University of Tennessee, Knoxville
Trace: Tennessee Research and Creative Exchange

Masters Theses | Graduate School

12-2007

A Longest-Queue-First Signal Scheduling Algorithm with Quality of Service Provisioning for an Isolated Intersection

Richard James Wunderlich
University of Tennessee - Knoxville

Recommended Citation

Wunderlich, Richard James, "A Longest-Queue-First Signal Scheduling Algorithm with Quality of Service Provisioning for an Isolated Intersection. " Master's Thesis, University of Tennessee, 2007.
https://trace.tennessee.edu/utk_gradthes/208

This Thesis is brought to you for free and open access by the Graduate School at Trace: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Masters Theses by an authorized administrator of Trace: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.

To the Graduate Council:

I am submitting herewith a thesis written by Richard James Wunderlich entitled "A Longest-Queue-First Signal Scheduling Algorithm with Quality of Service Provisioning for an Isolated Intersection." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Itamar Elhanany, Major Professor

We have read this thesis and recommend its acceptance:

Tom Urbanik, Ethan Farquhar

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:
I am submitting herewith a thesis written by Richard James Wunderlich entitled "A Longest-Queue-First Signal Scheduling Algorithm with Quality of Service Provisioning for an Isolated Intersection." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Itamar Elhanany
Itamar Elhanany, Major Professor

We have read this thesis and
recommend its acceptance:

Tom Urbanik
Thomas Urbanik II

Ethan Farquhar
Ethan Farquhar

Acceptance for the Council:

Carolyn R. Hodges
Carolyn R. Hodges, Vice Provost and
Dean of the Graduate School

(Original signatures are on file with official student records.)

A Longest-Queue-First Signal Scheduling Algorithm with Quality of Service Provisioning for an Isolated Intersection

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Richard James Wunderlich
December 2007

Copyright © 2007 by Richard James Wunderlich
All rights reserved.

Dedication

To my grandparents James and Mildred Almeida, who are no longer with us, thank you for your love and support. This thesis is dedicated to you.

Acknowledgments

There are several people who deserve my acknowledgment as I complete my graduate studies, and I take this opportunity to thank them.

First, I thank my advisor Dr. Itamar Elhanany, whose support and direction have allowed me to reach this point. Of course the support of Dr. Tom Urbanik has also been invaluable over the past year.

I also thank all of my teachers, professors, and student colleagues throughout my high school and college education who have prepared the way for my education and helped me all along the way. In particular, many thanks go out to Mrs. Lampkin, Mr. Margetson, Coach Wright, David DeRisi, Ben Elton, Sam Caylor, Josh Combs, and of course the brothers Wolf. From the Machine Intelligence Laboratory, my thanks go out to Zhenzhen Liu, Derek Rose, Brad Matthews, and Cuibi Liu for putting up with me.

Last but certainly not least, I would like to thank my family. Without your enduring support, none of this would ever have been possible. Thanks, Mom and Dad.

Abstract

In today's fast-paced society, the need to travel using automobiles is increasingly important. Aside from the road itself, the intersection is the most basic unit of a traffic system. As such, controlling the flow of traffic through intersections in an efficient manner has become a task of the utmost importance. The signal-scheduling algorithm described in this thesis is designed for just such a task. Concepts are drawn from the field of packet switching in computer networks and are applied to the traffic control problem. The method proposed utilizes a maximal weight matching algorithm to minimize the queue sizes at each approach to the intersection. The goal is to provide lower average vehicle delay as compared to a current state-of-the-art traffic signal control method. In particular, a focus is given to providing increased levels of service to high-priority vehicle classes (such as emergency vehicles or large trucks). Because the minimization of vehicle queues forms the basis of the algorithm, it is important to establish the conditions under which the system is guaranteed to be stable (i.e. the queue sizes are finite); to this end, Lyapunov function-based analysis is provided. Using a traffic simulation environment, the proposed control method is compared to control methods currently implemented in the field. The results of the simulations show that the performance gain obtained when using the proposed method can be substantial, particularly in the case where prioritization among multiple classes of vehicles is desired.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

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. Electric signals soon replaced mechanical signals, but were still either manually controlled or ran on a static cycle. The latter was the case for many decades until computers became common enough (and sufficiently inexpensive) to use in the signal controllers.

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 number of vehicles moving through the controlled intersections in the shortest amount of time while maintaining driver safety. Modern control methods are based either on a macroscopic view of the network or on an individual intersection view; some incorporate features of both, but most do not.

1.2 Motivation

While there is an extensive body of work concerning the design and optimization of large-scale traffic networks (on the scale of neighborhoods or even entire cities), they are all predicated on the fact that classic ring-and-phase signal controllers are used to control the individual intersections. These controllers are nearly mechanical in their operation, and provide little in the way of "intelligent" traffic control. Many of the optimization techniques rely on modifying the timing offset between the control cycles of adjacent intersections. Some adjust the timings of the individual intersections, but the basic cycle structure is still strictly adhered to.

In order to accommodate the increasing complexity of traffic flows, a fundamental change must be introduced into the most basic unit of the traffic network, the intersection controller. Once this change has been introduced, the focus may be shifted to higher levels of optimization in order to further increase the performance of the system. Arguably the most outdated feature of modern traffic controllers is the tendency to follow the predefined phase cycle. One advantage of this cyclic behavior is the inherent fairness of the system: a vehicle waiting at an approach to an intersection will definitely be served before a vehicle coming to an approach that is currently being halted. However, if this cyclic behavior could be left out without sacrificing performance or fairness – at least not too great a sacrifice – while allowing for the (live) prioritization of traffic flows, certainly the situation would be improved.

1.3 Prior Work

Several common adaptive traffic control systems have been tested and deployed in recent years. Some of the most widely used include SCATS Adaptive Traffic Control System, SCOOT (Split Cycle and Offset Optimization Technique), and in the United States, the FHWA commissioned the development of several adaptive control strategies, such as the RHODES (Regional Hierarchical Optimized Distributed Effective System), VFC-OPAC (Virtual Fixed Cycle-Optimization Policies for Adaptive Control) and RTACL (Real Time Traffic Adaptive Control Logic). The latter are implemented on the RT-TRACS (Real-Time

Traffic Adaptive Control System) platform [2].

SCATS Adaptive Traffic Control System is installed in many cities worldwide with the objective to minimize stops, delay, and travel time [3]. The system architecture contains local traffic controllers, regional computers, the management computer, the simplest configuration, and the operator interface. In this system, a maximum of degree of saturation is maintained by selecting optimal cycle length and phase split times. Offset plans are selected by comparing the amount of traffic flows on the links.

SCOOT, also regarded as an effective traffic adaptive control system, consists of a simulation model and an optimization model [4]. The system architecture contains second-by-second systems with timing algorithms in a central processor, a local controller which deals with clearance and minimums, local vehicle actuation determined by traffic engineering priorities, and a hierarchical transmission system with flexibility to suit local traffic control needs.

The adaptive control strategy RHODES, taking advantage of natural stochastic variations in traffic flow, is designed to minimize traffic delay [5]. The RHODES architecture can be decomposed into three hierarchical levels, including intersection control, network control, and network loading. At the intersection control level, traffic flow predictions and signal phase and duration decisions are made based on the real-time traffic condition on a second-by-second basis. At the network control level, predictions are made periodically to establish coordination constraints for each intersection in the network [6]. The general travel demand over longer periods of time is predicted on the network loading level. The software has been developed to interface with the latest version of the CORSIM traffic simulation model.

Another traffic adaptive control strategy, VFC-OPAC, consists of three layers: the local intersection control Layer, which calculates optimal switching sequences for the projection horizon; the coordination layer, which optimizes the offsets at each intersection in real-time; and the network synchronization layer, which calculates the network-wide virtual fixed cycle [7]. Cycle length determination is made at the central controller and communicated periodically to the intersection controllers.

RTACL is a distributed strategy which uses a macroscopic simulator to estimate traffic

flow and evaluates signal-phasing alternatives [8]. Based on the queues of all links, each local controller optimizes its own timings and decides signal timings for two cycle lengths. These signal timings contain short-term recommendations for the current phase length and the next, and also give temporary recommendations for the future phases. The network model and local controllers of the nearby intersections can then use these recommendations to predict traffic flows.

There are several computational intelligence-based techniques that have been applied for the designing of real-time traffic signal controllers, such as fuzzy logic system (FLS), neural networks (NN) and genetic algorithms [9]. Along those lines, an intelligent isolated intersection control system was recently proposed by applying a two-step process that develops the rules of fuzzy control [10].

Some traffic control simulations focus on isolated intersections. The MOVA (Microprocessor Optimized Vehicle Actuation) system is based on stage control and is a self-optimizing system designed to reduce delays and stops and to maximize capacity during peak periods [11][12]. The LHOVRA system is based on signal group control and refinement of traditional interval measuring techniques [11][13].

VS-PLUS, a system in use today in Europe and in the United States, is a standardized system for traffic-actuated controllers that can synchronize controllers and assign priority to vehicles, especially for public transportation. This system is one of several that are included in a movement sponsored by the United States Department of Transportation called Transit Signal Priority (TSP) [14]. TSP is an operational strategy that seeks to improve the throughput of service vehicles through signal-controlled intersections. TSP systems are currently deployed in many cities and allow for the recognition of priority vehicles through optical detection, inductive loop-based detection, or even GPS-based tracking. The systems employ strategies involving phase modifications such as early green, green extension, phase insertion, and preemption. These systems act on the level of individual intersection control.

1.4 Current Work

The difference between current adaptive control schemes and our work is that most other systems simulate multiple-intersection traffic networks, while our work is focused on the analysis of an isolated intersection. Moreover, other techniques are mostly based on optimizing the coordination of multiple intersections, and they are not concerned with the individual light cycle changes at each intersection (with the exception of RTACL and VS-PLUS). Our approach seeks to improve the performance of each intersection individually. For research purposes, this is a very valuable approach; it allows us to concentrate on developing an effective arbitration policy without the large overhead of intersection coordination. This provides a solid base upon which to build future work concerning multiple-intersection traffic networks. In addition, the prioritization of vehicles allows for the controller to adjust its behavior based on the particular vehicles that are currently approaching the intersection.

In recent work, the problem of scheduling traffic at an intersection has been addressed by structuring the problem as a Markov decision process (MDP) [15]. 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 [16]. 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 thesis, LQF-MWM is presented as an algorithm used for signal control at an isolated intersection. Its goal is 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 using a traffic simulation platform that allows for testing under various vehicular traffic patterns.

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 then choose the cycle most appropriate for the current traffic conditions. In

addition, many controllers have the ability to modify the light cycle depending on detection of vehicles, the time of day, as well as other factors. However, 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 the traffic flows in order to facilitate the light cycle testing and verification process.

The rest of the paper is structured as follows: Chapter 2 includes several sections pertaining to the technical aspects of the current work. Section 2.1 provides a description of the intersection model used along with a discussion of the signal cycle attributes and constraints. Section 2.2 describes the proposed algorithm, and its stability properties are then obtained. Section 2.3 details the signal controller's interface with the traffic simulator. In Chapter 3 we present simulation results, and in Chapter 4 the conclusions are drawn and directions for future work are outlined.

Chapter 2

Technical Approach

2.1 System Model

2.1.1 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 employed by the simulations, and to define some performance metrics upon which to base qualitative comparisons between the control methods.

Figure 2-1 depicts a four-way intersection with through lanes and separate left-turn lanes. Each of these lanes is called a movement, or an approach. Any of these movements that are to be permitted simultaneously are grouped into phases. For example, the straight and the right-turn movements are grouped into one phase because, in this intersection, they occupy the same physical lane. The phases of this intersection are numbered according to the standard NEMA (National Electrical Manufacturers Association) convention.

Intuitively it can be noted that not all of the 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 an identical indication during an interval, are referred to as a signal group. A signal controller is thus responsible for controlling all of the signal groups at the intersection.

The signal controller is programmed with information about the intersection and its

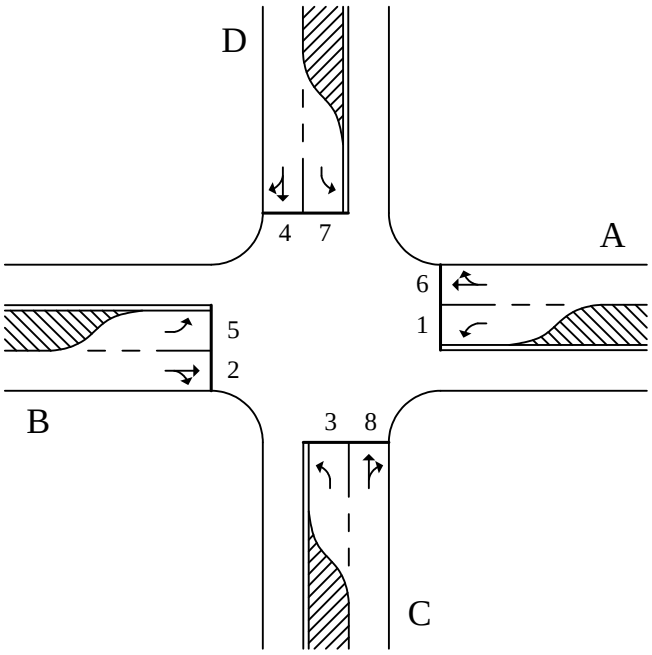

Figure 2-1: Intersection model with standard movement numbering.

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. For our test intersection, the yellow time and clear time of the intersection comprise a five second idle time between phase changes.

In the stability discussion of Section 2.2.3 below, for the purpose of clarity, the lanes leading to the intersection are referred to as links, and the intersection is 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 described in terms of

vehicles per hour.

2.1.2 Intersection Configuration

The intersection under consideration is illustrated in Figure 2-1. The labeling of the signal groups in this intersection follows the NEMA convention. This intersection is an adequate 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 2000 meters from its endpoint to the intersection. The left turn lanes provide 200 meters of vehicle queue space to help avoid blocking at the lane branching point. All vehicles within 100 meters of the signal are counted as being in the queue for that signal. The long leads into the intersection help to ensure that arriving traffic is distributed properly, and that vehicles do not build up at the inputs of the network. This is particularly important for simulation runs with traffic levels approaching the saturation level.

2.1.3 Performance Metrics

Fundamental metrics for evaluating the performance of a traffic controller (particularly at an isolated intersection) include: vehicle delay, traffic throughput, vehicle stops, and average queue size. Analyzing the overall delay experienced by a vehicle as it exits the network is a direct indication of how long the vehicle has had to wait at the intersection prior to traversing it. This is true in our case due to the inherent symmetry of the network under consideration. At the desired speed, it takes a certain amount of time to pass through the network. Subtracting this time from the total time that the vehicle has been in the network reveals the delay associated with the intersection itself. Throughput measures the number of vehicles per hour that pass through an intersection, and is also indicative of the overall controller performance as it is directly related to the vehicle delay.

Vehicle stops refers to the number of times that a vehicle must come to a stop while attempting to traverse a traffic network. In general, a vehicle is counted as having been stopped only once for each intersection. Naturally, the minimization of stops is a primary concern of traffic network optimization techniques.

Figure 2-2: The eight-phase ring diagram for the intersection considered.

The queue sizes are the most important metric that we study in this work. As expressed by Little's theorem, the queue size is directly proportional to the delay experienced by the vehicles in the network. Thus, in order to minimize the average vehicle delays, one should seek to minimize the average queue sizes. This is exactly the approach taken by our control method.

Although minimizing the queue sizes is a primary motivation of the LQF-MWM algorithm, the overall vehicle delay is the metric that we use to compare the performance of the different control methods. Given the symmetry of the network, all vehicles should experience the same amount of delay as they traverse it. That is, if every vehicle were to travel through the network without stopping, they would all have identical mean delay time, regardless of the path taken. For this reason, we view the results in terms of the average vehicle delay.

2.1.4 Traffic Cycle Attributes and Constraints

In order for the intersection to operate properly, and in an effort to ensure the safety of the vehicles approaching the intersection, the traffic light cycle must be valid. Commonly, the cycle consists of phases placed in a particular order, each interval of which is given some amount of the total cycle time. The ring diagram for our intersection is depicted in Figure 2-2. In this diagram, time progresses from left to right, indicating that the left-turn phases become active before the corresponding straight/right phases (referencing the movement numbers of Figure 2-1). A vertical barrier (double line) 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 barriers at the same time. Note that the separations of the phases between the barriers (e.g. the separation between phases 1 and 2), are based on the interval times assigned to each phase, and have been drawn arbitrarily in this diagram. Careful observation reveals that there are only eight unique phase combinations that are compatible out of all possible combinations.

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 vehicle detectors are used to skip phases when no vehicles are present for that movement. In our method, the phases have no particular order, and are actuated based only on the queue sizes. This does not conflict with the simulation environment since we have the benefit of perfect data; however, modifications to the general scheme may be required when applying it to real-life systems, where factors such as imperfect vehicle detection, hybrid traffic flows, and side friction must be considered.

For our purposes, we assume that certain vehicle information is always available to us. Working off of the assumption that every 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 telemetry information from each vehicle. At the most basic level, we assume knowledge of the position of the vehicle in the network. An IVIS-equipped vehicle with a GPS module could easily provide this information in near real-time.

The United States Department of Transportation (US DOT) and the Research and Innovative Technology Administration (RITA) are working towards making intelligent transportation systems a reality. In the future, traffic signal controllers will be provided information about the vehicles around them through Vehicle Infrastructure Integration (VII), an initiative toward the deployment of advanced vehicle-to-vehicle and vehicle-to-infrastructure communications. Our work assumes that such a VII system exists and allows us to gather information directly from the vehicles approaching the intersection.

The position and the type (priority) of the vehicle is the only information that we use for

From \ To	A	B	C	D
A		6	1	6
B	2		2	5
C	8	3		8
D	7	4	4	

Figure 2-3: Phase connection diagram for the test intersection.

our 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 in each time step based on this information. While this simplifies the physical setup of the intersection, the simulation is slowed down by having to request information from each vehicle in the network at every step of the simulation. It is, of course, entirely possible to use detectors to perform the queue counting function without loss of performance. However, the increased intersection complexity and the inability to determine the type of the vehicle make it impractical for our purposes. Using the position information to build the queues is thought to be a more generalized solution, because the method can be applied to any intersection without the need for a complicated detector setup.

2.2 Signal Scheduling Algorithm

2.2.1 Principles of Operation

Consider the phase connection diagram shown in Figure 2-3. This diagram indicates which phases are used to mobilize a vehicle through the intersection from any input to any output. Note that a vehicle cannot leave on the same link from which it arrived. This data is given to the signal scheduling algorithm, along with phase compatibility information, in order for it to evaluate the size and weight of each queue. These weights reflect the service urgency of each queue.

At a high level, the operation of the algorithm may be thought of in the following way.

Make a collection of all matrices of the same size as the phase connection matrix of Figure 2-3. These matrices have at least a single 1 and at most all of the 1's it contains cover only compatible phase combinations. Eliminating the redundancies creates the set of matrices shown in Figure 2-4. A queue value matrix is then created with the same form as the phase connection matrix. The entries of this matrix are the values of each queue placed in the position of its respective phase number (including any duplications). The link permutation matrices can then be sequentially multiplied by the queue value matrix on an entry-by-entry basis (not by true matrix multiplication). This effectively "masks" the queue value matrix with the permutation matrix, leaving only the values of the queues that will be selected. Summing the entries of these resultant matrices yields the value of each possible phase combination. By selecting the permutation that produces the highest value, the queue or queues with the highest vehicle priorities are served. This is, in essence, what the algorithm that we describe next does. It should be noted that if there is a tie in the queue values, it is broken randomly; however, this behavior could be modified to take into account additional information in the future.

2.2.2 Longest-Queue-First Maximal Weight Matching (LQF-MWM) Signal Arbitration

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

$$\sum_{l=1}^{N} \lambda_{il} < C, \sum_{l=1}^{N} \lambda_{lj} < C, \tag{2.1}$$

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 (2.1) 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

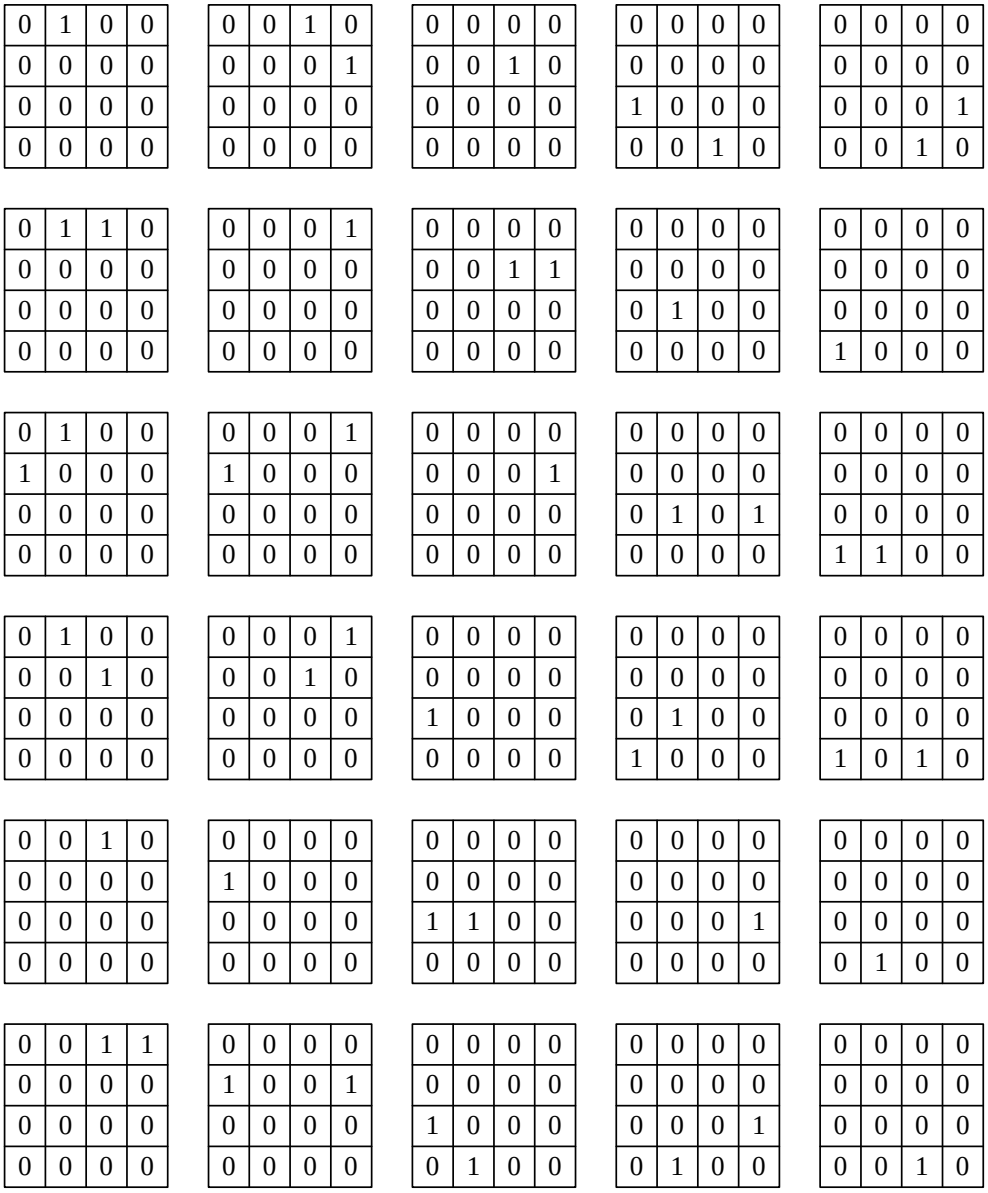

Figure 2-4: Link permutation matrices dervied from the phase connection matrix.

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 Figure 2-5. By comparing Figure 2-5 to Figure 2-3, and referencing back to the ring diagram of Figure 2-2, one can see that all possible phase combinations are "covered" by these matching matrices. In addition, these matching matrices cover all valid link permutation matrices; this is a necessary condition for the validity of the algorithm.

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 Figure 2-5, 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.

It should be noted 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 intersection configurations that has the highest aggregate weight. We further note that an increased measure of priority can be applied to a particular vehicle in the queue by simply giving it an increased individual value. That is, when the queues are being considered by the algorithm,

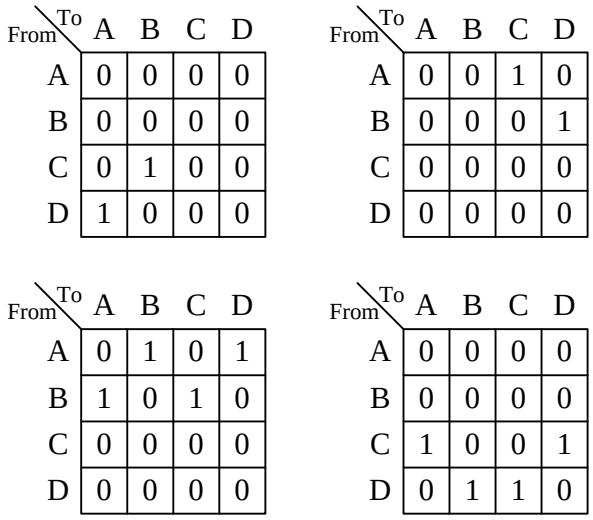

Figure 2-5: Intersection configuration sets considered by the algorithm.

the value of each vehicle can be independently chosen beforehand (based upon vehicle type, for example). By artificially inflating the values of the queues, we force the algorithm to service the queues containing the high-priority vehicles. Selecting values for the different vehicle classes then becomes a matter of balancing between the maximum queue sizes and the importance of the vehicles considered. For example, one may want a single emergency vehicle in one queue to outweigh another queue that is filled with lower-priority vehicles.

2.2.3 Stability of the Algorithm

In this section we provide a comprehensive stability proof for the proposed algorithm. At its core, 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 [17], the following inequality holds:

$$\Lambda < \sum_{j} \alpha_j P_j, \tag{2.2}$$

where $\sum_j \alpha_j = C$. Equation (2.2) 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

$$Q(t+1) = Q(t) + A(t) - D(t), \tag{2.3}$$

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 LQF-MWM algorithm at time t is given by*

$$W'(t) = \left\langle Q(t), S'_{ij}(t) \right\rangle = \sum_{i,j} Q_{ij}(t) S'_{ij}(t), \tag{2.4}$$

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 [18]): 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:*

$$E[Q(t+1)Q^T(t+1) - Q(t)Q^T(t) \mid Q(t)] < -\epsilon ||Q(t)|| \tag{2.5}$$

then the system of queues is strongly stable.

The LQF-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 LQF-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:

$$Q_{ij}(t+k-1) - Q_{ij}(t) \leq k, \tag{2.6}$$

from which we can write

$$Q_{ij}(t+k-1) - Q_{ij}(t) \leq \sum_{m=0}^{k-1} A_{ij}(t+m) - kS_{ij}(t), \tag{2.7}$$

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 [19], $L(t)$, 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

$$\begin{aligned} &L(t+k-1) - L(t) = \\ &\sum_{ij} \left(Q_{ij}(t+k-1) - Q_{ij}(t)\right) \left(Q_{ij}(t+k-1) + Q_{ij}(t)\right). \end{aligned} \tag{2.8}$$

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

$$\begin{aligned} & E\left[L(t+k-1) - L(t) | Q(t)\right] \leq \\ & \leq \sum_{ij} \left(\sum_{m=0}^{k-1} A_{ij}(t+m) - kS_{ij}(t) \right) E\left[2Q_{ij}(t) + k\right] \\ & \leq \sum_{ij} 2E\left[Q_{ij}(t)\right] \left(k\lambda_{ij} - kS_{ij}(t)\right) + \sum_{ij} k^2 \\ & \leq 2k\left[\langle \Lambda, Q_t \rangle - \langle S, Q_t \rangle\right] + k^2 N^2. \end{aligned} \tag{2.9}$$

Using (2.2) we know that

$$\begin{aligned} \langle \Lambda, Q_t \rangle &= \left\langle \sum_j \alpha_j P_j, Q_t \right\rangle = \sum_j \alpha_j \langle P_j, Q_t \rangle \\ &< \sum_j \alpha_j \max_k \langle P_k, Q_t \rangle . \end{aligned} \tag{2.10}$$

given that $\sum_j \alpha_j = 1$ (after normalizing the load matrix), 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

$$\max_k \langle P_k, Q_t \rangle < \langle R_l, Q_t \rangle + \langle R_m, Q_t \rangle + \langle R_n, Q_t \rangle \tag{2.11}$$

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

$$\langle R_l, Q_t \rangle + \langle R_m, Q_t \rangle + \langle R_n, Q_t \rangle < 3 \max_j \langle R_j, Q_t \rangle , \tag{2.12}$$

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. For the case of quality of service provisioning, the situation does not change, so long as the incoming and outgoing traffic loads remain admissible over time. Note that this stability proof holds for any finite-expectation switching interval, and that stability implies 100% throughput of the rated capacity ($C/3$).

2.3 Evaluation Environment

2.3.1 General Requirements

In order to fully understand the performance of our control algorithm, it would be helpful to compare it to a current control method. In the results section, we study the performance as compared to a fixed-time controller. This is hardly a fair comparison due to the static nature (and partial observability) of the fixed-time controller versus the proposed controller, since the latter relies on gathering data pertaining to approaching vehicles and making subsequent informed signal control decisions. We therefore need to compare our method against another controller that also utilizes vehicle information.

There are many vehicle-actuated traffic controllers currently in use. One such controller is the Econolite ASC/3, which is the latest in a series of Advanced System Controllers produced by Econolite. The ASC/3 is a NEMA TS1/TS2- and NTCIP-compatible traffic signal controller. The controller offers standard ring-and-cycle control of intersections with up to 16 phases in four rings with up to 64 detectors. The platform is generic such that it can be effectively utilized by almost any intersection that is physically possible to construct.

Up until recently, if one wanted to use such a controller in a traffic simulation, the actual hardware controller needed to be connected directly to the computer running the simulation to form an HIL (Hardware-In-the-Loop) simulation setup. At most, the HIL setup is only capable of running in real-time, due to the limitation of having the actual hardware control the simulated intersection. This issue is resolved by utilizing the Econolite ASC/3 SIL (Software-In-the-Loop) controller, which is fully integrated into the software simulation environment and executes the same control logic as the real-life hardware controller. The

greatest advantage of the SIL system is the ability to simulate faster than real-time. In addition, multiple intersections in the simulated traffic network can use the virtual controller without the cost or complexity of having the actual hardware controllers connected to the simulator.

2.3.2 Traffic Simulator

In order to test and compare control methods, we use the VISSIM traffic simulation environment from PTV America. VISSIM is a microscopic, multi-modal traffic simulator that gives the user control over all aspects of the network, such as vehicle type, driver behavior, intersection control, and statistical data collection.

The general operation of the simulator is as follows. The layout of the road links and connectors is first modeled in VISSIM, possibly even 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 controllers are defined for each intersection. Vehicle speed profiles and driver behaviors can also be specified in order to better mimic specific conditions under which the controller is expected to operate. Different vehicle classes exist, including cars, trucks, busses, heavy vehicles, trams, and trains. The simulator provides support for hybrid traffic flows that include bicycles, as well as pedestrian traffic models.

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, the virtual ASC/3 SIL module, and also an interface for an external hardware controller. It is even possible for each intersection in the network to have a different type of signal controller, if desired.

2.3.3 Signal Controller

Our control method relies on an algorithm, discussed in Section 2.2 above, 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. VISSIM provides a Component Object Model (COM) interface in order to give control of the simulator to external applications. Using this COM interface from Matlab, we are able to load the traffic

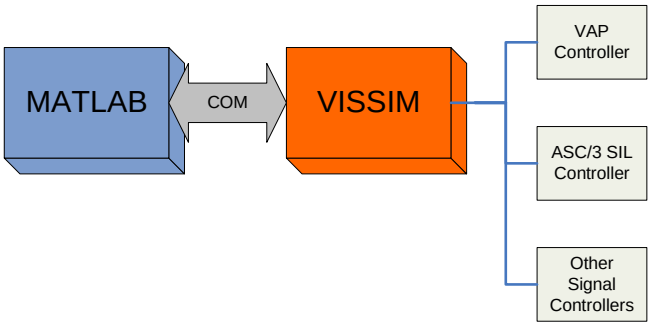

Figure 2-6: Schematic representation of the control interface flow.

network, set simulation parameters, execute simulations, and collect data. The control routine single-steps through the traffic simulation while controlling the signal group with the custom-designed control logic. A schematic representation of the control interface flow is shown in Figure 2-6. It is important to note that we have no direct control over the actual signal controllers, which interface directly with the simulator.

Due to a limitation (or perhaps a protective behavior) of the simulator, the COM interface allows for the reading of the state of a signal, but does not allow for the changing of the traffic signal states directly. That is, one cannot force a state change of a phase from red to green and apply this change directly through the COM interface. Thus, a more involved method is required to affect the state of the signals in the network.

In order to control the many phases of the intersection from Matlab, a simple VAP code controller monitors vehicle detectors in the network that are associated with each phase. In the simulator, the detectors are disabled to prevent vehicles from triggering them. Instead, when our controller determines that a phase should be changed, a trigger is sent from the Matlab environment via the COM interface to the vehicle detector in the simulator. This in turn prompts the VAP controller to change the state of the corresponding signal group.

This control method is used to both activate and deactivate the phases. The VAP controller simply swaps the state of the signal group whenever the corresponding detector is activated. The simulator automatically inserts the appropriate yellow time previously defined for the intersection. It is, however, up to the Matlab controller to allow for the cor-

responding intersection clear time. There are no safeguards provided against unsafe phase changes other than those attributed to the nature of the controller, which is fundamentally incapable of changing to a phase combination that is unsafe. Care is taken in the Matlab control framework to keep track of the state of the signals in order to avoid hazardous conditions. To date, no such condition has been observed.

During a simulation, the ASC/3 controller reads the state of its detectors ten times per second (once every simulation step), while the proposed controller only reads the detectors once every second. This is due to the fact that it steps the simulation forward by ten steps prior to invoking the control algorithm. This is because during each control step (once every second), the algorithm loops through a list of all of the vehicles in the network and requests information – like position, speed, and vehicle type – in order to build the vehicle queues and to determine if any vehicles have stopped. The reduction in simulation speed due to this vehicle information loop necessitates some amount of speedup in order to make running the simulations practical. Simulation results have shown that the effect of this reduction in resolution is negligible.

Chapter 3

Simulation Results

3.1 Control Method Comparison

Using the VISSIM simulation environment, the intersection described is tested under various traffic conditions for multiple control schemes. The primary variable considered is the average traffic load. A value of 1800 vehicles per hour (or one vehicle every two seconds) is taken to be the maximum traffic load for each of the four approaches to the intersection. Incoming traffic load is varied from near zero up to half of the maximum load ($C/2$). For each data point, an average is taken over three separate simulation runs in order to obtain sufficient statistics. Along those lines, each run simulates 40 minutes of traffic flow.

At first, the control schemes were evaluated using only cars in the network. This provided for some uniformity with respect to the behaviors of the vehicles in the network. However, a goal of this effort was to provide some increased measure of service to trucks. Therefore, truck traffic is added to the network in order to compare the per-class quality of service between cars and trucks in the network. The weighting scheme is such that a queue with a single truck will always outweigh another queue filled completely with cars. The exact value of the differenced depends on the maximum physical length of the queues.

Two types of traffic flows are studied: uniform traffic distribution, in which there is an equal probability for all vehicles to either turn left, turn right, or go straight through the intersection; and non-uniform traffic flow, in which there is a more realistic distribution of destinations. For the latter, the distribution used is 70 percent straight, 20 percent

right and 10 percent left. Since the distribution is the same for all approaches, the average outbound load never exceeds the maximum physical capacity. While this distribution is arbitrarily chosen, it represents, in general, a more pragmatic traffic pattern. As proven above, any traffic distribution that does not violate the admissibility criteria results in a stable operation.

Considered next is the truck traffic load allocated to each approach. Initially, five percent of the traffic is selected as trucks, and this load is appended uniformly to all approaches (and uses the same destination distribution as the cars). This results in rather uninteresting behavior, because all approaches end up having the same long-term average priority. That is, the resulting prioritized traffic flow experiences the same delay characteristics as the non-prioritized traffic flow.

In order to avoid this, and to more clearly identify the impact of prioritization, an imbalance is enforced such that the approaches receive 0, 5, 10, and 15 percent truck traffic clockwise from the North approach. This means that the North-South directions have 5 percent truck traffic, and the East-West directions have 10 percent truck traffic.

Figure 3-1 summarizes the phase count percentages, enabling an effective comparison of the different traffic composition scenarios. We observe medium and high traffic loads between three variations of truck distribution: no trucks present, uniform truck distribution, and non-uniform truck distribution. Note that the phase call percentages are fairly well matched between the first two cases. A closer look at Figure 3-1, reveals that indeed the disparity between phases 2 and 6 (eastbound and westbound) and phases 4 and 8 (southbound and northbound) has become more significant; 2 and 6 have increased while 4 and 8 have decreased, as would be expected.

In addition, differences between the average vehicle delay are also notable. Intuitively, the addition of trucks to the traffic flow should increase the average delay. Trucks are slower to change speed and, therefore, slow vehicles behind them. The delay characteristics of the three cases are shown in Figure 3-2. It can be seen that the addition of trucks does in fact increase the average delay. Moreover, the adjustment of the distribution adds delay as well. This is due in part to the overall increase in truck traffic from 5 to 7.5 percent of the total number of vehicles in the network. Certainly the most challenging aspect of any scheduling

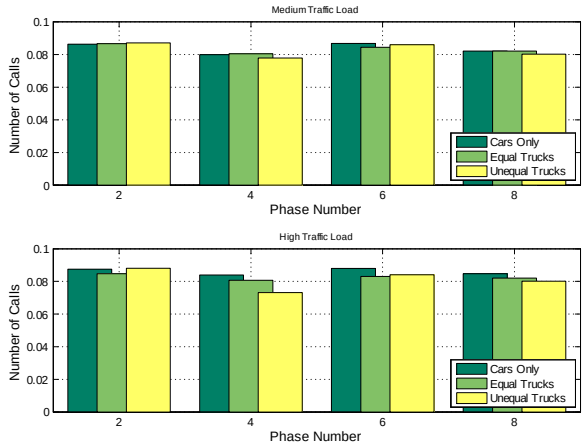

Figure 3-1: Phase selection counts for cars only, homogeneous truck distribution and heterogeneous truck distribution.

scheme would be to minimize the delay for this worst case component, as we address next.

3.2 Uniform Traffic Distribution

The first case under consideration is the intersection with uniform traffic flow. This means that each vehicle tends to make straight, left turn, and right turn decisions with equal probability. Such a setup facilitates the most basic form of traffic controller, the fixed-time controller, with a best-case scenario for traffic routing. Therefore, we compare our method, along with the ASC/3, to the fixed-time controller. Note that the fixed-time controller has been optimized for the $C/3$ traffic load point (i.e. 0.33 relative traffic load). Two variants of the LQF-MWM algorithm are compared to both the fixed-time controller and the ASC/3. One uses no priority, and the weight matrices are based on the queue sizes alone. The other utilizes priority, where the trucks are counted with a higher weight than the cars. This controller also has extensions enabled, whereby the phase interval is extended when there is a high-priority vehicle still in the queue when the interval first comes to an end. The number of extensions is limited to at most double the overall maximum green time for that phase interval.

The results of the delay analysis for this intersection configuration are shown in Figure

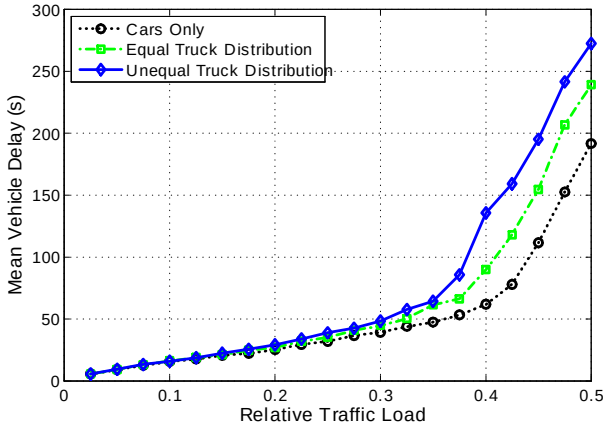

Figure 3-2: Vehicle delay comparison between three different traffic compositions.

3-3 and Figure 3-4. Figure 3-3 depicts a comparison of the mean car delays for the four control variants considered. All controllers exhibit similar behavior, and there is almost no difference between them through a wide range of traffic volumes. An exception, of course, is the fixed-time controller, which stands out as a poor performer at low volumes due to its ignorance of the presence of vehicles to be served, as it follows its strictly cyclic behavior.

Turning now to Figure 3-4, we observe a larger variation between the different control methods. This figure provides only the average delay of the trucks in the network. In general, the ASC/3 performs marginally better than the other controllers at very low traffic volumes. Again, the fixed-time controller is a poor performer at lower volumes, however, its performance closely follows that of the ASC/3 at medium and high loads for both cars and trucks. This is due to the fact that the ASC/3 inherently follows a cyclic pattern as it progresses through the phases; the only difference being the skipping of phases when no vehicles are detected and the ending of intervals when the detectors become deactivated.

The difference between the algorithms becomes much more apparent when examining the vehicle delay histogram. The latter for uniform traffic flow is shown in Figure 3-5. This shows a comparison between the ASC/3, shown on the top, and the LQF-MWM controller (with priority and extensions) on the bottom. There are two different load points illustrated: the graphs on the left are for $C/4$ (0.25 relative load), and the graphs on the right are for

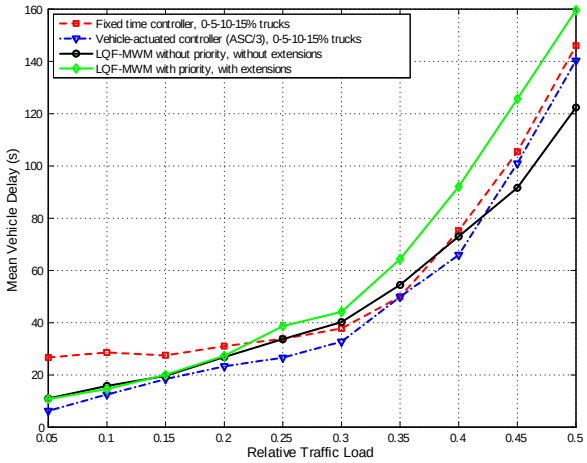

Figure 3-3: Average vehicle delay for uniform traffic comprising of cars only.

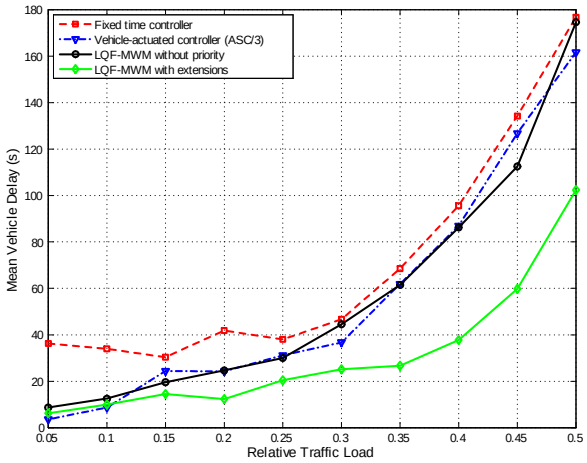

Figure 3-4: Average vehicle delay for uniform traffic comprising of trucks only.

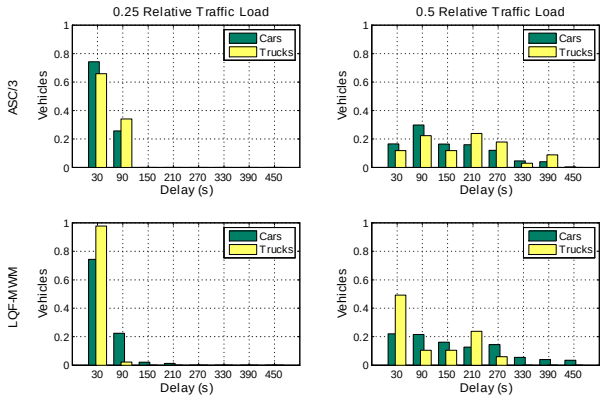

Figure 3-5: Average vehicle delay histogram for uniform traffic at medium and high traffic loads, for the ASC/3 and LQF controllers.

$C/2$ (0.5 relative load). For the lighter loading condition, it can be clearly seen that the ASC/3 delays both classes of vehicles equally. On the other hand, with the prioritized traffic flow of the LQF controller, we see that nearly all truck traffic is delayed less than a minute (the first bar group). The effect of priority on the truck delay is also clear from the histograms for $C/2$. At this load point, the ASC/3 again delays the traffic fairly evenly. If anything, a slight edge is given to the cars given the more aggressive speed profile that they exhibit. With the LQF controller, however, advantage is clearly given to the trucks, at the expense of delaying some cars significantly.

The last performance study pertains to vehicle stops. Recall that a vehicle is considered stopped if it ever comes to rest during its approach or passage through the intersection. Figure 3-6 shows the results of a comparison between the ASC/3 and the LQF algorithm for uniform traffic flow. In general, the percentage of vehicles stopped grows as the traffic load increases. As expected, the ASC/3 stops both classes of vehicles almost equally. The LQF algorithm, using priority and interval extensions, however, is able to maintain substantially lower stops for the trucks across all traffic loads. In fact, stops for both cars and trucks are decreased when using the LQF algorithm. At very low traffic volumes, however, the results are not entirely clear given the limited number of trucks that actually enter the network.

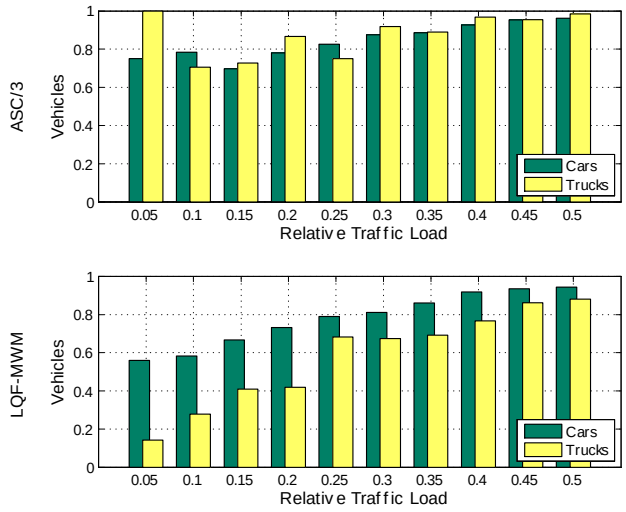

Figure 3-6: Vehicle stops for uniform traffic comparing the ASC/3 controller with LQF-MWM (using priority scheduling and extensions).

3.3 Non-uniform Traffic Distribution

The case for which the traffic routing is non-uniformly distributed is far more realistic, and therefore more interesting to study. Certainly, the conditions of this isolated intersection are far from reality, but the potential of the control frameworks may still be evaluated through these results. First, we examine the average vehicle delay for different control methods for the two classes of vehicles under consideration. We compare three variants of our approach to the ASC/3. The first uses no priority, and the weight matrices are based on the queue sizes alone. The second uses priority, but no interval extensions. The third uses both priority and extensions, enabling the latter when a high-priority vehicle is in the queue while the interval reaches its end. Again, the number of extensions is limited to at most double the overall maximum green time for any phase interval, thereby retaining the condition need for stability which is a finite expectation on the interval durations.

The car and truck delays are illustrated in Figure 3-7 and Figure 3-8, respectively. In Figure 3-7, we see that the performance of the ASC/3 and the LQF algorithm with no priority are quite comparable. Moreover, it should be noted that car delays for the two

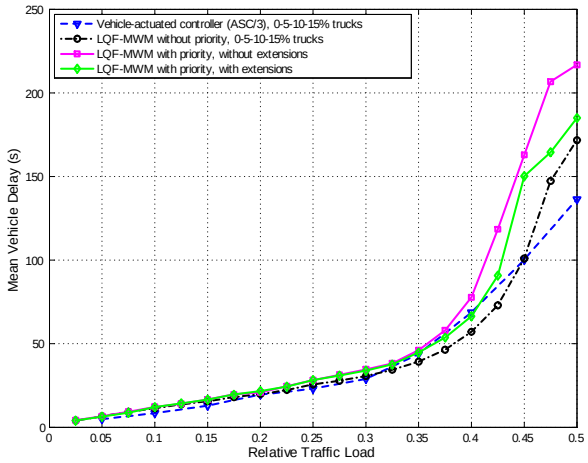

Figure 3-7: Avereage vehicle delay for non-uniform traffic distribution comprising of cars only.

LQF variants with priority are appreciably higher than the ASC/3 delay. This is due to the preference given to servicing the trucks. Referring to Figure 3-8, we observe that truck delay for the LQF algorithms are much lower for all medium and high traffic loads when compared to the ASC/3. Clearly, the addition of interval extensions does not affect the results substantially, but it does help both trucks and the cars that are around them to traverse the intersection more quickly than would be the case with the ASC/3.

Turning out attention to the delay histogram in Figure 3-9, one observes similar results as to those found in the uniform case. Again, at low volumes, nearly all truck traffic is ushered through the intersection with minimal delay for the LQF controller, while the ASC/3 delays more than 15 percent of the trucks for more than one minute. At the higher volumes, we see that the LQF controller is able to shift substantial numbers of trucks towards the lower end of the delay spectrum. Referring back to Figure 3-5, note that the performance is much better for the non-uniform case than it is for the uniform case at the lower traffic load. This is explained by the fact that more vehicles are headed straight and right than are turning left. This increases the throughput, allowing more vehicles to traverse the intersection in a shorter amount of time.

Finally, we examine the stops comparison in Figure 3-10. The ASC/3 appears to per-

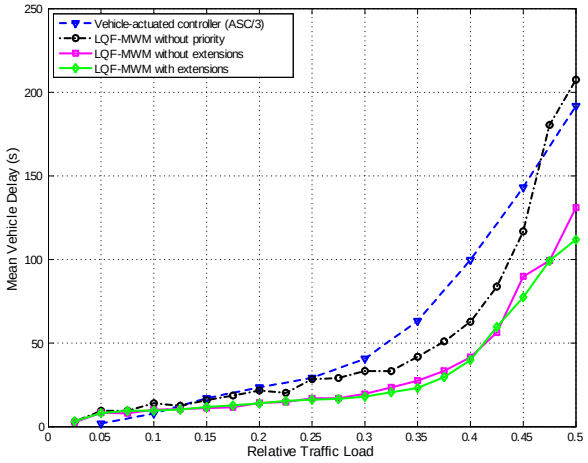

Figure 3-8: Average vehicle delay for non-uniform traffic comprising of trucks only.

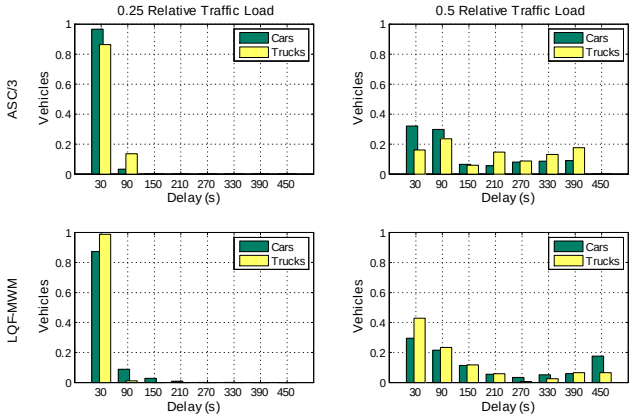

Figure 3-9: Average vehicle delay histogram for various control schemes.

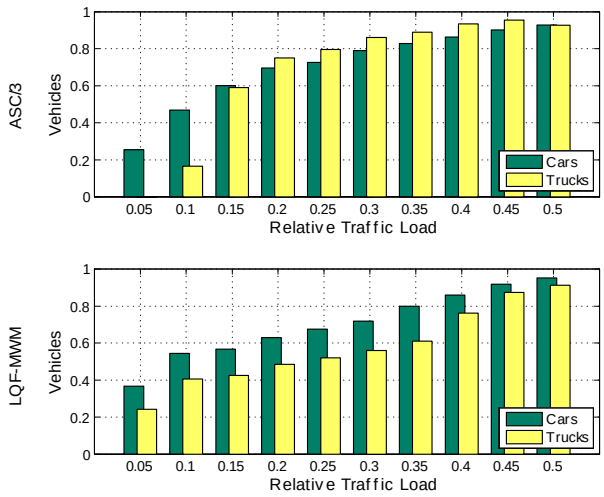

Figure 3-10: Comparison of percentage of truck that reach a stop for ASC/3 and LQF-MWM.

form much better in this case than in the case of uniform traffic flow. Again, this is due to its ability to allow the majority of traffic (straight and right) to move through the intersection unimpeded, breaking only briefly to enable the left-turning traffic through. The LQF controller has more stops at the lowest traffic loads, which is most likely due to the small delay that exists between the time the algorithm detects the vehicle in the queue and when it can activate the traffic light. In particular, the ASC/3 reads the detector information ten times per second, while the LQF controller only reads the detectors once every second. This second of added delay decreases the responsiveness of the controller, affecting the trucks the most because they are unable to change speed as quickly as the cars. Overall, and especially at high traffic loads, the LQF-MWM controller with priority outperforms the ASC/3 by a substantial margin.

Chapter 4

Concluding Remarks

4.1 Conclusion

This paper presents a novel approach to controlling an intersection using notions adapted from the field of computer networking. A stable arbitration algorithm was developed and compared to an existing control technique. A rigorous stability proof has been provided, suggesting that the algorithm is guaranteed to be stable for aggregate traffic flows that do not exceed one third of the physical capacity of each link connected to the intersection. The approach is extremely flexible, enabling dynamic quality of service provisioning for diverse heterogeneous traffic scenarios. Moreover, a Matlab-VISSIM interface is presented 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.

4.2 Future Work

Vehicles entering the traffic network were modeled to follow a Bernoulli process, that is at every time step, there is a certain probability (directly proportional to the relative traffic load) that a vehicle will arrive. While this may be an interesting case in theory, real traffic flows follow a more "bursty" arrival pattern. In future analyses, it may be more appropriate to use a bursty arrival process with *average* vehicle arrival rates equal to those used in this

analysis. With the current simulation interface, it is possible to actually generate the traffic from within the Matlab programming environment. Vehicles can then be added to the simulation either by dynamically changing the input volume or by directly injecting them into the network on a per-vehicle basis.

Having found an algorithm to efficiently control the flow of traffic through an isolated intersection, the way is paved for future work concerning the routing of vehicles through a more complicated traffic network consisting of multiple intersections. This will undoubtedly increase the complexity of the system by an enormous amount. However, with the encouraging results found here, it is possible that the ongoing effort to extend this work will result in a system capable of competing with if not championing current state-of-the-art systems.

Bibliography

Bibliography

[1] E. A. Mueller, "Aspects of the history of traffic signals," *IEEE Transactions on Vehicular Technology*, vol. 19, no. 1, pp. 6–17, Feb 1970.

[2] R. Ghaman, D. Gettman, L. Head, and P. B. Mirchandani, "Adaptive control software for distributed systems," in *IECON 02 [Industrial Electronics Society, IEEE 2002 28th Annual Conference of the]*, vol. 4, Nov 2002, pp. 3103–3106.

[3] T. C. A3A18, "SCATS adaptive traffic system," 1998.

[4] I. Day, "SCOOT-split, cycle, and offsets optimization technique," 1998.

[5] S. S. K. Larry Head, Pitu B. Mirchandani, "The RHODES prototype: A description and some results."

[6] L. H. Pitu Mirchandani, "RHODES traffic-adaptive control systems," 2001.

[7] F. Pooran, "RT-TRACS adaptive control algorithms vfc-opac," Pacific Grove,CA, 1998.

[8] T. Fairbank, "Adaptive control software," 2005.

[9] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural networks for real-time traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 261–272, Sep 2006.

[10] D. Teodorovic, P. Lucic, J. Popovic, S. Kikuchi, and B. Stanic, "Intelligent isolated intersection," in *Fuzzy Systems, 2001. The 10th IEEE International Conference on*, vol. 1, Melbourne, Vic., Australia, 2001, pp. 276–279.

[11] P. Kronborg and F. Davidsson, "MOVA and LHOVRA: traffic signal control for isolated intersections," 1993.

[12] J. R. Peirce and P. J. Webb, "MOVA control of isolated traffic signals-recent experience," in *Road Traffic Control, 1990., Third International Conference on*, London, May 1990, pp. 110–113.

[13] A. Engstrom, "10 years with LHOVRA-what are the experiences?" in *Road Traffic Monitoring and Control, 1994., Seventh International Conference on*, London, Apr 1994, pp. 97–100.

[14] H. R. Smith, B. Hemily, and M. Ivanovic, "Transit signal priority (TSP): A planning and implementation handbook," May 2005.

[15] 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, Dec 1997, pp. 4782–4787.

[16] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[17] G. Birkhoff, "Three observations on linear algebra," in *Univ. Nac. Tucum'n. Rev. Ser. A*, vol. 5, 1946, pp. 147–151.

[18] F. G. Foster, "On the stochastic matrices associated with certain queueing processes," in *Ann. Math Statist.*, vol. 24, 1953, pp. 355–360.

[19] L. R. A. Bacciotti, *Liapunov Functions and Stability in Control Theory, second ed.* Berlin: Springer, 2005.

Vita

Richard Wunderlich was born in Knoxville, Tennessee on November 21, 1982. After graduating from Central High School in Fountain City in 2000, he went on to college at The University of Tennessee in Knoxville. He received his Bachelor of Science degree in Electrical Engineering in May of 2005. After working as an undergraduate research assistant, he began his graduate studies at The University of Tennessee in August of 2006. He received his Master of Science degree in Electrical Engineering in December of 2007.