability properties of the algorithm and demonstrate its performance on uniform as well as non-uniform vehicular traffic patterns.

The rest of the paper is structured as follows. In section II a description of the intersection model is provided. Section III discusses the signal cycle attributes and constraints, while in section IV the algorithm is described and its stability properties are obtained. Section V presents simulation results, and in section VI the conclusions are drawn.

II. System Model

A. Notation and Formulation

Prior to describing our methodology and analysis, it is necessary to outline the notation used, to illustrate the configuration of the intersection used in the simulations, and to define some performance metrics upon which to base qualitative comparisons between the control methods. The simulation interface is also briefly discussed.

An intersection, such as the one depicted in Fig. 1, has several terms associated with it, which are now explained. The figure shows a four-way intersection with through lanes and separate left turn lanes. Each these lanes is called a movement, or an approach. Any of these movements that can be permitted simultaneously are grouped into what are called phases. For example, in this intersection, the straight and the right turn movements are grouped into one phase because they occupy the same physical lane. These phases are numbered according to the standard NEMA (National Electrical Manufacturers Association) convention. Intuitively it can be noted, however, that not all of these phases can be given a green light at once. Thus, traffic signals are used to display the current right-of-way to the vehicles at the intersection. The signals associated with a phase, which all show the same signal at the same time during an interval, are called a signal group. A signal controller controls all of the signal groups of an intersection.

The signal controller is programmed with information about the intersection and its right-of-way characteristics. This information includes a description of the cycle, which is a sequence of intervals. Phases are made up of sets of intervals, e.g. the green, yellow, and red times of a signal group. There are further considerations such as yellow time and intersection clear time that are modeled in the simulation and held constant across control methods. The yellow time and intersection clear time of the simulated intersection combine to a total of five seconds.

In the stability discussion below, for the purpose of clarity, the lanes of the intersection are referred to as links, and the intersection referred to as a node at which the links are connected. This is intended to generalize the proof and to not introduce confusion between physical lanes of the intersection and the overall input-output characteristics of the intersection as a whole. Other terms used in the algorithm's description include traffic flow rate and link capacity. The flow rate is a value that describes how much traffic is flowing on a particular link relative to the overall capacity of the link. The capacity of the link is defined to be the maximum number of vehicles that could possibly traverse a link within a certain amount of time. These quantities are usually described in terms of vehicles per hour.

B. Intersection Configuration

The intersection under consideration is illustrated in Fig. 1. The labeling of the signal groups in this intersection follows the NEMA convention. This intersection is an interesting case, not only because it appears often in real-world traffic networks, but also because its symmetry allows for a fairly straightforward analysis.

The traffic distribution is identical for all possible routes. Each entry lane is 400 m from its endpoint to the intersection. The left turn lanes provide 100 m of vehicle queue space to help avoid blocking at the branching point. All vehicles within 100 m of the signal are counted as being in the queue for that signal. The long leads into the intersection help to ensure that the arriving traffic is distributed properly, and that vehicles do not build up at the inputs of the network for simulation runs with traffic levels below the saturation level.

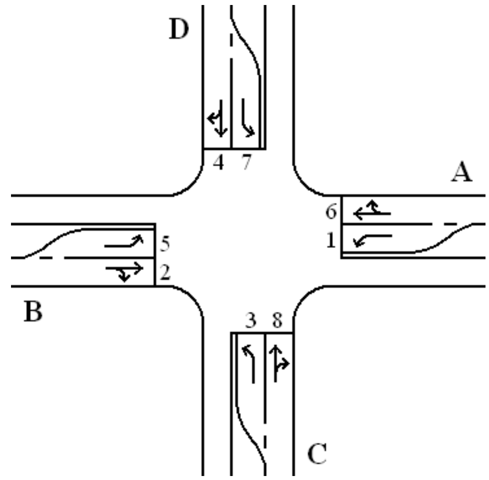

Fig. 1. Intersection with standard movement numbering.

C. Performance Metrics

Fundamental metrics for evaluating the performance of a traffic controller (particularly at an isolated intersection) include: vehicle delay, traffic throughput and average queue size. Analyzing the overall delay experienced by a vehicle that has traversed the network is a direct indication of how long the vehicle has had to wait at the intersection prior to being allowed through. This is true in our case due to the inherent symmetry of the network under consideration. The throughput measures the number of vehicles per hour that pass through an intersection, and is also indicative of the overall controller performance. The queue sizes are the most important metric that we study in this work. As expressed in Little's theorem, the queue size is directly proportional to the delay experienced by the vehicles in the network. Thus, to minimize the vehicle delays one should seek to minimize the queue sizes.

Although minimizing the queue sizes is the motivation of the LQ-MWM algorithm, the overall vehicle delay is the metric that we use to compare the performance of the two control methods. Because of the symmetry of the network, all of the vehicles should experience the same amount of delay as they traverse it. That is, if each vehicle were to travel through the network without stopping, they would all take the same amount of time regardless of the path taken. For this reason, we show the results in terms of the average vehicle delay.

D. Simulation and Control Interface

To test and compare control methods, we are using the VISSIM traffic simulation environment. VISSIM is a microscopic multi-modal traffic simulator that allows the user fine control over all aspects of the network, such as vehicle type, driver behavior, intersection control, and statistical data collection. The layout of the road links and connectors is first modeled in VISSIM (possibly drawn over aerial photography of the intersection of interest), the traffic composition is then defined, traffic volumes at the link entry points are set, and the signal controls are inserted at the intersections. Optionally, vehicle speed profiles and driver behaviors can be modified in order to better mimic specific conditions under which the controller is expected to operate. Different vehicle classes exist, including cars, trucks, busses, and heavy vehicles. The simulator even provides for hybrid traffic flows that include bicycles and pedestrian traffic.

The VISSIM simulator allows for many types of signal controllers to be used. These include, but are not limited to, a built-in fixed time controller, a standard NEMA controller, a VAP (Vehicle Actuated Programming) code controller, a virtual Econolite ASC/3 module, and even provides an interface for an external hardware controller. It is also possible for each intersection in the network to have a different type of controller, if desired. We compare our control method to a fixed time controller that has been optimized for the traffic flow under consideration.

Our control method relies on an algorithm, discussed in section IV below, the calculations for which are carried out in Matlab. Matlab provides a high-level programming language for interactive technical computing, and functions for algorithm development, data analysis and visualization, and numeric computation. VISSIM provides a COM (Component Object Model) interface in order to give control of the simulator to other programs. Using this COM interface from Matlab, we are able to load the traffic network, set simulation parameters, run simulations, and collect data. The control routine single-steps through the traffic simulation while controlling the signal group with the custom-designed control logic.

In order to control the many phases of the intersection, a simple VAP code controller monitors vehicle detectors in the network that are associated with each phase. The controller changes the state of the phase whenever the detector is tripped. In the simulator, the detectors are disabled so as to prevent vehicles from triggering them. Instead, when the controller determines that a phase should be changed, Matlab triggers the detector that, in turn, signals the VAP controller to change the state of the signal group. This control method provides no safeguards against unsafe phase changes other than the nature of the controller itself, which is fundamentally incapable of changing to a phase combination that is unsafe.

III. Traffic Cycle Attributes and Constraints

In order for the intersection to operate properly, and in an effort to ensure the safety of vehicles in the network, the traffic light cycle must be valid. Usually, the cycle consists of phases placed in a particular order, each interval of which is given some amount of cycle time. The cycle diagram for our intersection is depicted in Fig. 2. In this diagram, time progresses from left to right, and indicates that the left-turn phases become active before the corresponding straight/right phases (referencing the movement numbers of Fig. 1). A vertical barrier separates the East-West phases from the North-South phases. Phases are said to be compatible if they can be green concurrently without creating traffic flow conflicts. Each of the phases is compatible with the phases above or below itself and on the same side of the barrier. All other phase combinations are incompatible. The rows of the diagram are referred to as rings, which can be timed independently, so long as all rings cross the barrier at the same time. Careful observation reveals that there are only ten unique phase combinations that are compatible out of a possible 48 $(2 \cdot 4!)$.

R1	1	2	3	4
R2	5	6	7	8

Fig. 2. The eight-phase ring diagram for the intersection under study.

Normally, the cycle is executed in an end-to-end fashion with every phase receiving some interval time. Perhaps the only deviation would be if detectors are used to skip phases when there are no vehicles present for that movement. In our method, the phases have no particular order, and are actuated based on the queue sizes alone. This does not conflict with the simulation environment since we have the benefit of perfect data; however, modifications to the general scheme are required when applied to real-world systems, where factors such as hybrid traffic flows and side friction must be considered.

For our purposes, we assume that certain vehicle information is available to us. Working off of the assumption that each vehicle in the network is running an in-vehicle information system (IVIS) that is capable of communicating with the signal controller in some fashion, we are able to obtain vital information about each vehicle. At the most basic level, we assume knowledge of the vehicle's position in the network. An IVIS-equipped vehicle with a GPS (Global Positioning System) module could easily provide this information in near real-time. This is the only piece of information about the vehicle that we use for the control algorithm. Other information may include the vehicle's speed, its intended route, or other characteristics. Instead of counting vehicles with a complicated set of detectors, we simply ask the vehicles for their position and build the queues based on this information. While simplifying the physical setup of the intersection, the simulation is slowed down by having to request information from each vehicle in the network. It is, of course, entirely possible to use detectors to perform the queue counting function without loss of performance.

IV. Signal Scheduling Algorithm

A. Longest Queue Maximal Weight Matching (LQ-MWM) Signal Arbitration

We next describe the proposed signal arbitration algorithm. First, let us define the traffic load matrix as a doubly substochastic matrix, $\Lambda = \|\lambda_{ij}\|$ with admissible arrival rates, such that

From \ To	A	B	C	D
A	0	0	0	0
B	0	0	0	0
C	0	1	0	0
D	1	0	0	0

From \ To	A	B	C	D
A	0	0	1	0
B	0	0	0	1
C	0	0	0	0
D	0	0	0	0

From \ To	A	B	C	D
A	0	1	0	1
B	1	0	1	0
C	0	0	0	0
D	0	0	0	0

From \ To	A	B	C	D
A	0	0	0	0
B	0	0	0	0
C	1	0	0	1
D	0	1	1	0

Fig. 3. Allowable intersection configurations considered by the algorithm.

$$\sum_{l=1}^{N} \lambda_{il} < C, \sum_{l=1}^{N} \lambda_{lj} < C, \tag{1}$$

where λ_{ij} denotes the average rate of vehicles moving through the intersection from input link i destined for output link j, C the physical capacity of the links, and N the number of links that are connected at the intersection node. The first part of (1) states that no link has more than its capacity in traffic traversing it. The second part guarantees that overloading any of the destination links will not occur.

Let $Q(t) = [Q_{11}(t), ..., Q_{1N}(t), ..., Q_{NN}(t)]^T$ be the queue occupancy vector in which each component represents the number of vehicles currently queued at time t. For links that are associated with two destinations (e.g. link 6), we assume an equal queue size distribution between the flows destined to each of the two output links. Queues are served in accordance with the policy dictated by the signal control algorithm. Due to the nature of the traffic flow, all $Q_{ii}(t) = 0 \, \forall i$ (it is assumed that there is no loopback traffic). The signal control algorithm selects a set of compatible matches between a set of input and output links. The set of matchings is represented by a *matching matrix*, $\|S_{ij}(t)\|$, $1 \leq i \leq N$, $1 \leq j \leq N$, whose binary elements $S_{ij}(t) = 1 \; iff$ input link i is selected by the control algorithm to connect to output link j, otherwise $S_{ij}(t) = 0$.

There are four intersection matching matrices considered by the algorithm, as shown in Fig. 3. Letting the *weight* of a matching be denoted by $W(t) = \langle Q(t), S_{ij}(t) \rangle$, it is noted that given the four configurations of the intersection (i.e. matching matrices) described in Fig. 3, there are four corresponding weights, which we label $W_i(t)$ $i = 1, 2, 3, 4$. We further define κ_i $(i = 1, 2, 3, 4)$ as the *sum* of weights corresponding to every combination of three weights $(W_i(t))$. The indices of the such combination of weights are $\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{2, 3, 4\}$. The algorithm selects the matching matrix which has the highest value within the set of weights corresponding to the largest element in κ_i. It is noted that the algorithm requires the calculation of the weights and κ_i to take place prior to each configuration of the intersection. However, the computational complexity involved in such arithmetic is rather low.

One observation that may be pointed out is that, inherently, the algorithm tends to select links with larger queues. However, it is not necessarily the case that the link corresponding to the largest queue will be selected. This happens if and only if that link resides within the set of three intersection configurations that has the highest aggregate weight.

B. Stability of the Algorithm

In this section we provide a comprehensive stability proof for the proposed algorithm. At its core, the stability implies that the expected value of the queue sizes are all bounded. Another way of expressing this notion is to say that given the proposed signal control algorithm, there will never be an infinite backlog of traffic accumulating in any of the queues. Let us further define $P_k \subset \mathbb{R}^{N \times N}$ as the set of $(N!)$ permutation matrices of an $N \times N$ matrix, i.e. matrices with only a single 1 in each row and in each column. According to Birkhoff's theorem [6], the following inequality holds:

$$\Lambda < \sum_{j} \alpha_j P_j, \tag{2}$$

where $\sum_j \alpha_j = C$. Equation (2) states that any doubly sub-stochastic matrix can be decomposed into a convex sum of permutation matrices. Let $D(t) = [D_{11}(t), ..., D_{1N}(t), ..., D_{NN}(t)]^T$ be a vector denoting the departure process, for which the element $D_{ij}(t)$ represents the number of vehicles departed from link i for link j during time slot t. Hence, the evolution of the queue occupancy can be expressed as

$$Q(t+1) = Q(t) + A(t) - D(t), \tag{3}$$

where $A(t)$ is the number of vehicles arriving to the queue at time t. The intersection under study will be modeled by discrete time queues that, in turn, will be analyzed using Discrete Time Markov Chain (DTMC) models.

Definition 1: The weight produced by the LQ-MWM algorithm at time t is given by

$$W'(t) = \langle Q(t), S'_{ij}(t) \rangle = \sum_{i,j} Q_{ij}(t) S'_{ij}(t), \tag{4}$$

where $S'_{ij}(t)$ denotes the matching configurations established by the algorithm at time t. We next provide stability-related definitions which will aid in establishing the stability properties of the algorithm.

Theorem 2: (*Variation of Foster's Criterion* [7]): Given a system of queues whose evolution is described by a DTMC with state vector $Q(t) \in \mathbb{N}^M$, if there exist $\epsilon \in \mathbb{R}^+$ and $B \in \mathbb{R}^+$ such that given the function $L(Q(t)) = Q(t)Q^T(t)$, $||Q(t)|| > B$, the following holds:

$$E[Q(t+1)Q^T(t+1) - Q(t)Q^T(t) \mid Q(t)] < -\epsilon||Q(t)|| \quad (5)$$

then the system of queues is strongly stable.

The LQ-MWM signal control algorithm determines the configuration of the signals in the intersection once every k time slot units, which defines the switching interval. The latter loosely refers to the number of vehicles that can arrive or depart and would typically be on the order of a few seconds. We next present the core theorem of this paper.

Theorem 3: An intersection running the LQ-MWM signal control algorithm with aggregate traffic load destined to any output link that is less than C/3 is stable for any finite switching interval.

Proof: Since at most k vehicles may arrive during k time slots, the following inequality holds:

$$Q_{ij}(t+k-1) - Q_{ij}(t) \leq k, \quad (6)$$

from which we can write

$$Q_{ij}(t+k-1) - Q_{ij}(t) \leq \sum_{m=0}^{k-1} A_{ij}(t+m) - kS_{ij}(t), \quad (7)$$

for $Q_{ij}(t) \geq \eta k$. The term $kS_{ij}(t)$ expresses the k consecutive vehicle traversals that may occur during a switching interval. Next, we construct a discrete-time quadratic Lyapunov function, $L(t)$, defined as $L(t) = \langle Q_t, Q_t \rangle = \sum_{i,j} Q_{ij}^2(t)$. In order to prove the algorithm yields a stable queueing system, we would like to show that beyond a given threshold of maximum weight there is a negative drift in the state (queue occupancies) of the system. As an expression of a k time slot lag, we can write

$$L(t+k-1) - L(t) = \quad (8)$$
$$\sum_{ij} \left(Q_{ij}(t+k-1) - Q_{ij}(t)\right)\left(Q_{ij}(t+k-1) + Q_{ij}(t)\right).$$

For the case of $Q_{ij}(t) \geq k$, we deduct the following

$$\begin{aligned} &E\left[L(t+k-1) - L(t) | Q(t)\right] \leq \\ &\leq \sum_{ij} \left(\sum_{m=0}^{k-1} A_{ij}(t+m) - kS_{ij}(t) \right) E\left[2Q_{ij}(t) + k\right] \\ &\leq \sum_{ij} 2E\left[Q_{ij}(t)\right]\left(k\lambda_{ij} - kS_{ij}(t)\right) + \sum_{ij} k^2 \\ &\leq 2k\left[\langle \Lambda, Q_t \rangle - \langle S, Q_t \rangle\right] + k^2N^2. \end{aligned}$$

Using (2) we know that

$$\begin{aligned} \langle \Lambda, Q_t \rangle &= \left\langle \sum_j \alpha_j P_j, Q_t \right\rangle = \sum_j \alpha_j \langle P_j, Q_t \rangle \\ &< \sum_j \alpha_j \max_k \langle P_k, Q_t \rangle. \end{aligned}$$

given that $\sum_j \alpha_j = 1$, we obtain $\langle \Lambda, Q_t \rangle < \max_k \langle P_k, Q_t \rangle = \langle S^*, Q_t \rangle = W^*(t)$, which would conclude the proof if all permutation matrices were applicable to the intersection. To evaluate the impact of the partial connectivity that may be applied to the intersection, we note that any permutation matrix can be majorized (or covered) by at most three of the allowable intersection configurations. In other words, there exist l, m and n such that

$$\max_k \langle P_k, Q_t \rangle < \langle R_l, Q_t \rangle + \langle R_m, Q_t \rangle + \langle R_n, Q_t \rangle$$

for some $l \neq m \neq n \subset \Psi$, where Ψ is the set of allowable intersection configurations. Since

$$\langle R_l, Q_t \rangle + \langle R_m, Q_t \rangle + \langle R_n, Q_t \rangle < 3 \max_j \langle R_j, Q_t \rangle,$$

we conclude that $\left\langle \frac{\Lambda}{3}, Q_t \right\rangle < \max_j \langle R_j, Q_t \rangle$. ∎

This result states that if the *average* aggregate traffic heading to any given output link from all associated input links does not exceed $C/3$ (i.e. a third of the maximal physical capacity of the lane), the algorithm will always yield a stable system. Instantaneously exceeding the capacity is acceptable, so long as the average rate is bounded by $C/3$. This is irrespective of the distribution of traffic across the different input links.

V. Simulation Results

Using the intersection and simulation environment described in section II, the LQ-MWM algorithm and the fixed time controller are compared. Parameters of the simulation are varied to obtain comparisons between control methods for different traffic patterns and loads. For the control method comparisons, the traffic load is varied up to half of the maximum capacity of the inbound links. Traffic loads above this amount result in total gridlock, hence simulation outcomes are not considered valid. As a general rule-of-thumb, the maximum capacity of any one lane is taken to be 1800 vehicles per hour, or one vehicle every two seconds. The desired speed of the vehicles in the network is 60 kilometers per hour. Each simulation run is 40 minutes long to allow for the intersection to reach a steady state.

In general, when establishing timing parameters, the types of vehicles under consideration must be taken into account. Large trucks move more slowly than cars, and turning vehicles move more slowly than vehicles going straight through the intersection. In the case of vehicles turning left, the vehicle may even block the lane for some period of time while waiting for a chance to turn. However, for our intersection, a yield condition never exists; all lanes, including the left turn lanes, are only given a green light when they have the absolute right-of-way. Moreover, we have made passenger cars the only type of vehicle in our network, in an effort to provide consistent results.

The VISSIM traffic simulator allows for the initialization of a random seed for each simulation run. To help ensure that the data collected is well distributed, each data point is calculated as the average of three separate simulated runs, each having a different random seed. The two methods are compared using

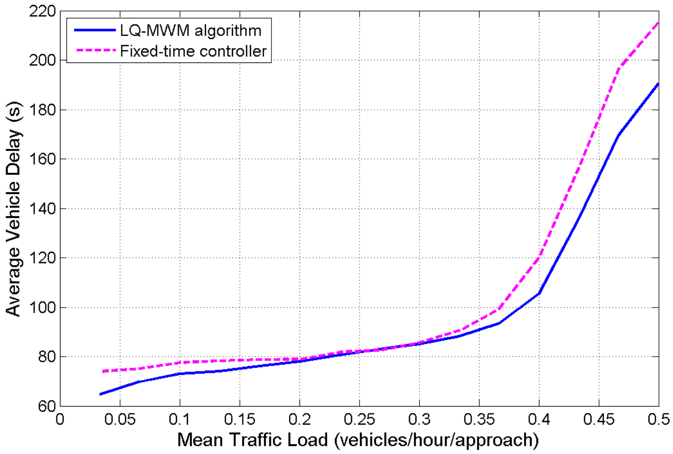

Fig. 4. Average vehicle delay for uniform traffic distribution.

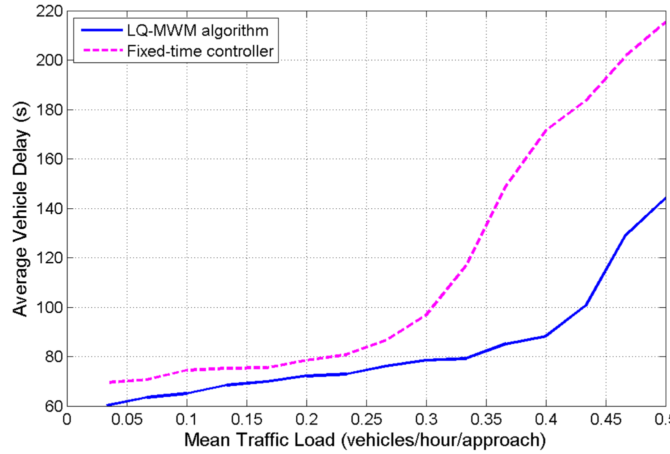

Fig. 5. Results of the vehicle delay comparison for nonuniform traffic flow.

the same set of random seeds, which provides identical traffic conditions to each technique, giving as direct a comparison as possible. Two sets of simulations are run: one with uniform traffic flows and one with nonuniform traffic flow distribution.

A. Uniform Traffic Flow

The first comparison between the two methods is made with uniform traffic flows; that is, all inbound traffic has an equal probability of turning left, going straight, or turning right. This automatically satisfies the load requirement stating that no outbound link is oversubscribed. This configuration also provides the fixed time controller with the best-case scenario for traffic routing. For this reason, the performance of the two methods is very similar. The results of the first comparison are illustrated in Fig 4. As can be noted from the latter, the LQ-MWM algorithm yields lower average delay when compared to the fixed time controller, in particular for loads that are distant from the 600 vehicles per hour point (0.33 mean traffic load) around which it was optimized.

B. Non-uniform Traffic Flow

The second set of simulations covers a nonuniform traffic distribution. In this case, the inbound traffic is split unevenly between the actions of turning left, going straight, and turning right. The distribution is such that the average load for any outbound link does not exceed the maximum physical capacity. The conditions of this simulation are different in that the fixed time controller is no longer giving appropriately adjusted intervals to the inbound traffic.

The results of this comparison are shown in Fig. 5. The LQ-MWM algorithm outperforms the fixed time controller over the entire range of traffic load. The difference between the two control methods is more pronounced here than in the uniform case. This difference would be less evident if the LQ-MWM algorithm was compared to a vehicle-actuated controller.

By comparing the two result figures, one can see that the LQ-MWM algorithm performed significantly better with the nonuniform traffic flows than it did with the uniform traffic flows. This result should be expected due to the adaptive nature of the LQ-MWM algorithm, which considers the state of the queues as part of the decision making process. The fixed time controller blindly gives time to phases with few or no vehicles, while our algorithm adapts to the change in the traffic distribution.

VI. CONCLUSIONS

We have presented a novel approach to controlling an intersection using techniques and ideas adapted from the field of computer networking. A stable arbitration algorithm was developed and compared to an existing control technique. Rigorous stability proof has been provided, suggesting that for aggregate traffic flows that do not exceed a third of the physical capacity, the algorithm is guaranteed to be stable. Moreover, a Matlab-VISSIM interface was described as a powerful platform for research on traffic management. The formal framework introduced in this paper is novel and generic, such that it has the potential to be applied to much more intricate traffic networks and flow management tasks.

REFERENCES

[1] E. A. Mueller, “Aspects of the history of traffic signals,” *IEEE Transactions on Vehicular Technology*, vol. 19, pp. 6–17, Feb. 1970.
[2] B. PARK and M. Chang, “Realizing benefits of adaptive signal control at an isolated intersection,” 2002.
[3] S. G. Shelby, “Single-intersection evaluation of real-time adaptive traffic signal control algorithms,” 2004.
[4] X.-H. Yu and A. R. Stubberud, “Markovian decision control for traffic signal systems,” in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, (San Diego, CA, USA), pp. 4782–4787, Dec. 1997.
[5] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.
[6] G. Birkhoff, “Three observations on linear algebra,” in *Univ. Nac. Tucum’n. Rev. Ser. A*, vol. 5, pp. 147–151, 1946.
[7] F. G. Foster, “On the stochastic matrices associated with certain queueing processes,” in *Ann. Math Statist.*, vol. 24, pp. 355–360, 1953.