A Self-Adaptive Collaborative Multi-Agent based Traffic Signal Timing System

Behnam Torabi and Rym Z. Wenkstern
Department of Computer Science
University of Texas at Dallas
Richardson, TX, 75080
{behnam.torabi, rymw}@utdallas.edu

Robert Saylor
City of Richardson's Division of Traffic
and Transportation
Richardson, TX, 75080
robert.saylor@cor.gov

Abstract—**In this paper, we present DALI, a self-adaptive, collaborative multi-agent Traffic Signal Timing system (TST). Intersection controller agents collaborate with one another and adapt their timing plans based on the traffic conditions. Reinforcement learning is used to optimize values for the various thresholds necessary to dynamically determine the scope of collaboration between the agents. DALI was implement in MATISSE 3.0, a large-scale agent-based micro-simulator. Experimental results show an improvement over traditional and reinforcement learning TSTs.**

I. Introduction

Traffic signals impact virtually everyone, every day. Whether on congested or uncongested routes, traffic signals punctuate every urban trip and have a direct impact on drivers, the environment, and the economy [6].

Several Traffic Signal Timing systems (TST) have been proposed by manufacturers, traffic engineers and researchers. The purpose of a TST is to coordinate individual traffic signals to achieve network-wide traffic operational objectives. A TST usually consists of several components: a) a number of intersection controllers, i.e., devices which control the operation of the intersection's traffic signals; b) a communication network; and c) either a central computer or network of computers to manage the overall system. Coordination is implemented through a number of techniques including time-base and hardwired interconnection methods.

Modern TSTs rely upon the detection of traffic conditions in real-time to determine effective signal settings. Generally, conventional TSTs define the traffic signal timing problem as the optimization of a set of timing parameters (e.g., split, cycle length, offset) for an objective function (e.g., minimizing delay, minimizing travel time, maximizing traffic flow). Many conventional TSTs have been proposed by traffic engineers and researchers. *Fully centralized* TSTs [15], [16] allow for efficient coordination of intersection controllers under normal traffic conditions but do not perform well when major traffic disruptions occur. *Partially centralized* TSTs [17], [11] adapt to certain traffic variations within fixed constraints, but require knowledge that is difficult to obtain in practice. *Decentralized* TSTs respond quickly to any traffic demand by generating unconstrained signal timings [13] but use complex optimization algorithms which severely limit their scalability.

The application of the agent paradigm to traffic signal timing has been of interest to Multi-Agent Systems researchers for some time. Distribution, autonomy and coordination are agent properties that are naturally suited for the traffic domain. In the context of traffic signal timing, researchers have proposed the use of a variety of techniques (e.g., game theory [4], neural networks [18], fuzzy logic [5]), including the commonly used Reinforcement Learning (RL). RL-based-solutions attempt to address two types of traffic signal timing problems: non-coordinated and coordinated. In non-coordinated RL-systems, an agent's goal it to optimize the signal timing at its intersections only. The lack of coordination between agents often leads to a degradation of the overall traffic conditions. On the other hand, in coordinated agent-systems, agents implicitly coordinate with their direct neighbors by sharing their states and intended actions. Given the astronomical number of state-action pairs that need to be considered for any realistic traffic model, coordinated RL-systems have no option but to simplify the traffic model. Another category of agent-based systems based on vehicle-to-vehicle and vehicle-to-infrastructure (V2X) communications has been proposed [21], [8]. Although some of the systems in this category provide impressive simulation results [8], they are based on assumptions that do not have their counterparts in the real world. In addition, V2X communication technologies are still in their infancy and their global deployment is decades away.

In this paper we present a self-adaptive, collaborative multi-agent Traffic Signal Timing system (TST) that we call DALI (Distributed Agent-based traffic LIghts). In DALI, intersection controllers are augmented with software agents which dynamically form groups and collaboratively adapt their signal timings by considering the feedback of all controller agents impacted by a change. DALI is intended to be deployed in the City of Richardson, Texas, with minimal changes to the TST infrastructure. As such, our model is based on parameters and data currently used by the city. Our work differs from existing solutions in that it offers the following combined set of features: 1) it uses a dynamic, collaborative agent-based strategy that involves the feedback of more than direct neighbors; 2) it makes use of an RL method to dynamically adapt the value of thresholds; 3) it is based on a real-world TST and makes use of real-world traffic data; 4) it has been validated on a simulated model of the City of Richardson comprising 128 signalized intersections and 1365 road segments.

We implemented our agent-based coordinated model in MATISSE 3.0, a large-scale multi-agent traffic simulation system. Experimental results show that DALI outperforms the SCATS-based system currently in use in the City of Richardson as well as a version of the RL-based MARLIN-ATSC [10] system.

The remainder of this paper is organized as follows: Section II reviews existing works. Section III gives an overview of the concepts used in this paper. Section IV discusses the agent algorithms, and Section VI presents the experimental results.

II. Related Work

A. Conventional TST

As mentioned in Section I, conventional Traffic Signal Timing systems (TST) formulate the traffic signal timing problem as an optimization problem. We classify these systems as *fully centralized*, *partially centralized* and *decentralized*.

Fully centralized TST are systems in which intersection controllers are fully controlled by one or several higher level entities often called *master* or *regional* computers. The higher-level computer is responsible for processing traffic data received from the intersection controllers. In SCOOT [15], the master computer optimizes the traffic signal timing settings to reduce vehicle delays and stops. In SCATS [16], regional computers modify the signal timing parameters for their regions. The intersection controllers are then responsible for executing the new timing plans. Although both SCOOT and SCATS have been deployed since the 80s, they are not adequate to deal with unexpected traffic disruptions. TUC [7] is a recent centralized TST which formulates the signal timing problem as a linear-quadratic optimal control problem. TUC considers all traffic intersections simultaneously through the application of a simple matrix equation. As such, TUC is able to achieve highly efficient and extremely simple traffic signal timing strategies in large traffic networks. Although TUC was deployed in an area of Glasgow and has proven to be efficient, its centralized architecture requires that the strategy be completely re-designed when the traffic network is modified or expanded [7].

Partially centralized TST are systems in which the intersection controllers have full control over the definition and execution of the signal timing plans for their intersection, but the intersection controllers' coordination or control strategies are defined and monitored by a higher-level computer. Examples in this category include OPAC [11] and RHODES [17]. Both systems do not employ the traditional traffic parameters in their optimizations. They utilize traffic flow models that predict vehicle arrivals at the intersection, and adjust the timing of each phase to optimize an objective function such as delay. Because they emphasize traffic prediction, these systems can respond to variations in traffic flow. Nevertheless, the use of dynamic programming increases the computational load. In addition, both systems require the complete knowledge of arrivals over the control period, which is difficult to obtain with accuracy in practice. RHODES has been validated on a simulated network consisting of nine intersections. OPAC has been integrated as part of the RT-TRACS system and was tested in a network consisting of 16 signalized intersections in Northern Virginia [12].

Finally, *decentralized TST* are systems where decision making for signal timing plans and network coordination is given to the intersection controllers. Although a central computer may exist, its responsibility is limited to overall traffic monitoring and data management. PRODYN (Programmation Dynamique) was an attempt at developing a distributed TST. In PRODYN, the basic optimization criterion is the minimization of delay which is achieved by a Bayesian estimation of queue lengths. PRODYN employs dynamic programming, and signal plans ar generated in real-time. Due to the exponential complexity of the method, the solutions cannot be applied to more than a very few intersections.

B. Agent-based TST

Agent-based TST utilize a variety of techniques such as game theory [4], neural networks [18], fuzzy logic [5], and Reinforcement Learning (RL) [9], [1], [3], [10]. In this section we restrict our discussion to the agent-based systems which implement the commonly used RL. A comprehensive review of the use of other techniques can be found in [19].

In [1] and [9] the authors discuss non-coordinated agent-based RL models which use Q-learning to approximate the reward values. The proposed models aim at finding optimal signal timings at isolated intersections. In MAS, it is well-known that isolated decision making may lead to undesirable effects on the global system. The proposed models were tested on one simulated intersection and the experimental results show that they both outperformed a pre-timed strategy. In addition the model discussed in [9] is based on the assumption that parameters such as cumulative delay are widely available, which is not the case in a real-world setting.

In coordinated models, agents implicitly coordinate with their direct neighbors by sharing their states and intended actions. In [3] the RL approach considers joint states and actions. Given that, generally, this approach would lead to an exponential number of $< state, action >$ pairs, the authors propose to partition the set of local agents into groups of three, and assign a supervisor agent whose responsibility is to recommend actions to local agents. In addition, each intersection can be in only one of three "coarse-grained" states. Experiments were conducted on a simulated grid network consisting of 64 nodes connected through unidirectional links (i.e., one way roads). [10] discusses the well referenced MARLIN-ATSC. In MARLIN-ATSC, agents can operate in either independent or integrated mode. In integrated mode, agents coordinate actions with neighboring agents by implementing a multi-agent modular Q-learning. Similarly to [3], in order to reduce the problem complexity, only $< state, action >$ pairs with neighboring controllers are considered. At each time step, the intersection agents communicate their $< state, action >$ with their direct neighbors. Following this exchange of information, each agent selects the action that maximizes its reward as well as one of its neighbors'. MARLIN-ATSC was tested on a simulated network of the Lower Downtown Toronto network comprising 59 intersections. Experimental results show that the proposed approach reduced the average intersection delay compared to the traditional pre-timed, semi-actuated and fully actuated approaches. The main drawback of MARLIN-ATSC is the assumption that an intersection controller can only consider the interest of one neighboring controller. In addition, the computation of the reward function is based on the total cumulative delay of vehicles at the intersection. Although this parameter can be easily obtained in a simulated environment, it requires technologies that are not widely deployed in the field.

Other varieties of RL-based approaches for TST have been discussed in the literature. Generally speaking, the main limitations of RL-based models are their restricted collaboration with direct neighbors, their poor performance in highly dynamic scenarios, and their use of parameters that cannot be easily obtained in a real-world setting. For a comprehensive review of these models, we refer the interested reader to [20].

In this paper we present DALI, a self-adaptive coordinated agent-based traffic signal timing model. DALI uses a collaborative agent-based strategy that considers the feedback of all agents that may be affected by a change at any given time. As such, agents form collaboration groups dynamically and adapt their signal timings based on current traffic conditions. They also use an RL-based method to adapt the values of certain thresholds. Unlike most RL-based systems, DALI does not make use of parameters that cannot be obtained in the field. Finally, it has been validated on the largest realistic simulated traffic network published to date for collaborative multi-agent based TSTs.

III. Traffic Concepts

The definitions, standards and procedures given in this section are based on the U.S. Department of Transportation Traffic Signal Timing Manual [14] and the City of Richardson's Traffic & Transportation procedures.

A. Traffic Concepts

An *intersection controller* is an electrical device mounted in a cabinet at an intersection to control the operations of the traffic lights. Controllers receive real-time information from detection systems and adjust the signal timings based on the sensed information.

Timing Parameters

Phase: A controller timing unit associated with the control of one or more movements (i.e., through movement, right turn movement) at an intersection. Most controllers sold today provide eight phases to serve standard four-legged intersections

Interval: Duration of time during which the signal indications do not change. Examples of intervals include green, yellow and red intervals.

Minimum Green: The first timed portion of the *green interval* which may be set in consideration of the number of vehicles between the phase detector and the stop line.

Maximum Green: This time setting defines the maximum length of time that a phase can be green where there is a demand for a conflicting vehicle flow.

Coordination of traffic signal phases

It is the ability to synchronize multiple intersections to enhance the operation of one or more directional movements in a traffic system. In general terms, there are three basic parameters that, when taken together, define a coordinated traffic signal plan. These are:

Cycle Length: This is the total time to complete one sequence of signalization around an intersection.

Offset: This is the time relationship, expressed in seconds, between coordinated phases at subsequent traffic signals.

Split: This is the time assigned to a phase during coordinated operations.

It is important to note that in the context of traffic management, coordination refers to the setting of the above mentioned parameters. It does not correspond to the collaboration concept used in multi-agent systems.

Traffic Signal Operation Modes

In *pre-timed mode*, phases and cycles are pre-set according to a predetermined schedule, based on historic traffic patterns. In *semi-actuated mode*, detectors are placed only on the main street approaches. The main street has green until the actuation of a side street detector. The side street then receives a green phase until either all vehicles are served served (gap out) or a preset maximum green is reached (max out). In *fully actuated mode*, all approaches have detectors. The signal phases are controlled by detector actuations. Minimum greens and maximum greens are specified for each phase.

B. The City of Richardson's TST

The City of Richardson is located 15 miles north of downtown Dallas and is part of the Dallas-Fort Worth Metroplex. The city has four major highways, eleven major and six minor arterial roads and 128 intersections with traffic signals.

The 128 signalized intersection are controlled by SCATS-controllers [16] mounted in cabinets at the intersections. The controllers run Linux on an ATC-compliant motherboard offering speed, performance and multi-thread capabilities. A central traffic management center communicates with the traffic controllers via a WiMAX wireless network operating in the licensed 4.9 GHz public safety band with about 2.5 GB/s total throughput. Controller-to-Controller communication links exist but are not used in the current traffic system. Traffic controllers operate in various modes. During the day, a variety of pre-timed plans designed to address variable traffic patterns are executed based on traffic conditions. Past midnight, controllers operate either in pre-timed, semi-actuated or fully-actuated modes depending on the road types and the existence of a detection system.

Vehicles at an intersection are detected through inductive loops. An inductive loop is a coiled wire that is formed into a loop and installed under the surface of roadways at appropriate distances before the stop bar based on the traffic and roadway conditions. When a vehicle passes over the loop or is stopped within its area, a pulse is sent to the traffic signal controller signifying the passage or presence of a vehicle. The controller stores the detection information and the time of its occurrence in a local database.

The City of Richardson maintains a traffic count program which conducts scheduled counts on major arterial roads as well as collector streets, i.e., roads which move traffic from local streets to arterial roads. The traffic counts are used for a variety of purposes including the definition of coordinated traffic signal timing along arterial streets.

In order to define traffic signal timing plans, traffic engineers assign values to cycle length, offset and splits based on historical data. Given that inductive loops are positioned a few feet from the stop bar, the vehicles that can be realistically detected are those that cross the inductive loop area. With the inductive loop technology a complete vehicle count on a road segment is not possible. In addition, except for the induction loop area, the vehicle positions on road segments cannot be obtained.

IV. Algorithms for an Agent-Based TST

In this section we discuss the core algorithms executed by DALI's intersection controller agents. We start by defining the sets and functions used in the algorithms. We restrict our discussion to the main scenario. A detailed discussion on the special cases is given in [19].

A. Model Definition

$T = \{t_1, .., t_i\}$ is the set of time-stamps at which traffic conditions are evaluated.
$C = \{c_1, .., c_n\}$ is the set of intersection controllers. An intersection controller c_n is assigned a weight ω which corresponds to its priority in the road network.
$Rd = \{r_{c_1,c_2}, .., r_{c_m,c_n}\}$ is the set of road segments between intersections.
$LN_{r_{c_m,c_n}}$ is the set of lanes for road segment r_{c_m,c_n}.
$PH_{c_n} = \{ph_{c_n,1}, ..ph_{c_n,k}\}$ is the set of phases for the intersection controlled by c_n.
A phase $ph_{c_n,k}$ is defined in terms of γ, the split time, ν, the minimum green time, η, the maximum green time, the yellow time, the red time and $LN_{ph_{c_n,k}}$, the set of lanes it applies to.
$p(r_{c_m,c_n}.ln_w, r_{c_n,c_p}.ln_u)$ is the probability that a vehicle exiting lane w in road segment r_{c_m,c_n} enters lane u in road segment r_{c_n,c_p}. This probability is computed by traffic engineers based on historical data.
$p(r_{c_m,c_n}, r_{c_m,c_n}.ln_w)$ is the probability that a vehicle which enters road segment r_{c_m,c_n}, leaves it from lane w. This probability is also computed by traffic engineers based on historical data.
$rateOut(t_i, \tau, r_{c_m,c_n}.ln_w)$ is the rate of vehicles (per second) that can leave the intersection through lane w of road segment r_{c_m,c_n} within the time interval τ that ends at time t_i.
$rateIn(t_i, \tau, r_{c_m,c_n})$ is the rate of vehicles (per second) that enter road segment r_{c_m,c_n} in the time interval τ that ends at time t_i.
$\xi_{r_{c_m,c_n}.ln_w}(t_i, \tau)$ is the *traffic flow rate* for lane $r_{c_m,c_n}.ln_w$, i.e., the ratio of vehicles getting in and leaving the lane. It is defined as:

$$\xi_{r_{c_m,c_n}.ln_w}(t_i, \tau) = \frac{rateIn(t_i, \tau, r_{c_m,c_n}) \times p(r_{c_m,c_n}, r_{c_m,c_n}.ln_w)}{rateOut(t_i, \tau, r_{c_m,c_n}.ln_w)}$$

B. DALI Agent Algorithms

In DALI, agents collaborate with one another to dynamically respond to traffic changes. In this section we discuss the agent algorithms at the basis of the collaborative approach. The algorithms make use of parameters which are assigned fixed values based on historical traffic data.

An intersection controller c_n continuously evaluates the traffic state to determine if a re-timing operation is necessary.

1) Detecting Congestion: At each evaluation cycle, c_n receives $rateIn$ (detected through c_m's induction loops) and determines $rateOut$.

At time t_i, controller c_n computes $Cong_{t_i,ph_{c_n,k}}$ as the average throughput for the set of lanes controlled by $ph_{c_n,k}$.

$$Cong_{t_i,ph_{c_n,k}} = \sum_{r_{c_m,c_n}.ln_w \in LN_{ph_{c_n,k}}} \xi_{r_{c_m,c_n}.ln_w}(t_i, \tau)$$

If $Cong_{t_i,ph_{c_n,k}}$ is greater than threshold a, then c_n considers that there is an *instant congestion* and assigns the value of 1 to $InstantCongestion$.

$$InstantCongestion_{t_i,ph_{c_n,k}} = \begin{cases} 1 & Cong_{t_i,ph_{c_n,k}} \geq a \\ 0 & Cong_{t_i,ph_{c_n,k}} < a \end{cases}$$

It proceeds by considering the past b evaluation cycles to determine the percentage of evaluation cycles in which the phase was congested. This is defined as

$$PercentCong_{t_i,ph_{c_n,k}} = \frac{\sum_{j=i-b}^{i} InstantCongestion_{t_j,ph_{c_n,k}}}{b} \times 100$$

If $PercentCong_{t_i,ph_{c_n,k}} > d$ then c_n considers the road lanes controlled by $ph_{c_n,k}$ as congested.

2) Generate New Plan: c_n deliberates to determine the value of a new split that will alleviate congestion on $ph_{c_n,k}$. The value of the new split is calculated as:

$$plan_{new}.phase.\gamma = plan_{cur}.phase.\gamma \times (e + \frac{\sum_{j=i-\nu}^{i} Cong_{t_j,ph_{c_n,k}}}{\nu} \times f)$$

where e and f are coefficients that regulate the influence of the traffic throughput and the current split time.

3) Request For Evaluation: c_n determines the impact of executing the new plan on its neighboring intersections in terms of κ, the increment in vehicle rate. $\kappa_{r_{c_m,c_n}.ln_w}$ is calculated for road lane $r_{c_m,c_n}.ln_w$ as:

$$\kappa_{r_{c_m,c_n}.ln_w} = rateOut(t_i, \tau, r_{c_m,c_n}.ln_w) \times \frac{(plan_{new}.phase.\gamma - plan_{cur}.phase.\gamma)}{plan_{new}.phase.\gamma}$$

$\kappa_{ph_{c_n,k}}$ for a phase $ph_{c_n,k}$ is defined as the sum of $\kappa_{r_{c_m,c_n}.ln_w}$ for the set of lanes controlled by the phase. In the same way, $\kappa_{r_{c_n,c_p}}$ for a road segment r_{c_n,c_p}, is the sum of $\kappa_{r_{c_n,c_p}.ln_w}$. Controller c_n proceeds by sending $plan_{new}$, $\kappa_{r_{c_n,c_p}}$ and $\kappa_{ph_{c_n,k}}$ to each adjacent controller c_p for evaluation.

4) Compute Level Of Agreement: Upon receipt of a new plan, c_n's neighboring controller c_p computes $\kappa_{r_{c_p,c_q}}$ for each of its neighbor c_q and request that they in turn evaluate the plan. The process propagates until at a given intersection, either the value of κ is smaller than threshold g or the plan reaches the road network boundaries. Following this step and recursively, each controller sends back its level of agreement in terms of a real number Ψ, to the controller from which it has received the request. A c_p, calculates Ψ_{c_p} based on the existing traffic throughput, its priority ω and the ratio of the received additional vehicle throughput. After receiving the level of agreement from all involved neighbors, c_p combines them with its own level of agreement Ψ_{c_p} and sends the value back to c_n. The final decision is made based on the value of Ψ_{c_n} representing the feedback of all involved controllers.

C. Adaptive Assignment of Thresholds Values

The algorithms discussed in Section IV-B assign fixed values to various thresholds based on historical data. In order to make the collaboration process between agents more adaptive, it is necessary that the values be updated dynamically. In this section we discuss how an RL-based approach is used to dynamically select favorable values for thresholds a, d and g. Threshold a controls the agent's sensitivity to detecting congestion. The lower the values of a, the higher the likelihood for an agent to detect congestion. Threshold d controls the time duration that a phase should be flagged as congested in order to be considered as congested. Finally, g controls the collaboration scope. With lower values of g, a higher number of agents will be involved in the decision-making for a new timing plan.

1) Environment State: Given that queue lengths are not measurable in a real-world TST, we use the traffic flow rate to define the environment state. As mentioned in Section IV-B, at time t_i, the traffic flow rate for road segment r_{c_m,c_n} is defined as:

$$\xi_{r_{c_m,c_n}.ln_w}(t_i, \tau) = \frac{rateIn(t_i, \tau, r_{c_m,c_n})}{\sum_{r_{c_m,c_n}.ln_w \in LN_{r_{c_m,c_n}}} rateOut(t_i, \tau, r_{c_m,c_n}.ln_w)}$$

The environment state for each intersection controller c_n is the set of states of its incoming roads. In order to avoid the state-action dimensionality problem inherent to RL approaches, we assign the rates of *low*, *medium* or *high* to the traffic flow rates, based on their values.

2) Action Selection: An agent c_n's action is to assign a value to a threshold. In order to avoid the dimensionality problem, we only allow the assignment of specific values for a, d, and g. These correspond to values which are meaningful in a real-world setting and were derived from experimental data. For example, d which represents the duration that a phase should be flagged as congested in order to be considered as congested, can take the values of 25%, 50%, 75% or 99%. This means that, for the case d is given the value of 75, a controller c_n – which analyzed the state of phase $ph_{c_n,k}$ for the past x minutes and found that 75% of the time the phase was congested – will change the phase status to congested. a can take values of $\{0.5, 1, 1.5, 2\}$, and g values of $\{0.01, 0.2, 0.4, 0.6\}$.

3) Rewards: In this paper we discuss two reward types.
Rewards for minimizing delay (r_D). This reward is used to update the Q-values of a, d and g. Reward r_D is defined as the variation in the traffic flow rate. For road segment r_{c_m,c_n} in the time interval τ (e.g., three minutes) that ends at time t_i, $\xi_{r_{c_m,c_n}}(t_i, \tau)$ is defined as the sum of traffic flow rate of its lanes.

Intersection controller c_n computes the reward of an action act at time t_i and state s as:

$$r_D(act, s, t_i) = \xi_{r_{c_m,c_n}}(t_i, \tau) - \xi_{r_{c_m,c_n}}(t_i + \tau, \tau)$$

Rewards for controlling collaboration scope (r_C)
This reward is used together with reward r_D to update the Q-value of threshold g. Reward r_C is computed as:

$$r_C(act, s, t_i) = r_D(act, s, t_i) + (1 - 2 \times \frac{N_{plan_{new}}}{N})$$

where $N_{plan_{new}}$ is the number of intersections that are involved in the decision about the new plan and N is the total number of intersections in the network.

V. MATISSE

In this section we give a brief overview of MATISSE 3.0 (MultiAgent based TraffIc Safety Simulation systEm). MATISSE 3.0 is a microscopic multi-agent based simulation system for the specification and execution of simulation scenarios for agent-based intelligent transportation systems. Figure 1 shows the high-level architecture of MATISSE.

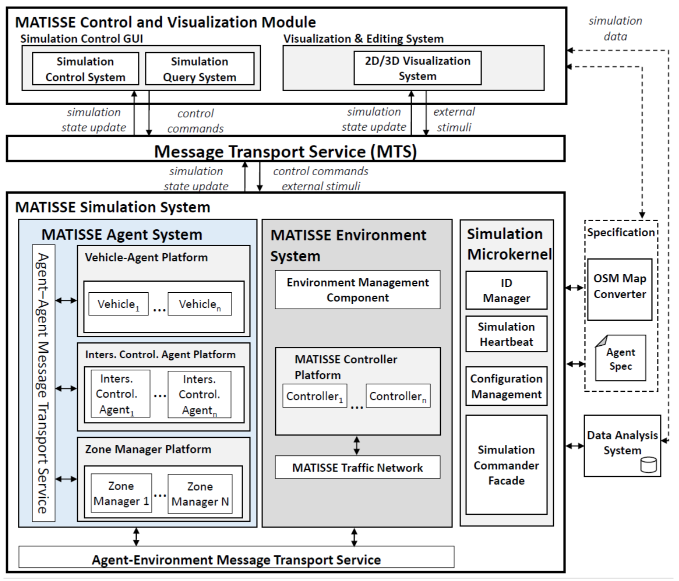
MATISSE Control and Visualization Module
Simulation Control GUI
Simulation Control System
Simulation Query System
Visualization & Editing System
2D/3D Visualization System
simulation data
simulation state update
control commands
simulation state update
external stimuli
Message Transport Service (MTS)
simulation state update
control commands external stimuli
MATISSE Simulation System
MATISSE Agent System
Agent–Agent Message Transport Service
Vehicle-Agent Platform
Vehicle$_1$ … Vehicle$_n$
Inters. Control. Agent Platform
Inters. Control. Agent$_1$ … Inters. Control. Agent$_n$
Zone Manager Platform
Zone Manager 1 … Zone Manager N
MATISSE Environment System
Environment Management Component
MATISSE Controller Platform
Controller$_1$ … Controller$_n$
MATISSE Traffic Network
Simulation Microkernel
ID Manager
Simulation Heartbeat
Configuration Management
Simulation Commander Facade
Specification
OSM Map Converter
Agent Spec
Data Analysis System
Agent-Environment Message Transport Service

Fig. 1. High-level architecture of MATISSE.

MATISSE consists of three modules. The main constituent is the MATISSE Simulation Module which consists of three subsystems. The Agent System creates and manages simulated traffic agents. Due to the complexity of the simulated agents and to enhance future extensiblity, agent types are implemented as three separate platforms: 1) The Vehicle Platform creates and manages Vehicle agents representing regular or autonomous vehicles; 2) the Intersection Control Agent Platform creates and manages agent-based or regular intersection controllers; and 3) the Zone Manager Platform creates and manages service and traffic manager agents.

The Agent Message Transport Service is used to simulate various types of communications (vehicle-to-vehicle, vehicle-to-infrastructure, infrastructure-to-infrastructure) .

The Environment System maintains a detailed specification of any real-world complex traffic network topology.

The Control and Visualization Module receives traffic simulation information through the MTS. It renders 2D and 3D representations of the simulation and provides mechanisms for the user to interact with the simulation and modify parameters at run-time.

MATISSE can simulate traffic networks imported directly from Open Street Map. It uses advanced algorithms to automatically generate missing information (e.g., unknown road types or traffic light locations). During the simulation, vehicles enter and exit the simulation from entry and exit points. At the start of the simulation, the user can define their own vehicle distribution for preferred entry and exit points or let MATISSE generate an initial normal distribution. During the simulation, the user can modify the driver's level of distraction. This may dynamically introduce unexpected accidents and unpredicted traffic behavior. Demos of the MATISSE simulator are available at *mavs.utdallas.edu/its*

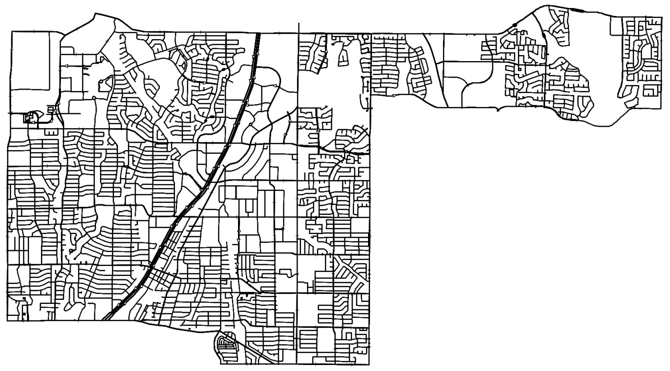

Fig. 2. 2D visualization of Richardson's Traffic Network.

VI. Experimental Results

Various measurements are commonly used in Texas to evaluate the effectiveness of a signal timing plan. These include *delay*, *queue length*, *number of stops* and *occupancy*. In this section, we discuss the evaluation of DALI with respect to delay. Delay is defined as the additional travel time caused by traffic control devices, compared to the travel time if a vehicle was to maintain its expected speed in the absence of a control device [2].

The experiments were run on a multicore PC (Intel Core i7 X980 CPU (3.33GHz), 6.00 GB, 64-bit Windows 7). A simulated model of the City of Richardson's road network was created in MATISSE. The model includes 1365 road segments and the city's 128 signalized intersections in addition to the 965 non-signalized intersections. Figure 2 shows a 2-D representation of the traffic network. Tables I and II summarize the types of signalized and non signalized intersections, classified based on the number of incoming and outgoing lanes.

TABLE I
Number of Signalized Intersection with various incoming and outgoing lanes

Type	1×1	1×2	1×3	2×2	2×3	3×3
Count	0	4	8	18	29	69

TABLE II
Number of Non-Signalized Intersection with various incoming and outgoing lanes

Type	1×1	1×2	1×3	2×2	2×3	3×3
Count	533	241	175	16	0	0

Simulation settings were run five times for 86,400 simulation cycles representing a 24-hour time period. The average delay for all vehicles was measured. In the experiment, we use real-world data provided by the City of Richardson to simulate regular traffic patterns. We compare the efficiency of DALI with fixed threshold values of $a = 0.5$, $d = 75$

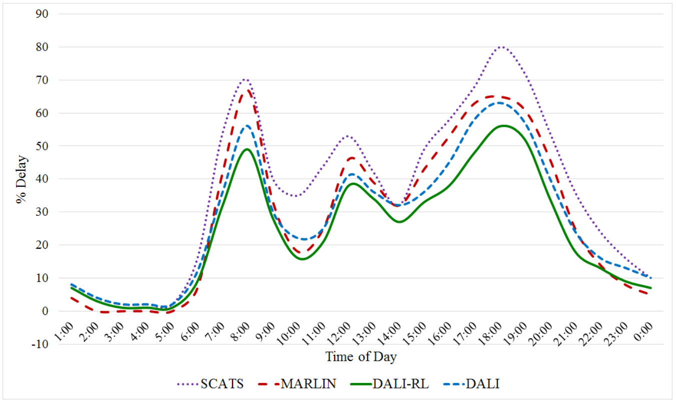

Fig. 3. Average delay using traffic data from the City of Richardson

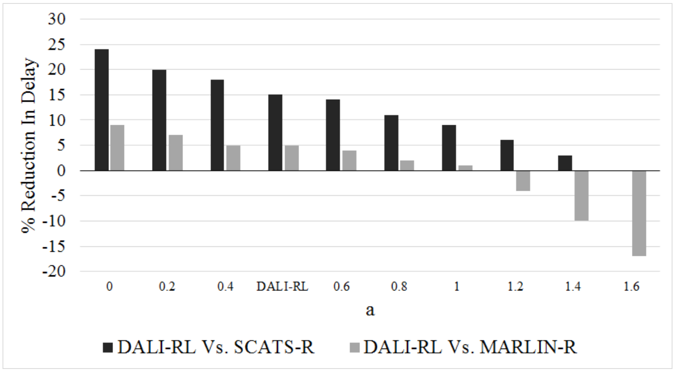

Fig. 4. Average Delay For Different Values of a.

and $g = 0.4$, the SCATS-based system currently in use in Richardson (SCATS-R), DALI with adaptive threshold values (DALI-RL), and a model of the RL-based MARLIN-ATSC [10] (MARLIN-R). Both DALI-RL and MARLIN-R Q-values were initialized based on estimations derived from historical data provided by the City of Richardson.

Experiment 1: Assessing Delay

As shown in Figure 3, between the times of 00:30 am and 5:30 am, DALI and SCATS-R perform at the same level with respect to delay. This is due to the fact that during this time period, traffic is very light and therefore DALI agents do not perform any action. As expected, DALI-RL performs better (21% delay reduction) in comparison with DALI and SCATS-R. MARLIN-R agents also perform better (53% delay reduction) than DALI because of their flexibility in changing the traffic phases at any time. As we progress during the day (i.e., 6:30 am to 8:30 am) the traffic flow increases, and congestion is detected. DALI agents naturally collaborate with one another to define and implement timing plans that meet the network conditions. As such, DALI performs significantly better than SCATS-R (23% delay reduction). DALI performs slighly better than MARLIN-R (4% delay reduction). The simulation shows that this is due to the fact that MARLIN-R agents do not handle heavy traffic in small network areas with a large number of intersections efficiently. In those cases, MARLIN-R agents give the right-of-way to vehicles without taking into account the downstream roads which are congested. DALI-RL agents perform better in comparison with DALI agents (7% delay reduction) by adaptively selecting threshold values.

Experiment 2: Assessing Changing Values of Threshold a
Figure (4) compares the performance of DALI with MARLIN-R and SCATS-R for different values of a. For lower values of a, agents almost continuously collaborate to adapt their traffic signals. This results in lower average delay since agents do not wait until higher levels of congestion are reached to act. Nevertheless, as shown in table III, lower values of a result in a very large number of exchanged messages. Higher values of a decrease requests for retiming and consequently the average delay is increased. As shown in figure 4 and table III, the adaptive selection of a allows DALI-RL agents to perform better for both average delay and number of message exchanges.

TABLE III
NUMBER OF MESSAGE EXCHANGES FOR DIFFERENT VALUES OF a.

Value of a	Number of Exchanges
0.0	7,155,289
0.2	358,401
0.4	156,272
0.6	95,478
DALI RL	30,409
0.8	17,654
1.0	7,689
1.2	4,859
1.4	468
1.6	200

Experiment 3: Assessing Changing Values of Threshold g

Figure (6) shows the average size of groups that are formed dynamically when a re-timing is called for by a controller agent.

When g is equal to zero, the propagation of requests does not stop, and therefore all the controller agents end up being involved in the collaborative re-timing process.

As g increases, the average group size becomes smaller and therefore fewer communications are needed. Using a fixed value for g is not always efficient because in certain unexpected circumstances it may better to increase the collaboration scope. Figures 5 and 6 show that when agents use RL to determine g values, better performance is achieved with fewer communications due to the smaller group size. As illustrated in figure 5, when the value of g is less than 0.4, no significant improvement occurs with respect to average delay. This is explained by the fact that, broadening the collaboration scope to include agents that are not impacted by the plan does not have any effect on the final outcome.

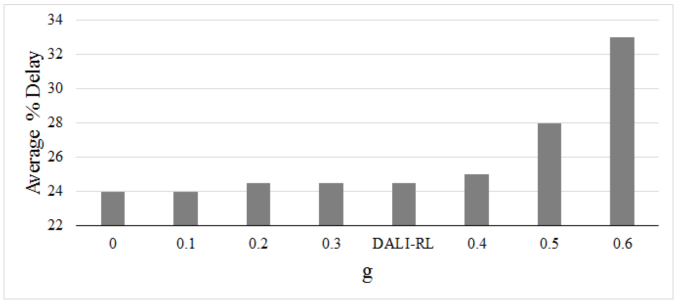

Fig. 5. Average delay For Different Values of g.

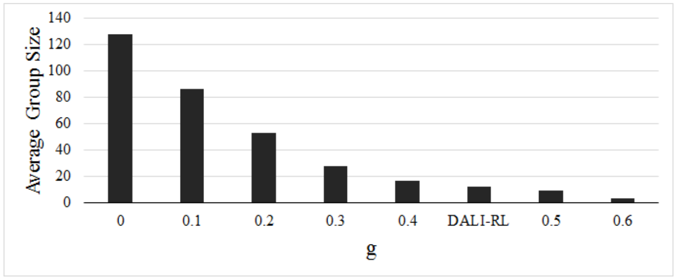

Fig. 6. Average Group Size For Different Values of g.

VII. Conclusion

In this paper we presented DALI, a self-adaptive, collaborative multi-agent traffic signal timing (TST) system for congestion management. DALI has been validated on a simulated model of City of Richardson's traffic network. The experimental results show that with or without the addition of RL, the collaborative DALI controllers outperforms the traditional SCATS-based system currently used by the City of Richardson. While a simulated model of MARLIN-ATSC performs better than DALI (with fixed threshold values) in light traffic conditions, it does not operate efficiently in traffic conditions with heavy traffic in condensed network areas. DALI-RL performs well in all traffic conditions. Our goal is to further explore the added value of RL to DALI agents.

This work is a first step towards the implementation of an agent-based TST for the City of Richardson, in Texas. Before the deployment of the first prototype, agent-to-agent communication costs need to be assessed. Our assumption is that, given that the currently deployed SCATS controllers communicate through a WiMAX network with a speed of up to 2.5 Gbps, direct agent communication may take less than a tenth of a second, and communications for decision making no more than a few seconds. Also, in its current form, the proposed agent-based TST does not take pedestrians into consideration. Given that close to 90% of Richardson's population commutes by either driving alone or carpooling, it is reasonable to assume that current pedestrian signal operations may not need to be modified. Nevertheless, we plan to incorporate pedestrian signal timing in future versions of our agent-based model.

References

[1] Baher Abdulhai, Rob Pringle, and Grigoris J Karakoulas. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, 129(3):278–285, 2003.

[2] Kevin N Balke and Curtis Herrick. Potential measures of assessing signal timing performance using existing technologies. Technical Report FHWA/TX-04/0-4422-1, Texas A&M Transportation Institute, College Station, Texas 77843-3135, July 2004.

[3] Ana LC Bazzan, Denise de Oliveira, and Bruno C da Silva. Learning in groups of traffic signals. *Engineering Applications of Artificial Intelligence*, 23(4):560–568, 2010.

[4] Shih-Fen Cheng, Marina A Epelman, and Robert L Smith. Cosign: A parallel algorithm for coordinated traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):551–564, 2006.

[5] Mario Collotta, Lucia Lo Bello, and Giovanni Pau. A novel approach for dynamic traffic lights management based on wireless sensor networks and multiple fuzzy logic controllers. *Expert Systems with Applications*, 42(13):5403–5415, 2015.

[6] Christopher M Day, Thomas M Brennan Jr, Hiromel Premachandra, Alexander M Hainen, Stephen M Remias, James R Sturdevant, Greg Richards, Jason S Wasson, and Darcy M Bullock. Quantifying benefits of traffic signal retiming. Final Report FHWA/IN/JTRP-2010/22, Joint Transportation Research Program, Purdue University, 610 Purdue Mall, West Lafayette, IN 47907, October 2010.

[7] Christina Diakaki, Markos Papageorgiou, and Kostas Aboudolas. A multivariable regulator approach to traffic-responsive network-wide signal control. *Control Engineering Practice*, 10(2):183–195, 2002.

[8] Kurt Dresner and Peter Stone. A multiagent approach to autonomous intersection management. *Journal of artificial intelligence research*, 31:591–656, 2008.

[9] Samah El-Tantawy and Baher Abdulhai. An agent-based learning towards decentralized and coordinated traffic signal control. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 665–670. IEEE, 2010.

[10] Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): methodology and large-scale application on downtown toronto. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1140–1150, 2013.

[11] Nathan H Gartner. *OPAC: A demand-responsive strategy for traffic signal control*. Number 906. January 1983.

[12] Nathan H Gartner, Farhad J Pooran, and Christina M Andrews. Implementation of the opac adaptive control strategy in a traffic signal network. In *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE*, pages 195–200. IEEE, 2001.

[13] Jean-Jacques Henry, Jean Loup Farges, and J Tuffal. The prodyn real time traffic algorithm. *IFAC Proceedings Volumes*, 16(4):305–310, 1983.

[14] Peter Koonce, Lee Rodegerdts, Kevin Lee, Shaun Quayle, Scott Beaird, Cade Braud, Jim Bonneson, Phil Tarnoff, and Tom Urbanik. Traffic signal timing manual. Publication FHWA-HOP-08-024, Department of Transportation, 1200 New Jersey Ave, SE, Washington, DC 20590, June 2008.

[15] UK's Transport Research Laboratory. Scoot: Split cycle and offset optimisation technique, 2018. https://trlsoftware.co.uk/products/traffic_control/scoot, accessed 2018-04-30.

[16] P R Lowrie. Scats: sydney co-ordinated adaptive traffic system, 2018. http://www.scats.com.au/, accessed 2018-04-30.

[17] Pitu Mirchandani and Larry Head. A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415–432, December 2001.

[18] Dipti Srinivasan, Min Chee Choy, and Ruey Long Cheu. Neural networks for real-time traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 7(3):261–272, 2006.

[19] Behnam Torabi. A self-organizing traffic management system and its real-world implementation. Ph.D. Proposal, 2017.

[20] Kok-Lim Alvin Yau, Junaid Qadir, Hooi Ling Khoo, Mee Hong Ling, and Peter Komisarczuk. A survey on reinforcement learning models and algorithms for traffic signal control. *ACM Computing Surveys (CSUR)*, 50(3):34, 2017.

[21] Maram Bani Younes and Azzedine Boukerche. Intelligent traffic light controlling algorithms using vehicular networks. *IEEE transactions on vehicular technology*, 65(8):5887–5899, 2016.