2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

A Collaborative Agent-Based Traffic Signal System For Highly Dynamic Traffic Conditions

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 distributed, collaborative multi-agent Traffic Signal Timing system (TST) for highly dynamic traffic conditions. In DALI, intersection controllers are augmented with software agents which collaboratively adapt signal timings by considering the feedback of all controller agents that may be affected by a change. The model is based on a real-world TST and is intended to be deployed with minimal changes to the infrastructure. DALI has been validated on a simulated model of the City of Richardson, Texas, comprising 128 signalized intersections. The experimental results show that it outperforms the conventional traffic operation modes as well as an RL-based TST in highly dynamic scenarios.**

I. INTRODUCTION

Modern Traffic Signal Timing systems (TST) rely upon the detection of traffic conditions in real-time to determine effective signal settings. Generally, TSTs define traffic signal planning as an optimization problem where solutions are timing plans which meet objectives such as delay and stop minimization.

Several TSTs tackle the traffic signal timing optimization problem at the network level [19], [18]. These TSTs are fully centralized and, although reliable and robust, do not perform well in highly dynamic traffic conditions [5]. Another class of TSTs aims at finding optimal solutions on isolated intersections. These systems make use of a variety of techniques such as dynamic programming [16], game theory [4], [10], neural networks [24], [9], fuzzy logic [7], [11], [17], and Reinforcement-Learning (RL) [20], [3], [2], [14]. The main drawback of isolated intersection signal planning is the lack of interactions between intersection controllers which leads to sub-optimal solutions at the traffic system-level. A third category of TSTs aims at solving the optimization problem for a subset of intersections [15], [5], [8], [23], [22]. Most systems in this category consider interactions among controllers but limit coordination to neighboring junctions.

The application of the agent paradigm to traffic signal timing has been of interest to Multi-Agent System (MAS) researchers for some time. Distribution, autonomy and coordination are agent properties that are naturally suited for the traffic domain. In this paper, we present a distributed, collaborative multi-agent-based TST that we call DALI (Distributed Agent-based traffic LIghts). In DALI, intersection controllers are augmented with software agents which collaboratively adapt signal timings by considering the feedback of all controller agents that may be affected by a change. The proposed approach is intended to be deployed in the City of Richardson with minimal changes to the traffic infrastructure. As such, our model is based on parameters and data currently used by the city, and does not make assumption on the availability of data not obtainable in the field.

The paper is organized as follows: the next section gives an overview of related works. Section IV presents the agent algorithms and Section VI discusses the experimental results based on a simulation model of the City of Richardson.

II. RELATED WORK

Numerous authors have studied the use of agents in TST, and comprehensive reviews have been published in the literature [6], [26]. In this section we discuss the approaches that compare best to DALI, i.e., the collaborative agent-based TSTs which aim at defining signal timing plans for a subset of intersections. Most published systems in this category use Reinforcement Learning (RL) with Q-learning algorithms [26]. They are either hierarchical or fully distributed.

[5], [1] propose hierarchical models where intersection controller agents are organized in small groups each supervised by a higher level regional agent. Agents at both levels use RL. Communications occur only between agents in a group and their regional agent whose role is to recommend actions. Controller agents make use of local information for their learning process, whereas regional agents use the group's information. Given the large state-action space that needs to be considered by a regional agent, the signal timing problem is simplified in [5]. In order to address the dimensionality problem, [1] proposes the use of a linear function approximation method. Systems in this category have been tested on small simulated grid networks.

In fully Distributed, Collaborative multi-Agent-based (DCA) approaches, controller agents use their local information as well as information learned from their direct neighbors (e.g., state, action, reward, Q-value) via direct or indirect communication to select an action (e.g., shorten or lengthen the split). The actions may result in traffic phases that are out-of-order or even skipped when traffic conditions are unpredictable. MARLIN-ATSC [15] discusses an RL model where agents select actions based on predictions of their neighboring agents' action selection. MARLIN-ATSC was tested on a simulated model of Lower Downtown Toronto comprising 59 intersections. In [23], each controller agent uses Q-learning with either a greedy approach or an exploration strategy using information obtained from its direct neighbors. The approach was tested on a 20 junction network of a simulated network model of Bangalore, India. [21] uses a max-plus RL approach where a controller agent learns about its neighboring agents' locally optimized payoffs rather than the traditional rewards, and then determines the maximum value for the sum of these payoffs. [13] proposes a multi-policy RL-based technique which allows multiple traffic optimization goals simultaneously. The approach was tested on a simulated model of Cork city comprising 6 intersections.

Although the agent-based TSTs in this category have proven to enhance the traffic system performance under nominal conditions, they have limitations: their application to highly dynamic traffic conditions has not been thoroughly evaluated and some argue that the difficulties that may arise with learning speed and performance require that the proposed RL models be extended [26]. In addition, in the proposed DCA systems, agents consider information learned from only their direct neighbors. Moreover, most systems make use of parameters that cannot be easily obtained in the field (e.g., total cumulative delay, queue length, etc.). Finally, the largest realistic simulated network that was built for validation purposes includes 62 intersections [12].

In this paper we present DALI, a fully distributed, collaborative multi-agent based traffic signal timing system for highly dynamic traffic environments. DALI extends existing DCA systems as follows: a) it uses a dynamic, collaborative agent-based strategy to handle unexpected traffic events in real-time; b) agents consider the feedback of all agents that may be affected by a change; c) DALI only considers traffic data readily available in the field; and d) it was validated on a model of the City of Richardson's traffic network comprising 1365 road segments and 128 intersections.

III. 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 6 minor arterial roads and 128 intersections with traffic signals.

The 128 SCATS-based intersection control computers (traffic controllers) are mounted in cabinets at intersections. They 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.

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. Agent Algorithms

In DALI, agents communicate with each other through direct links and do not have a supervising agent to oversee coordination. Agents have knowledge of the traffic network topology. They receive information about the incoming traffic flow from their neighboring controllers and determine the outgoing traffic flow based on the data sensed by their inductive loops (see Figure 1). Intersections are assigned weights to indicate their criticality in the traffic network.

A. Model Definition

Set Definitions

$T = \{t_1, .., t_i\}$ 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. A road segment r_{c_m,c_n} is defined in terms attributes such as length l, speed limit sl and a set of lanes $LN_{r_{c_m,c_n}} = \{ln_1..ln_q\}$.

$LT_{r_{c_m,c_n}.ln_w}$ is the set of lanes that are accessible from $r_{c_m,c_n}.ln_w$

$LF_{r_{c_m,c_n}.ln_w}$ is the set of lanes that have access to $r_{c_m,c_n}.ln_w$

$PH_{c_n} = \{ph_{c_n,1}, ..ph_{c_n,k}\}$ 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, ϵ, the yellow time, ξ, the red time and $LN_{ph_{c_n,k}}$, the set of lanes it applies to.

Function Definitions

$p(r_{c_m,c_n}.ln_w, r_{c_n,c_p}.ln_u)$ 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.

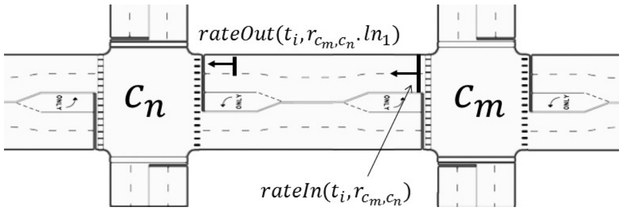

Fig. 1. c_n determines $rateOut$ and receives $rateIn$ from c_m.

$rateOut(r_{c_m,c_n}.ln_w)$ 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 current split time.
$rateIn(t_i, r_{c_m,c_n})$ is the rate of vehicles (per second) that enter road segment r_{c_m,c_n} in the evaluation interval τ that ends at time t_i
$\xi_{t_i,r_{c_m,c_n}.ln_w}$ is the *traffic throughput* 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:

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

B. Algorithms

In this section we give a detailed description of the algorithms executed by the intersection controller agents. Due to space limitation, we restrict our discussion to the main scenario. A detailed discussion of special cases is given in [25].

Algorithm 1 Controller Congestion Reduction

```
Require: $PH_{c_n}, t_i$
 1: for all $ph_{c_n,k} \in PH_{c_n}$ do
 2:     $EvaluateTraffic(ph_{c_n,k}, t_i : TotalInstCong)$
 3:     if $\frac{TotalInstCong}{b} > d$ then
 4:         $GeneratePlan(ph_{c_n,k}, t_i : plan_{new})$
 5:         $RequestForEvaluation(ph_{c_n,k}, plan_{new} : \Psi_{c_n})$
 6:         if $\Psi_{c_n} > h$ then
 7:             $ExecutePlan(plan_{new})$
 8:         end if
 9:     end if
10: end for
11: if $ReceiveRequestForEvaluation(c_p, \kappa_{r_{c_p,c_n}}, \kappa_{ph_{c_q,j}})$ then
12:     $ComputeLevelOfAgreement(\kappa_{r_{c_p,c_n}}, \kappa_{ph_{c_q,j}})$
13: end if
14: if $ReceiveRequestForExecution(c_p, plan_{new})$ then
15:     $AdjustTiming(plan_{new})$
16: end if
```

1) Detecting Congestion: Intersection controller c_n continuously evaluates the traffic state by executing Algorithm 1 to determine if a re-timing operation is necessary. As shown in Figure 1, at each t_i, c_n receives $rateIn$ (detected through its neighbors' 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 $ph_{c_n,k}$.

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

Algorithm 2 Evaluate Traffic

```
Require: $PH_{c_n}, t_i$
 1: for all $ph_{c_n,k} \in PH_{c_n}$ do
 2:     $TotalInstCong \leftarrow 0$
 3:     for $j = 0$ to $b$ do
 4:         $\delta = 0$
 5:         for each $r_{c_m,c_n}.ln_w \in LN_{ph_{c_n,k}}$ do
 6:             $\delta \leftarrow \xi_{ti-j,r_{c_m,c_n}.ln_w} + \delta$
 7:         end for
 8:         $Cong_{t_i,ph_{c_n,k}} \leftarrow \delta$
 9:         if $Cong_{t_i,ph_{c_n,k}} \geq a$ then
10:             $TotalInstCong \leftarrow TotalInstCong + 1$
11:             \* TotalInstCong Represent Sum Over InstantCongestion
12:         end if
13:     end for
14: end for
```

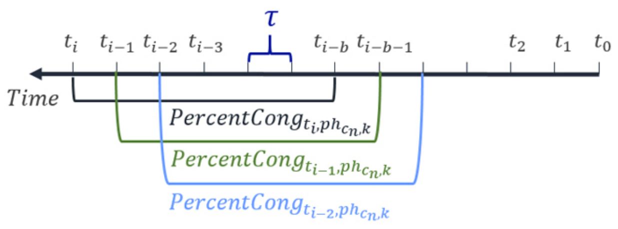
τ
t_i
t_{i-1}
t_{i-2}
t_{i-3}
t_{i-b}
t_{i-b-1}
t_2
t_1
t_0
Time
$PercentCong_{t_i,ph_{c_n,k}}$
$PercentCong_{t_{i-1},ph_{c_n,k}}$
$PercentCong_{t_{i-2},ph_{c_n,k}}$

Fig. 2. $PercentCong$ of phase $ph_{c_n,k}$.

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 $InstantCongestion$ defined as:

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

It proceeds by considering the past b evaluation cycles to determine the percentage of evaluation cycles in which the phase was congested (see Figure 2). 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 the road lanes controlled by $ph_{c_n,k}$ are considered to be congested.

To illustrate the various steps we use the traffic scenario illustrated in Figure 3. c_2 has four incoming roads each with two lanes. The four phases for c_2's intersection are $\{ph_{c_2,1}, ph_{c_2,2}, ph_{c_2,3}, ph_{c_2,4}\}$. These phases apply as follows: $ph_{c_2,1}$ for r_{c_1,c_2}, $ph_{c_2,2}$ for r_{c_4,c_2}, $ph_{c_2,3}$ for r_{c_8,c_2} and $ph_{c_2,4}$ for r_{c_5,c_2}. The phases have the following attribute values. $\gamma = 40$, $\nu = 20$, $\eta = 60$, $\epsilon = 5$, $\xi = 5$. A number of constants have been used in this example. Their values are: $a = 1$, $b = 50$ and $d = 80$. In this example, c_2 evaluates the status of its intersection at the time-stamp t_{6000}. It starts with phase, $ph_{c_2,1}$ and calculates the average traffic throughput $Cong_{t_{6000},ph_{c_2,1}}$ for the set of road lanes that $ph_{c_2,1}$ controls. Given that $rateOut(t_{6000}, r_{c_1,c_2}.ln_1) = 1$,

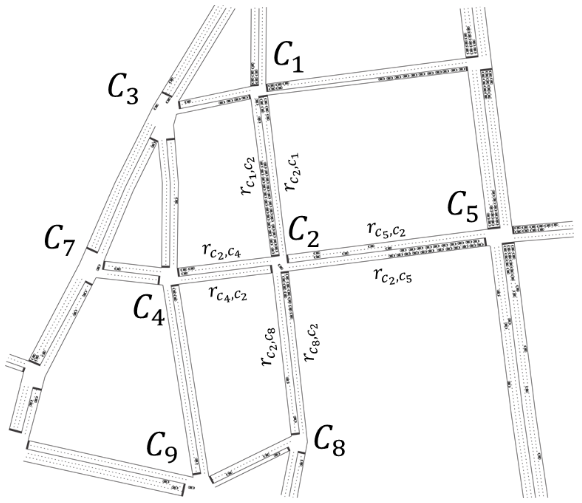

Fig. 3. Overview of the network in case study.

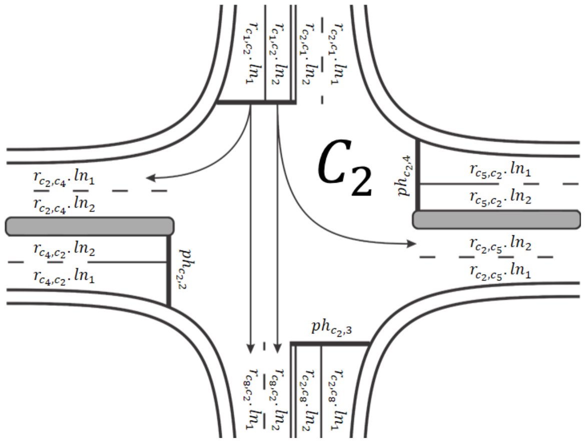

Fig. 4. Intersection assigned to c_2.

$p(r_{c_1,c_2}, r_{c_1,c_2}.ln_1) = 0.8$ and $rateIn(t_{6000}, r_{c_1,c_2}) = 2.4$, the value of $\xi_{t_{6000},r_{c_1,c_2}.ln_1}$ is:

$$\xi_{t_{6000},r_{c_1,c_2}.ln_1} = \frac{2.4\times 0.8}{1} = 1.92$$

For the sake of illustration, we assume that $Cong_{t_{6000},ph_{c_2,1}} = 1.31$ which is greater than the threshold $a = 1$. c_2, then retrieve the calculated values of $Cong$ between the time stamps t_{5999} and t_{5950}. We find that 43 of them are greater than a. Therefore,

$$PercentCong_{t_{6000},ph_{c_2,1}} = \frac{43}{50} \times 100 > 80$$

Consequently, c_2 detects a congestion on phase $ph_{c_2,1}$ and deliberates to define a new plan.

2) Generate New Plan: The controller deliberates to determine the value of a new split that will alleviate congestion on $ph_{c_n,k}$. This is achieved in step 7 in Algorithm 3. Value of the new split is calculated as:

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

e and f are coefficients that can be calibrated. They regulate the influence of the traffic throughput and the current split time for the new split time. Values of cycle length and offset change in the new split. If $plan_{new}.phase.\gamma$ is greater than the maximum allowed split time γ_{MAX} defined for phase $ph_{c_n,k}$ as:

Algorithm 3 Generate Plan

```
Require: $ph_{c_n,k}, t_i$
Ensure: $plan_{new}$
 1: $plan_{new}.phase \leftarrow ph_{c_n,k}$
 2: $\chi \leftarrow 0$
 3: for $j = i - \nu$ to $i$ do
 4:     $\chi \leftarrow \chi + Cong_{t_j,ph_{c_n,k}}$
 5: end for
 6: $\chi \leftarrow \frac{\chi}{\nu}$
 7: $plan_{new}.phase.\gamma \leftarrow plan_{cur}.phase.\gamma * (e + \chi * f)$
 8: if $plan_{new}.phase.\gamma > ph_{c_n,k}.\gamma_{MAX}$ then
 9:     $plan_{new}.phase.\gamma \leftarrow ph_{c_n,k}.\gamma_{MAX}$
10: end if
```

$$ph_{c_n,k}.\gamma_{MAX} = ph_{c_n,k}.\eta + ph_{c_n,k}.\epsilon + ph_{c_n,k}.\xi$$

then its value is set to $ph_{c_n,k}.\gamma_{MAX}$ (step 9).

In the example above, the average of $Cong$ for phase $ph_{c_2,1}$ in the last $\nu = 20$ evaluation cycle is 1.23. Given that $e = 1$ and $f = 0.2$, c_2 defines a new plan for $ph_{c_2,1}$, and computes $plan_{new}.phase.\gamma$ as:

$$plan_{new}.phase.\gamma = 40 \times (1 + 1.23 \times 0.2) \approx 50$$

Therefore, c_2 determines that it needs to increase $ph_{c_2,1}.\gamma$ by ten seconds.

3) Request For Evaluation: c_n determines the impact of executing the new plan on the 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:

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

$\kappa_{ph_{c_n,k}}$ 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 (Algorithm 4, Step 3). In the same way, $\kappa_{r_{c_n,c_p}}$ for a road segment r_{c_n,c_p}, is the sum of $\kappa_{r_{c_n,c_p}.ln_w}$ (Algorithm 4, Step 10). 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. $\kappa_{ph_{c_n,k}}$ corresponds to the increment in the rate of vehicles that exits the road lanes controlled by $ph_{c_n,k}$, in case the new plan is to be executed. $\kappa_{r_{c_n,c_p}}$ corresponds to the portion of $\kappa_{ph_{c_n,k}}$ that goes to road segment r_{c_n,c_p}. In the illustrative example, c_2 calculates $\kappa_{r_{c_1,c_2}.ln_1}$ as:

$$\kappa_{r_{c_1,c_2}.ln_1} = \frac{1 \times (50 - 40)}{50} = 0.25$$

Having $\kappa_{r_{c_1,c_2}.ln_2} = 0.3$, $\kappa_{ph_{c_2,1}}$ takes the value of 0.55. c_2 then calculates the effect of executing a new plan on its neighboring intersections including c_4. Assuming $p(r_{c_1,c_2}.ln_1, r_{c_2,c_4}.ln_1) = 0.7$, $p(r_{c_1,c_2}.ln_1, r_{c_2,c_4}.ln_2) = 0$, $p(r_{c_1,c_2}.ln_2, r_{c_2,c_4}.ln_1) = 0.2$

and $p(r_{c_1,c_2}.ln_2, r_{c_2,c_4}.ln_2) = 0$, $\kappa_{r_{c_2,c_4}}$ will be calculated as:

$$\kappa_{r_{c_2,c_4}} = 0.25 \times 0.7 + 0.25 \times 0 + 0.3 \times 0.2 + 0.3 \times 0 = 0.32$$

c_2 then sends a request for evaluation to c_4 with $\kappa_{r_{c_2,c_4}} = 0.32$ and $\kappa_{ph_{c_2,1}} = 0.55$. This means that by executing $plan_{new}$, an additional 0.55 vehicle per seconds (vps) will leave $ph_{c_2,1}$, and out of the 0.55 (vps), 0.32 (vps) will enter r_{c_2,c_4}.

Algorithm 4 Request for Evaluation

```
Require: $ph_{c_n,k}, plan_{new}$
Ensure: $\Psi_{c_n}$
 1: $\kappa_{ph_{c_n,k}} \leftarrow 0$
 2: for each $r_{c_m,c_n}.ln_w$ in $LN_{ph_{c_n,k}}$ do
 3:     $\kappa_{ph_{c_n,k}} \leftarrow \kappa_{ph_{c_n,k}} + \kappa_{r_{c_m,c_n}.ln_w}$
 4: end for
 5: $\Psi_{c_n} \leftarrow 0$
 6: for each accessible neighbor $c_p$ , in parallel do
 7:     $\kappa_{r_{c_n,c_p}} \leftarrow 0$
 8:     for $r_{c_m,c_n}.ln_w \in LN_{ph_{c_n,k}}$ do
 9:         for $r_{c_n,c_p}.ln_u \in LT_{r_{c_m,c_n}.ln_w}$ do
10:             $\kappa_{r_{c_n,c_p}} \leftarrow \kappa_{r_{c_n,c_p}} + (p(r_{c_m,c_n}.ln_w, r_{c_n,c_p}.ln_u) \times \kappa_{r_{c_m,c_n}.ln_w})$
11:         end for
12:     end for
13:     Send($c_p$, $\kappa_{r_{c_n,c_p}}$, $\kappa_{ph_{c_n,k}}$)
14:     Receive($c_p$, $\Psi_{c_p}$)
15:     $\Psi_{c_n} \leftarrow \Psi_{c_n} + \Psi_{c_p}$
16: end for
```

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 controllers c_q and request that they each 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. An intermediate controller, c_p, calculates Ψ_{c_p} based on the existing traffic throughput, its priority ω and the ratio of the received additional vehicle throughput. x, y and z are coefficients that calibrate the influence of variables in Ψ. After receiving the level of agreement from all affected neighbors, c_p adds them to 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 opinion of all affected controllers in the network.

In the illustrative example, given that $rateOut(t_{6000}, r_{c_2,c_4}.ln_1) = 1$, $rateOut(t_{6000}, r_{c_2,c_4}.ln_2) = 0.3$, $rateIn(t_{6000}, r_{c_2,c_4}) = 1.2$, $p(r_{c_2,c_4}, r_{c_2,c_4}.ln_1) = 0.8$ and $p(r_{c_2,c_4}, r_{c_2,c_4}.ln_2) = 0.2$, the value of Ψ_{c_4} is calculated as:

Algorithm 5 Compute Level Of Agreement

```
Require: $\kappa_{r_{c_n,c_p}}, \kappa_{ph_{c_n,k}}$
Ensure: $\Psi_{c_p}$
 1: $\Psi_{c_p} \leftarrow 0$
 2: for $r_{c_n,c_p}.ln_u \in LN_{r_{c_n,c_p}}$ do
 3:     $\Psi_{c_p} \leftarrow \Psi_{c_p} + x \times \omega(c_p) \times \frac{\kappa_{r_{c_n,c_p}}}{\kappa_{ph_{c_n,k}}} \times (y - z \times \frac{(\kappa_{r_{c_n,c_p}} + rateIn(t_i, r_{c_n,c_p})\ \times p(r_{c_n,c_p}, r_{c_n,c_p}.ln_u)}{rateOut(t_i, r_{c_n,c_p}.ln_u)})$
 4: end for
 5: for each accessible neighbor $c_q$ from $c_p$ , in parallel do
 6:     $\kappa_{r_{c_p,c_q}} \leftarrow 0$
 7:     for $r_{c_n,c_p}.ln_u \in LN_{r_{c_n,c_p}}$ do
 8:         for $r_{c_p,c_q}.ln_f \in LF_{r_{c_n,c_p}.ln_u}$ do
 9:             $\kappa_{r_{c_n,c_p}} \leftarrow \kappa_{r_{c_n,c_p}} + p(r_{c_n,c_p}, r_{c_n,c_p}.ln_u) \times p(r_{c_n,c_p}.ln_u, r_{c_p,c_q}.ln_f) \times \kappa_{r_{c_n,c_p}}$
10:         end for
11:     end for
12:     if $\kappa_{r_{c_n,c_p}} > g$ then
13:         Send($c_q$, $\kappa_{r_{c_n,c_p}}$, $\kappa_{ph_{c_n,k}}$)
14:         Receive($c_q$, $\Psi_{c_q}$)
15:         $\Psi_{c_p} \leftarrow \Psi_{c_p} + \Psi_{c_q}$
16:     end if
17: end for
18: Send($c_n$, $\Psi_{c_p}$)
```

$$\Psi_{c_4} = 1.0 \times 2.0 \times \frac{0.32}{0.55} \times ((1 - 1 \times \frac{(0.32 + 1.2) \times 0.8}{1}) + (1 - 1 \times \frac{(0.32 + 1.2) \times 0.2}{0.3})) = -0.2$$

c_4 calculates κ for c_3, c_7 and c_9 and ask them to evaluate the plan if κ is greater than threshold g. The result is added to Ψ_{c_4} and sent back to c_2. Upon receipt of Ψ_{c_4}, Ψ_{c_5} and Ψ_{c_8}, controller c_2 calculates Ψ_2. Given that $h = 0$, negative values of Ψ are considered as a level of disagreement (Algorithm 1,Step 6). Having $\Psi_2 = 2.34$, c_2 executes the new plan and announces the execution to all controllers 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 (ATS).

MATISSE consists of three modules. The main constituent is the MATISSE Simulation Module which includes 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; 2) the Intersection Control Agent Platform creates and manages intersection controller agents; and 3) the Zone Manager Platform creates

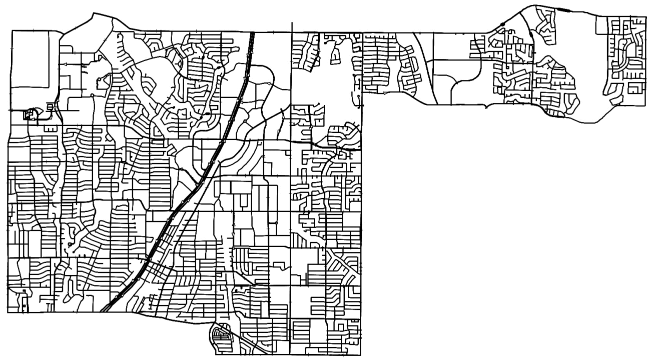

Fig. 5. 2D visualization of Richardson's Traffic Network.

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

and manages service and traffic manager agents. Communications between simulated agents within and across platforms take place over the Agent Message Transport Service. The Environment System maintains a detailed specification of the traffic network topology. The Control and Visualization Module 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 taken directly from Open Street Map. It uses advanced algorithms to automatically generate missing information such as unknown road types or traffic light locations. During the simulation, vehicles enter and exit the simulation from entry and exit points. The user can let MATISSE generate an initial normal distribution or define their own vehicle distribution for preferred entry and exit points.

Agents are equipped with sensors. They perceive and communicate with agents located within their sensor range, i.e., *circle-of-influence* for vehicle agents.

During the simulation, the user can modify the driver's level of distraction. This may dynamically introduce unexpected accidents and unpredicted traffic behavior.

VI. Experimental Results

In this section, we discuss the evaluation of DALI with respect to delay. Similar results were obtained for queue length [25].

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 5 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.

Three simulation settings were run eight times for 86,400 simulation cycles representing a 24-hour time period. The average delay for all vehicles was measured. In the first and second experiment, we use real-world data provided by the City of Richardson to simulate regular traffic patterns with and without accidents. In the third and fourth experiment we simulate continuous random traffic patterns with and without accidents. For all experiments, we compare the efficiency of DALI with the SCATS-based system currently in use in Richardson (SCATS-R), and a model of the RL-based MARLIN-ATSC [15] (MARLIN-R). To decrease the learning time of MARLIN agents, we initialized the Q-values based on estimations derived from historical data provided by the City of Richardson.

Experiment 1: Normal Traffic Conditions
In this experiment, we make use of the traffic data provided by the City of Richardson to determine the number of vehicles in the traffic network at any given time, as well as their distribution in the network. This experiment is intended to analyze the behavior of the three systems under nominal traffic conditions.

As shown in Figure 6, 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. MARLIN-R agents perform better (53% delay reduction) in this situation 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)

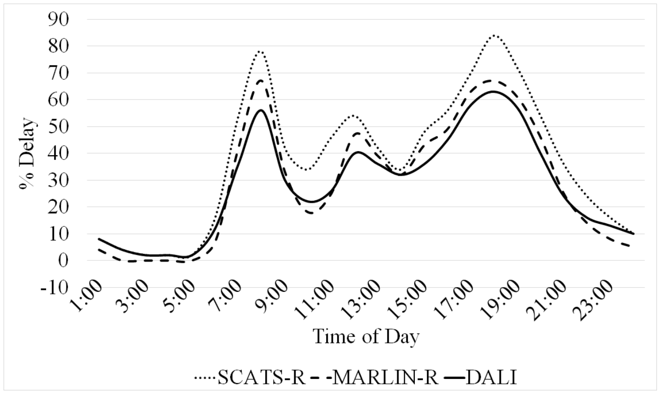

Fig. 6. Average delay using traffic data from the City of Richardson

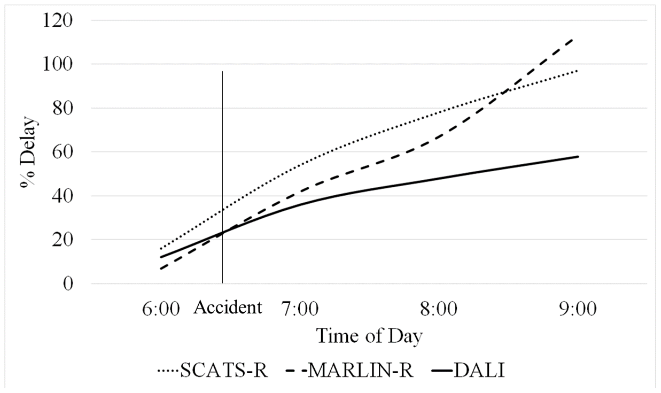

Fig. 7. Average delay with accident in peak morning hours using real traffic data

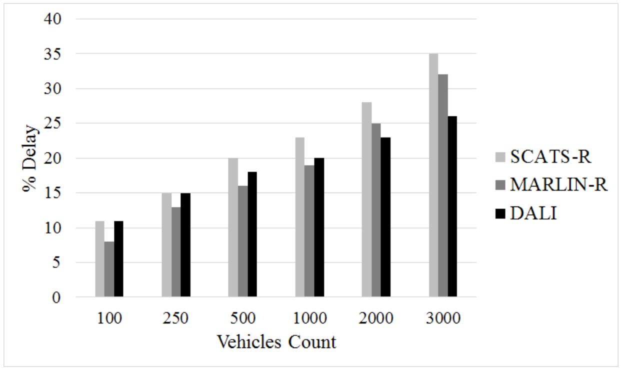

Fig. 8. Average delay for random traffic patterns

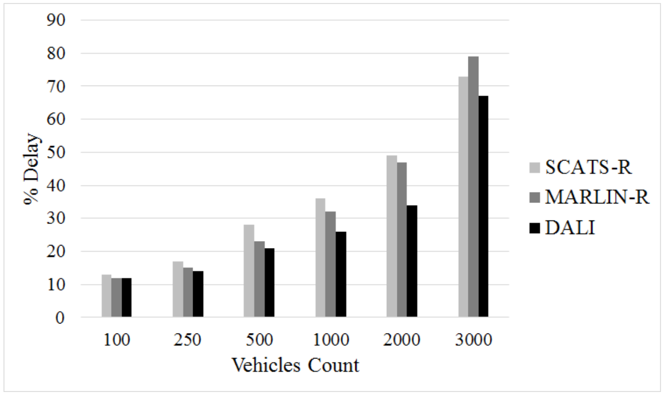

Fig. 9. Average delay for random traffic patterns with accidents.

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). MARLIN-R performs slightly less than DALI. 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.

Experiment 2: Normal Traffic Conditions With Accident
Figure (7) shows the performance of the systems when an accident is triggered at run time, during normal morning peak traffic. As expected, DALI handles the traffic much better than SCATS-R (35% delay reduction). It is notable that MARLIN-R agents are unable to control the congestion created by the accident since they have no prior knowledge of the unexpected traffic pattern. Similarly to Experiment 1, the simulation shows that, rather than leading the vehicles towards roads with lighter traffic, MARLIN-R agents send vehicles to congested areas.

Experiment 3: Continuous Random Traffic Conditions
In this experiment, the number of vehicles during the simulation remains constant but new vehicles are added randomly while others randomly exit the traffic network. This experiment is intended to illustrate random traffic patterns that are unprecedented. The experiment was run with 100, 250, 500, 1000, 2000 and 3000 vehicles.

Figure (7) shows that when the traffic is light, MARLIN-R agents perform (37%) better because they account for a variable phasing sequence. They can extend the current phase or switch to any other phase according to the changes in traffic. On the other hand, SCATS-R controllers and DALI agents execute a fixed phase sequence. Therefore, all phases are executed even in cases where it is not necessary. DALI and SCATS-R perform at the same level in lighter traffic conditions because the controller agents do not detect congestion and therefore, do not change the split. As the number of vehicles increases, DALI agents start to detect congestion and collaborate with other agents for retiming. The collaborative retiming procedure allows DALI to perform better than SCATS. As the number of vehicles increases, MARLIN-R still perform better than SCATS-R. However, DALI do better. This is due to the fact that MARLIN-R agents fail to handle heavy traffic in small, condensed network areas.

Experiment 4: Continuous Random Traffic Conditions with Accident
Figure (9) shows the performance of DALI, SCATS-R and MARLIN-R in the extreme situation where an accident is randomly triggered in unpredictable traffic conditions. When the traffic is light, the three systems nearly act the same. As traffic gets heavier, DALI operates better than the other two (20% decrease in delay compared to SCATS-R and 12% decrease in delay compared to MARLIN-R). When the number of vehicles reaches 3000, MARLIN-R operates worse that SCATS-R (8% delay increase) because SCATS-R controllers are committed to giving green signal to all movements in a cycle whereas MARLIN-R agents lack experience in dealing with new traffic conditions.

VII. Conclusion

In this paper we presented DALI, a distributed collaborative multi-agent traffic signal timing (TST) system for highly dynamic traffic conditions. DALI has been validated on a simulated model of City of Richardson's traffic network. The experimental results show that the collaborative multi-agent controllers outperforms the traditional SCATS-based system currently used by the City of Richardson. While a simulated model of MARLIN performs better than DALI in stable traffic conditions with light to normal traffic, the RL-based model does not operate efficiently in two settings: 1) random traffic conditions and 2) nominal traffic conditions with heavy traffic in condensed traffic network areas. Our goal is to investigate the development of a hybrid model that will integrate some RL in the DALI agents.

This work is a first step towards the implementation of an agent-based TST for the City of Richardson. 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] Monireh Abdoos, Nasser Mozayani, and Ana LC Bazzan. Holonic multi-agent system for traffic signals control. *Engineering Applications of Artificial Intelligence*, 26(5-6):1575–1587, 2013.

[2] 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.

[3] Sahar Araghi, Abbas Khosravi, and Douglas Creighton. Intelligent cuckoo search optimized traffic signal controllers for multi-intersection network. *Expert Systems with Applications*, 42(9):4422–4431, 2015.

[4] Ana LC Bazzan. A distributed approach for coordination of traffic signal agents. *Autonomous Agents and Multi-Agent Systems*, 10(1):131–164, 2005.

[5] 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.

[6] Ana LC Bazzan and Franziska Klügl. A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review*, 29(3):375–403, 2014.

[7] Yunrui Bi, Dipti Srinivasan, Xiaobo Lu, Zhe Sun, and Weili Zeng. Type-2 fuzzy multi-intersection traffic signal control with differential evolution optimization. *Expert Systems with Applications*, 41(16):7338–7349, 2014.

[8] Vinny Cahill et al. Soilse: A decentralized approach to optimization of fluctuating urban traffic using reinforcement learning. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 531–538. IEEE, 2010.

[9] Kuei-Hsiang Chao, Ren-Hao Lee, and Meng-Hui Wang. An intelligent traffic light control based on extension neural network. In *Knowledge-based intelligent information and engineering systems*, pages 17–24. Springer, 2008.

[10] 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.

[11] 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.

[12] Ivana Dusparic and Vinny Cahill. Autonomic multi-policy optimization in pervasive systems: Overview and evaluation. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 7(1):11, 2012.

[13] Ivana Dusparic, Julien Monteil, and Vinny Cahill. Towards autonomic urban traffic control with collaborative multi-policy reinforcement learning. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, pages 2065–2070. IEEE, 2016.

[14] 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.

[15] 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.

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

[17] Iisakki Kosonen. Multi-agent fuzzy signal control based on real-time simulation. *Transportation Research Part C: Emerging Technologies*, 11(5):389–403, 2003.

[18] UK's Transport Research Laboratory. Split cycle and offset optimisation technique. https://trlsoftware.co.uk/products/traffic_control/scoot. Accessed April 2018.

[19] UK's Transport Research Laboratory. Traffic network and isolated intersection study tool. https://trlsoftware.co.uk/products/junction_signal_design/transyt. Accessed April 2018.

[20] Patrick Mannion, Jim Duggan, and Enda Howley. Learning traffic signal control with advice. In *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS 2015)*, 2015.

[21] Juan C Medina and Rahim F Benekohal. Traffic signal control using reinforcement learning and the max-plus algorithm as a coordinating strategy. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 596–601. IEEE, 2012.

[22] Tong Thanh Pham, Tim Brys, Matthew E Taylor, Tim Brys, Madalina M Drugan, PA Bosman, Martine-De Cock, Cosmin Lazar, L Demarchi, David Steenhoff, et al. Learning coordinated traffic light control. In *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS-13)*, volume 10, pages 1196–1201. IEEE, 2013.

[23] KJ Prabuchandran, Hemanth Kumar AN, and Shalabh Bhatnagar. Multi-agent reinforcement learning for traffic signal control. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 2529–2534. IEEE, 2014.

[24] 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.

[25] Behnam Torabi. A self-organizing traffic management system and its real-world implementation, ph.d. proposal. Technical report, University of Texas at Dallas, October 2017.

[26] 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.