Hierarchical-distributed optimized coordination of intersection traffic

Pavankumar Tallapragada Jorge Cortés

***Abstract*—This paper considers the problem of coordinating vehicular traffic at an intersection and on the branches leading to it in order to minimize a combination of total travel time and energy consumption. We propose a provably safe hierarchical-distributed solution to balance computational complexity and optimality of the system operation. In our design, a central intersection manager communicates with vehicles heading towards the intersection, groups them into clusters (termed bubbles) as they appear, and determines an optimal schedule of passage through the intersection for each bubble. The vehicles in each bubble receive their schedule and implement local distributed control to ensure system-wide inter-vehicular safety while respecting speed and acceleration limits, conforming to the assigned schedule, and seeking to optimize their individual trajectories. Our analysis rigorously establishes that the different aspects of the hierarchical design operate in concert and that the safety specifications are satisfied. We illustrate its execution through a suite of simulations and compare its performance against optimized signal-based coordination over a wide range of system parameters.**

***Index Terms*—Intelligent transportation systems, hierarchical and distributed control, optimized operation and scheduling, state-based intersection management, networked vehicles**

I. Introduction

With rapidly growing urbanization and mobility needs of people across the world, existing transportation systems are in critical need of transformation. Apart from increased travel times, current burdened transportation systems have the side effects of increased pollution, increased energy consumption, and degradation of people's health, all of which have an immeasurable cost on society. The complexity of the challenge requires a multi-pronged approach, one of which is the development of new technologies. Emerging technologies such as vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication, and computer-controlled vehicles offer the opportunity to radically redesign our transportation systems, eliminating road accidents and traffic collisions and positively impacting safety, traveling ease, travel time, and energy consumption.

A particularly useful application of these technologies is the coordination of traffic at and near intersections for a smoother (with reduced stop-and-go) and fuel-efficient traffic flow. An intersection manager with knowledge of the state of the traffic could schedule the intersection crossings of the vehicles. With the assigned schedule, individual vehicles could further optimize their travel to the intersection in a fuel-efficient way. In contrast to traditional intersection management, networked vehicle technologies allow us to coordinate the traffic not just *within the intersection*, but also by controlling the vehicles' behavior much before they arrive at the intersection. Such a paradigm offers the possibility of significantly reduced stop times and increased fuel efficiency, and is the subject of this paper. This idea is not only applicable to intersection traffic management in cities but also to traffic management of automated vehicles/robots in warehouses, sea and airports. Here, our focus is on a single intersection, leaving for future work the design of efficient strategies for coordinated traffic management in networks of intersections.

Literature review: Much of the literature in the area of coordination-based intersection management focuses on collision avoidance of vehicles *within the intersection*. Supervisory intersection management (intervention only when required to maintain safety by avoiding collisions) is explored using discrete event abstractions in [2], [3] and reachable set computations in [4], [5]. On the other hand, [6] designs supervisory control for safe usage of an intersection while seeking to minimize the deviation of the supervisory control from a human driver's intent, while [7] proposes a least-restrictive supervisory control for multiple intersections. The works [8], [9] and references therein describe a multiagent simulation approach in which, upon a reservation request from a vehicle, an intersection manager accepts or rejects the reservation based on a simulation. Each vehicle attempts to conform to its assigned reservation and, if this is predicted not to be possible at any time, the reservation is canceled. [10] also uses a reservation-based system to schedule intersection crossing times and provides provably safe maneuvers for vehicle following in a lane as well as for crossing the intersection. [11], [12] use a method based on model predictive control to coordinate the intersection crossing by vehicles and obtain suboptimal solutions to a linear quadratic optimal control problem. [13] also proposes a model predictive control approach in which collision-free intersection crossing by vehicles is achieved through a combination of hard no-collision constraints as well as a soft constraint in the form of a term measuring collision risk in the cost function. [14], [15] formulate the problem of traffic coordination in the intersection as a mixed integer program with non-collision separation constraints. In [16], a heuristic policy assigns priorities to the vehicles, while each vehicle applies a priority-preserving control and legacy vehicles platoon behind a computer-controlled car. In this context, we note that the ability to efficiently coordinate diminishes as the vehicles get closer to the intersection. This is why here

A preliminary version of this work appeared as [1] at the 5th IFAC Workshop on Distributed Estimation and Control in Networked Systems.

Pavankumar Tallapragada is with the Department of Electrical Engineering, Indian Institute of Science, Bengaluru, and Jorge Cortés is with the Department of Mechanical and Aerospace Engineering, University of California, San Diego pavant@iisc.ac.in, cortes@ucsd.edu

we take an expanded view of intersection management that looks at the coordinated control of the vehicles much before they arrive at the intersection. The methods above are not suited for this setup or would prove to be too computationally costly in such scenarios. An example of the expanded view of intersection management is [17], [18], in which a polling-systems approach is adopted to assign schedules, and then optimal trajectories for all vehicles are computed sequentially in order. Such optimal trajectory computations are costly and depend on other vehicles' detailed plans, and hence the system is not robust. The work [19] assumes that the order in which the vehicles must use the intersection is given. The proposed design first finds throughput maximizing intersection usage times, which are then used to compute energy minimizing control input trajectories for each vehicle in a decoupled manner. Closer to our paper, the works [20], [21] describe a hierarchical setup, with a central coordinator verifying and assigning reservations, and with vehicles planning their trajectories locally to platoon and to meet the assigned schedule. The proposed solution is based on multiagent simulations and a reservation-based scheduling (with the evolution of the vehicles possibly forcing revisions to the schedule), both important differences with respect to our approach. [22] utilizes the ideas of classical queuing theory and explores a slot-based intersection management - wherein the vehicles are allotted a time slot for using the intersection and the vehicles in turn control their speed to use the intersection only within that time slot. [23] is a recent survey of traffic control with vehicular networks and provides other related references.

Statement of contributions: We propose a provably safe intersection management system aimed at optimizing a combination of cumulative travel time and fuel usage for all the vehicles. Our first contribution is the idea of coordinating the traffic at the intersection and on the branches leading to it in a unified, holistic way. The basic observation is that planning and controlling the vehicles from much before they arrive at the intersection should lead to better overall coordination and efficiency.

Our second set of contributions is a multi-layered design that combines hierarchical and distributed control and is applicable to a wide range of traffic conditions. Our hierarchical-distributed approach offers a balance between computational complexity of the solution and optimal operation. The proposed system is composed of three main elements: (i) clustering to identify vehicles that platoon before arriving at the intersection. We refer to such clusters of vehicles as *bubbles*. We use the term bubble, rather than platoon, to emphasize the dynamic, time-varying nature of the cohesiveness of the group of vehicles as they travel towards the intersection. With this terminology, a bubble becomes a cohesive group (i.e., a platoon) by the time it crosses the intersection; (ii) a branch-and-bound scheduling algorithm that, using aggregate information about the bubbles, allows a central intersection manager to find the optimal schedule of bubble passage; and (iii) a distributed control algorithm for the vehicles at the local level. This control policy ensures that the vehicles of each bubble platoon into a cohesive group when they cross the intersection and that each bubble conforms to the schedule prescribed by the intersection manager, while guaranteeing system-wide safety subject to speed limits and acceleration saturation. Additionally, each vehicle seeks to optimally control its trajectory whenever safety is not immediately threatened. The first two aspects are contributions of this paper while the aspect of local vehicular control incorporates the algorithmic solution from our previous work [24].

Our third and final contribution is the technical analysis leading to the provable safety of our design. In contrast to computationally intensive multiagent simulation-based methods, we provide analytical guarantees on correctness, safety, and performance. Further, the results provide good intuition and fundamental and reliable principles for future designs. We do acknowledge that the development of analytical guarantees comes at the cost of some conservatism in the design. We have performed a suite of simulations comparing our approach to optimized signal-based coordination that show a significant improvement in the cumulative energy consumption for a wide range of traffic densities and a more socially equitable distribution of cost. As a final note for the reader's sake, we have made every effort in the presentation to make the components of the paper understandable even if the proofs of the technical results are skipped in a first reading[1].

II. Problem Statement

Consider an intersection with eight incoming traffic lanes, or branches, labeled by $\{1,\dots,8\}$, cf. Figure 1. There are two

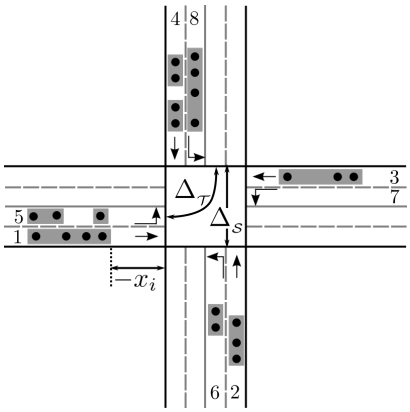

Fig. 1. Traffic near an intersection. $\Delta_{\mathcal{S}}$ is the length of the intersection, which must be covered by the vehicles going straight while the vehicles turning left must travel along a curve of length $\Delta_{\mathcal{T}}$. The numbers $\{1,\dots,8\}$ are labels for the incoming lanes, or branches. Black dots represent individual vehicles, which are clustered and contained within *bubbles*, represented by grey boxes.

branches in each incoming direction, one for going straight and the other for turning left. We implicitly assume that the

[1]Throughout the paper, we use the following notation. We let $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{N}_0$ denote the set of real, nonnegative real, integer, positive integer, and nonnegative integer numbers, respectively. For a non-empty ordered list $\mathcal{S} = \{j_1,\dots,j_s\}$, we let $|\mathcal{S}|$ denote the cardinality of $\mathcal{S}$. Further, $\mathcal{S}(i)$ denotes the i^{th} element j_i of $\mathcal{S}$. Thus, $\mathcal{S}(|\mathcal{S}|)$ denotes the last element of $\mathcal{S}$. For convenience, we also use the notation $j \in \mathcal{S}$ ($j \notin \mathcal{S}$) to denote that j is (is not) an element of $\mathcal{S}$. For two ordered lists $\mathcal{S}_1$ and $\mathcal{S}_2$, we let $\mathcal{S}_1 \setminus \mathcal{S}_2$ denote the ordered list of elements that belong to $\mathcal{S}_1$ but not to $\mathcal{S}_2$, while preserving the same order of $\mathcal{S}_1$. Given $u_m \leq u_M$, $[u]_{u_m}^{u_M}$ denotes the number u lower and upper saturated by u_m and u_M respectively, i.e., $[u]_{u_m}^{u_M} = u_m$ if $u \leq u_m$, $[u]_{u_m}^{u_M} = u$ if $u \in [u_m, u_M]$, and $[u]_{u_m}^{u_M} = u_M$ if $u \geq u_M$.

vehicles that need to turn right do so during the time interval in which the vehicles from the branch also go straight. This is a reasonable assumption since the time to turn right is typically smaller than the time to cross the intersection. For simplicity, we assume that (i) all vehicles are identical with length L, (ii) vehicles do not change lanes along the portion of the branches we consider, and (iii) there are no sources or sinks for vehicles along the branches (all new traffic appears at the beginning of the branches and must cross the intersection). We discuss later in Remark III.1 the extent to which these assumptions can be relaxed in our algorithmic solution. The intersection is a square with a side length of Δ_S. The vehicles turning left at the intersection traverse along a curve of length Δ_T. We say a pair of incoming branches are *compatible branches* if the vehicles coming from them can use the intersection simultaneously with guaranteed safety. Every other pair of branches are *non-compatible*. Vehicles from non-compatible branches may not use the intersection simultaneously. It is easy to see that the set of compatible branches is the set of unordered pairs

$$\begin{aligned}\mathcal{B} \triangleq \{&\{1,3\},\{2,4\},\{1,5\},\{2,6\},\\ &\{3,7\},\{4,8\},\{5,7\},\{6,8\}\}.\end{aligned} \tag{1}$$

The dynamics of each vehicle k is a fully actuated second-order system,

$$\dot{x}_k^v(t) = v_k^v(t), \tag{2a}$$
$$\dot{v}_k^v(t) = u_k^v(t), \tag{2b}$$

where $x_k^v,\ v_k^v \in \mathbb{R}$ are the position (negative of the distance from the front of the vehicle to the beginning of the intersection) and velocity of the vehicle, respectively and $u_k^v(t) \in [u_m, u_M]$, with $u_m \leq 0 \leq u_M$, is the input acceleration. We use the superscript v to emphasize that the state and control variables refer to individual vehicles. We assume that each branch has a maximum speed limit that the vehicles must respect. For the sake of easing the notation, we assume that the speed limit on all branches is the same and equals v^M. Thus, for each vehicle k, $v_k^v(t)$ must belong to the interval $[0, v^M]$ for all time t that the vehicle is in the system.

Each vehicle is equipped with vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication capabilities. With V2I communication, the vehicles inform a central *intersection manager* (IM) about their positions and velocities and receive from it commands such as prescribed time of arrival at the intersection. We assume the IM has the necessary communication and computing capabilities. We seek a design solution that aims to minimize a cost function that models a combination of cumulative travel time and cumulative fuel cost of the form

$$\sum_k \int_{t_k^{\text{spawn}}}^{T_k^{\text{exit}}} (W_T + |u_k^v|)\mathrm{d}t, \tag{3}$$

where k is the vehicle index, t_k^{spawn} is the time at which vehicle k 'spawns' into the problem domain and T_k^{exit} is the time at which the vehicle exits the intersection, i.e., $x_k^v(T_k^{\text{exit}}) = \Delta_S + L$ or $x_k^v(T_k^{\text{exit}}) = \Delta_T + L$ depending on whether vehicle k goes straight or turns left at the intersection. The weight W_T sets the relative importance of travel time versus fuel cost. Note that here we are using acceleration as a stand-in for fuel consumption, when in reality vehicle fuel consumption depends on many factors, including idling, accessory power draw, and engine efficiency. The vehicles over which the cost is summed may be chosen in different ways - for example it may be over all vehicles that cross the intersection in a time period or it may be over a fixed number of vehicles. The constraints in the problem arise from the speed limit, bounds on vehicle acceleration and deceleration, and the safety requirements - which require scheduling the intersection crossing of the vehicles and maintenance of safe distance between the vehicles. Solving this problem at the level of individual vehicles is computationally expensive and not scalable. Thus, we aim to synthesize a solution that makes this problem tractable to solve in real time and is applicable to a wide range of traffic scenarios.

III. Overview of Hierarchical Distributed Solution

This section gives an outline of our hierarchical distributed solution to the problem stated in Section II. Our algorithmic solution combines optimized planning and scheduling of groups of vehicles with local distributed control to ensure safety and execute the plans. Its three distinct aspects are:

(i) grouping the vehicles into clusters,
(ii) scheduling the passage of the clusters through the intersection,
(iii) local vehicular control to achieve and maintain cluster cohesion, avoid collisions, and ensure the clusters meet the prescribed schedule.

Each of these aspects is coupled with the other two. Moreover, an overarching theme is the dynamic nature of the problem due to the arrival and departure of vehicles. Any complete or partial solution has to be computed as new vehicles come in (event based) or at regular time intervals (time based). In what follows, we provide a general description of the main ingredients of each aspect. At any given time t, we let t_s be the latest time prior to t at which the IM samples the state of traffic and solves the corresponding static scheduling problem.

Aspect 1 – generation of bubbles: The primary motivation to cluster vehicles is to reduce the number of independent entities that need to be considered in the (computationally expensive) schedule optimization problem. For instance, the maximum number of clusters can be fixed according to the available computational resources so that the scheduling problem remains tractable. At time t_s, the vehicles present in the incoming branches are grouped into N clusters. We let N_p denote the number of clusters on branch p. Given the position information of the vehicles at t_s, we use n-means clustering on each branch individually to identify the clusters. The relative positions of the vehicles of a cluster may vary significantly over the course of their travel and the vehicles may not be in the form of a well-defined platoon at all times. Hence, we refer to a cluster of vehicles as a *bubble* (shown as grey boxes in Figure 1). The defining characteristic of a bubble is that all

the vehicles of a bubble cross the intersection together. The state of the i^{th} bubble is given by the tuple

$$\xi_i = (x_i, v_i, m_i, \bar{\tau}_i^{\text{occ}}, \mathcal{I}_i) \in \mathbb{R}^4 \times \{1, \ldots, 8\},$$

where x_i, v_i and m_i are, respectively, the position of the lead vehicle in the bubble, the velocity of the lead vehicle in the bubble, and the number of vehicles in the bubble. We denote by τ_i^{occ}, the *occupancy time* of bubble i, which is the time duration for which the intersection is occupied by bubble i. The quantity $\bar{\tau}_i^{\text{occ}}$ is an upper bound on the occupancy time that can be guaranteed *a priori*, and is a function of the bubble size m_i and various other system parameters. The quantity $\mathcal{I}_i$ denotes which of the incoming branches the bubble is on. Within each branch, we require the order of the bubbles to remain constant during the bubbles' travel (i.e., there is no passing allowed). To capture the ordering constraints on the bubbles, we define the function $\mathcal{R}$,

$$\mathcal{R}(i,j) \triangleq \begin{cases} 1, & \text{if } \mathcal{I}_i = \mathcal{I}_j,\ x_j(t_s) < x_i(t_s), \\ -1, & \text{if } \mathcal{I}_i = \mathcal{I}_j,\ x_i(t_s) < x_j(t_s), \\ 2, & \text{if } \{\mathcal{I}_i, \mathcal{I}_j\} \in \mathcal{B}, \\ 0, & \text{if } \mathcal{I}_i \neq \mathcal{I}_j,\ \{\mathcal{I}_i, \mathcal{I}_j\} \notin \mathcal{B}. \end{cases}$$

According to this definition, $\mathcal{R}(i,j) = 1$ if and only if bubbles i and j are on the same branch and bubble j follows bubble i. Similarly, $\mathcal{R}(i,j) = -1$ if and only if bubbles i and j are on the same branch and bubble j precedes bubble i. Additionally, $\mathcal{R}(i,j) = 2$ (respectively $\mathcal{R}(i,j) = 0$) if and only if the pair of bubbles $\{i,j\}$ belong to compatible (resp. non-identical non-compatible) branches. As a short-hand, we say bubbles i and j are compatible (resp. non-compatible) if $\mathcal{R}(i,j) = 2$ (resp. $\mathcal{R}(i,j) \neq 2$).

We describe in detail the generation of bubbles and the algorithm to select the bubbles to schedule in Section IV below. We impose a limit on the number of bubbles that are scheduled at any given time to $\bar{N}$, even if the actual number of bubbles in the system were greater, so as to keep the computational cost manageable. However, in the algorithm we describe in the sequel, each bubble is scheduled at least once and some bubbles may be scheduled more than once. We let t_{s_i} denote the latest time prior to t at which bubble i was scheduled.

We index the vehicles in bubble i as $(i,1), \ldots, (i,m_i)$, where $(i,1)$ refers to the lead vehicle in bubble i and so on until (i,m_i), the last vehicle in the bubble. We also find it convenient for the label $(i,0)$ to represent the last vehicle $(i',m_{i'})$ of the bubble i' that precedes bubble i on the same branch or, if such bubble does not exist, we let $(i,0)$ be an imaginary vehicle located at ∞. We drop the index i whenever there is no ambiguity with regard to the bubble.

Aspect 2 – scheduling of bubbles: The job of the scheduler is to prescribe to each bubble an *approach time* τ_i - the time at which the i^{th} bubble is to reach the beginning of the intersection, i.e., $x_i(\tau_i) = 0$, so that no two different bubbles collide. In solving this problem, the scheduler has to respect the order of bubbles on the same branch and take into account no-collision constraints between bubbles on two different branches that are not compatible. The preservation of the order of intersection crossing by the bubbles on the same branch takes the form,

$$\tau_j \geq \tau_i + \bar{\tau}_i^{\text{occ}}, \quad \text{if } \mathcal{R}(i,j) = 1, \tag{4a}$$

for $i,j \in \{1, \ldots, N\}$. Note that these constraints only ensure that the passage of bubbles on a branch through the intersection occurs in the same order as they have arrived, but they do not necessarily exclude collisions for the entire travel time. The intra-branch collisions are avoided at a local level and we accept the resulting sub-optimality. On the other hand, the no-collision constraint between bubbles on two different non-compatible branches takes the form,

$$\tau_i \geq \tau_j + \bar{\tau}_j^{\text{occ}} \text{ OR } \tau_j \geq \tau_i + \bar{\tau}_i^{\text{occ}}, \quad \text{if } \mathcal{R}(i,j) = 0, \tag{4b}$$

for $i,j \in \{1, \ldots, N\}$. Note that there are no scheduling constraints for pairs of bubbles belonging to compatible branches.

The constraints (4b) make the scheduling problem combinatorial in nature because of the need to determine whether i or j goes first. Since the order on each branch is to be preserved, the number of sub-problems is upper bounded by the number of permutations of the multiset $\{\mathcal{I}_p\}_{p=1}^N$, i.e.,

$$\frac{N!}{\prod_{p=1}^8 N_p!} = \frac{\left(\sum_{p=1}^8 N_p\right)!}{\prod_{p=1}^8 N_p!},$$

where recall that N_p is the number of bubbles on branch p and N is the total number of bubbles. This upper bound is obtained by ignoring the possibility of simultaneous use of the intersection by compatible bubbles. The precise number of sub-problems depends on the initial conditions as well as on the specific choices of the approach times τ_i. We describe in detail the algorithm for optimal scheduling of bubbles in Section V.

Aspect 3 – local vehicular control: The local vehicular control has various equally relevant goals. The first goal is to avoid collisions within each bubble and among different bubbles in the same branch. The second goal is for the local vehicular control to ensure that the bubble approaches the intersection at the prescribed time τ_i and that the occupancy time of the bubble, τ_i^{occ}, is no more than $\bar{\tau}_i^{\text{occ}}$. The scheduler requires the quantity $\bar{\tau}_i^{\text{occ}}$ and other quantities such as earliest and latest times of approach at the intersection for the bubble that are functions of the initial conditions. All these quantities may be computed by the bubble and passed on to the IM or, instead, the state of each car may be passed to the IM. We assume that the control law at the vehicle level ensures that a vehicle does not change bubbles during the course of its travel time. Thus, as far as the scheduling aspect is concerned, m_i may be assumed constant in time. We describe in detail the local vehicular control component in Section VI below.

Remark III.1. *(Relaxation of assumptions).* We discuss here the extent to which the assumptions made in Section II can be relaxed in our proposed design. We make assumption (i) only for the sake of simpler notation and ease of exposition. Our algorithm can handle non-identical vehicles with differing dimensions and differing acceleration limits, though those quantities need to be known. We can relax assumption (iii) if the sources or sinks are not close to the intersection with minor

changes in our algorithm for bubble generation. The relaxation of assumption (ii) about lane changing is more challenging and we hope to address it in future. Finally, we assume that vehicles/bubbles from non-compatible branches must not occupy the intersection simultaneously. However, this could be relaxed by imposing the constraint that vehicles from each pair of non-compatible branches do not occupy a small region (determined based on the dimensions and properties of the vehicles) around the intersection point of the geometric paths from that pair of branches. We have made our assumption to avoid additional notation and simplify the exposition given that the conceptual loss is not significant. •

IV. Dynamic vehicle clustering

The primary motivation for clustering vehicles into bubbles is to reduce the computational burden on the scheduler. Consequently, we impose the upper bound $\bar{\mathcal{N}}$ on the number of bubbles that the scheduler needs to consider at any given instance. Further, as new vehicles arrive, they need to be assigned to new bubbles. In order to balance both requirements, we divide each branch into three zones, as shown in Figure 2: staging zone (of length L_s), mid zone (of length L_m) and exit zone (of length L_e). For each branch $p \in \{1, \ldots, 8\}$, we let $\mathcal{Z}_p^s$, $\mathcal{Z}_p^m$ and $\mathcal{Z}_p^e$ be the set of positions on the branch p corresponding to the staging, mid and exit zones, respectively.

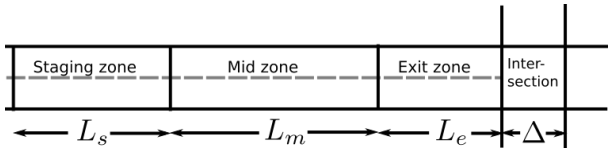

Fig. 2. Division of an incoming branch into zones.

The `clustering into bubbles` algorithm is executed every T_{cs} units of time. At each clustering instance $t_s = sT_{cs}$, $s \in \mathbb{N}_0$, the vehicles in the staging zone that do not already belong to a bubble are clustered. Thus, the choice $T_{cs} < \frac{L_s}{v^M}$, where recall that v^M is the max speed limit, ensures that every vehicle belongs to a bubble before it leaves the staging zone and enters the mid zone. We impose an upper bound $\bar{\mathcal{N}}_p$ on the number of new bubbles that may be created on branch p at any clustering instance. At a clustering instance $t_s = sT_{cs}$, let n_p^{ua} denote the number of vehicles to be clustered in the staging zone of branch p (the superscript ua stands for "unassigned"). Then, the n_p^{ua} vehicles are clustered based on their position using the $\mathcal{M}_p$-means algorithm, with $\mathcal{M}_p = \min\{n_p^{ua}, \bar{\mathcal{N}}_p\}$, see e.g., [25]. Thus, the n_p^{ua} vehicles on branch p are partitioned into $\mathcal{M}_p$ number of clusters or bubbles such that the sum of squares of the distances from each car to the center of its bubble is minimized. The clustering component in our design is modular and hence any alternative clustering algorithm may be used.

We make the assumption that $\bar{\mathcal{N}} \geq \sum_p \bar{\mathcal{N}}_p$. This allows for the possibility that, at each scheduling time instant, some of the previously scheduled bubbles may be scheduled again. The algorithm makes sure that no more than $\bar{\mathcal{N}}$ bubbles are passed to the IM manager for scheduling at any instance. This is achieved using two observations. First, previously scheduled bubbles that have already entered the exit zone of their branch are no longer fed to the IM for scheduling (i.e., its schedule is not modified any further). Second, if the number of newly created bubbles and the previously created bubbles yet to enter the exit zone exceeds $\bar{\mathcal{N}}$, then the algorithm pops out the required number of bubbles from the top of the list of bubbles previously scheduled (corresponding to the ones closer to their respective exit zones). We present the precise description of the `clustering into bubbles` algorithm in Algorithm 1.

Algorithm 1: `clustering into bubbles` at sT_{cs}

Input: $\mathcal{L}_0, \tau_0^{\min}$
`{Ordered list of bubbles scheduled at` $(s-1)T_{cs}$ `and earliest approach time used in scheduling them}`
1: $\mathcal{L} \leftarrow \mathcal{L}_0 \setminus \{j \in \mathcal{L} : \mathcal{I}_j = p \ \wedge \ x_j \notin \mathcal{Z}_p^s \cup \mathcal{Z}_p^m\}$
`{remove bubbles that are not completely within the staging and the mid zones}`
2: **for** $p = 1$ **to** 8 **do**
3: $\bar{\mathcal{N}}_p$ `{max new bubbles on branch` p`}`
4: $\mathcal{M}_p \leftarrow \min\{n_p^{ua}, \bar{\mathcal{N}}_p\}$ `{# new bubbles on branch` p`}`
5: Cluster new vehicles on branch p using $\mathcal{M}_p$-means algorithm
6: **end for**
7: $\mathcal{M} \leftarrow \sum_{p=1}^{8} \mathcal{M}_p$
8: **if** $\mathcal{M} + |\mathcal{L}| > \bar{\mathcal{N}}$ **then**
9: Remove first $\mathcal{M} + |\mathcal{L}| - \bar{\mathcal{N}}$ bubbles from $\mathcal{L}$ `{Ensure only` $\bar{\mathcal{N}}$ `bubbles provided to scheduler by dropping the earliest bubbles in previous schedule}`
10: **end if**
11: Append new bubbles to $\mathcal{L}$
12: $\tau^{\min} \leftarrow \max\left(\{\tau_0^{\min}\} \cup \{\tau_i + \bar{\tau}_i^{\text{occ}} : i \in \mathcal{L}_0 \setminus \mathcal{L}\}\right)$
`{earliest approach time for the bubbles in` $\mathcal{L}$`}`
Output: $\mathcal{L}$, $\tau^{\min}$

The algorithm takes in the list of bubbles $\mathcal{L}_0$ scheduled on the last iteration and an earliest approach time $\tau_0^{\min}$ used when scheduling it. The output is a list of bubbles $\mathcal{L}$ to be scheduled and the earliest approach time $\tau^{\min}$ for them. Note from step 12 of Algorithm 1 that $\tau^{\min}$ is an upper bound on the time by which all the bubbles not in the $\mathcal{L}$ list are guaranteed to cross the intersection. Thus, when scheduling $\mathcal{L}$, the scheduler imposes the constraint that the bubbles in $\mathcal{L}$ approach the intersection no earlier than $\tau^{\min}$.

Remark IV.1. *(Effect of zone lengths on clustering and scheduling).* The lengths of the three zones illustrated in Figure 2 directly affect the resulting traffic coordination. Although we do not pursue here a systematic design of these zone lengths, we can identify some basic observations of their effect on clustering and scheduling. We envision these zone lengths to be of the order of several tens of meters. The length L_s of the staging zone has a direct effect on the time step of the periodic execution of clustering and scheduling as well as on the number of vehicles per bubble. The length L_m of the mid zone has an effect on the likelihood of scheduling a bubble more than once. Finally, the length L_e of the exit zone has an effect on the feasibility of the scheduling problem, which we guarantee by assuming that L_e is large enough for a vehicle to come to a complete stop from a maximum speed of v^M in under a distance L_e. Feasibility of the scheduling problem does not affect liveness of our algorithm but rather affects guarantees on safety. If L_e was not large enough, then

vehicles may not be able to come to a complete stop before the intersection, which in turn would impose finite interval constraints on the time duration during which the bubble could safely use the intersection. •

Remark IV.2. *(Re-clustering).* The `clustering into bubbles` algorithm is just one method of defining bubbles and selecting which ones to schedule. In this algorithm, a vehicle is assigned to a bubble only once and the vehicle is part of that bubble throughout its travel. However, one could implement a strategy which re-clusters all vehicles in the staging and mid zones so that vehicles may be reassigned to a different bubble, bubbles may be merged or split as needed, and so on. Such an algorithm would also allow sources and sinks on the branch such as smaller streets, homes, and retail. •

V. Scheduling of Bubbles

This section describes the scheduling algorithm employed by the intersection manager (IM) to decide the order of passage, through the intersection, of the bubbles in $\mathcal{L}$, which is provided by the clustering algorithm. The scheduling algorithm is also executed every T_{cs} units of time. In this section, we let $\mathcal{L}$ be the set $\{1,\dots,N\}$, where $N=|\mathcal{L}|$, without loss of generality.

A. Cost function and constraints

In our approach, the IM schedules bubbles as a whole using an abstraction of the vehicle dynamics and the cost function. First, regarding the vehicle dynamics, we note that the inter-vehicle approach times at the intersection and the resulting occupancy time of a bubble is a degree of freedom. However, we have made the alternative choice of not considering it as such in the scheduling algorithm, and instead only use an upper bound on the occupancy time $\bar{\tau}_i^{\text{occ}}$ (that the local vehicular control component can guarantee) appearing in the constraints (4). Second, regarding the cost function, we abstract the fuel cost for the vehicles in a bubble i into a single function F_i that depends only on the average velocity of the bubble i (lead vehicle in the bubble) for $t\in[t_s,t_s+\tau_i]$, where $t_s=sT_{cs}$ is the time at which the scheduling algorithm is executed. We allow for the possibility of the function F_i depending on the initial conditions of the vehicles in bubble i. Thus, the scheduling algorithm minimizes the following simplified cost function $\mathcal{C}_{\mathcal{L}}$, where $\mathcal{C}_{\mathcal{L}}$ for a given list of bubbles $\mathcal{L}$ is

$$\begin{aligned}\mathcal{C}_{\mathcal{L}} &\triangleq \sum_{i\in\mathcal{L}} m_i(W_T\tau_i + F_i(\bar{v}_i))\\ &= \sum_{i\in\mathcal{L}} m_i\Big(W_T\frac{d_i}{\bar{v}_i} + F_i(\bar{v}_i)\Big) \triangleq \sum_{i\in\mathcal{L}}\phi_i(\bar{v}_i). \end{aligned} \tag{5}$$

Here, $\bar{v}_i$ is the average velocity of the lead vehicle in bubble i for $t\in[t_s,t_s+\tau_i]$, i.e., $\bar{v}_i=\frac{d_i}{\tau_i}$, where $d_i\triangleq -x_i(t_s)$. The optimization variables are $\bar{v}_i$ for each bubble $i\in\mathcal{L}$.

Note that in the cost function $\mathcal{C}_{\mathcal{L}}$, the functions F_i could, in general, depend on initial conditions modeled as parameters (such as the distance d_i to reach the intersection). The cost function (5) models a combination of cumulative travel time and total fuel usage. Motivated by the fact that fuel efficiency is typically an increasing function of vehicle speed for speeds under the limits enforced on most roads with intersections, we make the assumption that, for each $i\in\mathcal{L}$, $F_i:[0,v^M]\mapsto\mathbb{R}_{>0}$ is monotonically decreasing.

Remark V.1. *(Simplified cost function).* Note that (5) is a much simplified version of the original cost function (3). Here the aim is to come up with a cost function for a bubble as a whole. In general, the total cost incurred by the vehicles of a given bubble in the sense of (3) has quite a complicated dependence on the interactions among the vehicles of the bubble as well as those of other bubbles on the same branch and other branches. However, capturing this complicated relationship accurately may not sufficiently reduce, if at all, the complexity (in terms of optimization variables) compared to (3). Thus, we have used the heuristic that longer travel times (equivalently lower average velocities) mean idling for longer and lower fuel efficiency for the same distance, which hence must incur more cost. Nevertheless, we allow for the functions F_i to potentially depend on the initial conditions or any other relevant data. The task of systematically fine-tuning the functions F_i based on available data and quantifying the suboptimality is challenging, and we leave such analysis for future work. However, we would like to point out that even without any data-based customization of F_i, our simulations (cf. Figure 4 in Section VIII, which uses the actual cost function) show that the proposed algorithm in general outperforms signal-based traffic management in terms of fuel efficiency over a wide range of traffic flow rates. •

Regarding the constraints, conditions on the travel times can be re-expressed as conditions on average velocities as

$$\begin{aligned}&\tau_i\geq\tau_j+\bar{\tau}_j^{\text{occ}} \iff \frac{d_i}{\bar{v}_i}\geq\frac{d_j}{\bar{v}_j}+\bar{\tau}_j^{\text{occ}}\\ &\iff \bar{v}_j\geq c_{ji}\bar{v}_i+b_{ji}\bar{v}_j\bar{v}_i,\quad c_{ji}=\frac{d_j}{d_i},\ b_{ji}=\frac{\bar{\tau}_j^{\text{occ}}}{d_i}.\end{aligned} \tag{6}$$

Thus, we re-express the no-collision constraints (4) as

$$\bar{v}_i\geq c_{ij}\bar{v}_j+b_{ij}\bar{v}_i\bar{v}_j,\quad \text{if } \mathcal{R}(i,j)=1, \tag{7a}$$

$$\bar{v}_j\geq c_{ji}\bar{v}_i+b_{ji}\bar{v}_j\bar{v}_i \text{ OR } \bar{v}_i\geq c_{ij}\bar{v}_j+b_{ij}\bar{v}_i\bar{v}_j,\ \text{if } \mathcal{R}(i,j)=0. \tag{7b}$$

In addition, we also need to ensure that the scheduling at instance sT_{cs} of the bubbles in $\mathcal{L}$ does not conflict with the ones that have been previously scheduled. Formally, this corresponds to having the time τ_i to reach the intersection for bubble i be no less than $\tau^{\min}$ (cf. step 12 of Algorithm 1). Equivalently, we require

$$\bar{v}_i\leq\frac{d_i}{\tau^{\min}}. \tag{7c}$$

Note that the scheduling problem is combinatorial in nature due to the no-collision constraints (7b). Thus, even though the cost function $\mathcal{C}_{\mathcal{L}}$ is simple and the optimization variables are the average velocities $\bar{v}_i$, we believe this formulation provides a good balance between usefulness and computational tractability. Further, the local vehicular control we present in Section VI seeks an optimal control profile to achieve the prescribed average velocity for the bubble, which justifies the

restriction to $\bar{v}_i$ as the optimization variables in the scheduling aspect. Thus our proposed solution, although sub-optimal, is still principled.

We next describe our solution to the scheduling problem consisting of minimizing $\mathcal{C}_{\mathcal{L}}$ in (5) under the constraints (7) and $\bar{v}_i \in [\bar{v}_i^m, \bar{v}_i^M]$. The lower $\bar{v}_i^m \geq 0$ and upper $\bar{v}_i^M \leq v^M$ limits on the average velocity depend on the initial conditions of the vehicles and desired speed limits. The quantities $\bar{v}_i^m$ and $\bar{v}_i^M$ are inversely related to the *latest time of approach* and the *earliest time of approach* at the intersection for bubble i, respectively. The computation of these quantities is described in Section VI-A. Similarly, the upper bound $\bar{\tau}_i^{\text{occ}}$ on the occupancy times may be computed as in Section VI-B. In the first part of our solution to the scheduling problem, we determine the optimal schedule and optimal cost given a fixed *partial order* of bubble passage through the intersection. In the second part, we use a branch-and-bound algorithm to find the optimal order and schedule.

B. Optimal bubble average velocity for fixed partial order of passage

Given a desired partial order (which is also a list) of bubble passage through the intersection, here we address the problem of determining the optimal average velocities of the bubbles and the associated optimal cost. For this purpose, define a *partial order* of the approach times of the bubbles as a list, P, of the integers from 1 to $|P| \leq N$. We use $P(i)$ to denote the i^{th} element in the list and we use $\sigma_P(i)$ to denote the position of bubble i in the list P. Each partial order P encodes one out of all the sub-problems that are induced by the scheduling constraints (4b) (or equivalently (7b)). In particular, a partial order P refers to the sub-problem with the constraints

$$\tau(P(i)) \geq \tau(P(j)) + \bar{\tau}^{\text{occ}}(P(j)),$$

for all $i \in \{1, \ldots, |P|\}$ and for all $j \in \{1, \ldots, i-1 : \mathcal{R}(P(i), P(j)) \neq 2\}$. In other words, P encodes the sub-problem in which each bubble $P(i)$ enters the intersection only after all the non-compatible bubbles $P(j)$ that appear before $P(i)$ in the list P have passed through the intersection completely. Note that more than one list can refer to the same sub-problem. For example, assuming the bubbles 1 and 2 are compatible, then the lists $\{1, 2\}$ and $\{2, 1\}$ each refer to the same sub-problem. In the sequel, we consider only those lists that respect the intra-branch orders, that is $\sigma_P(i) < \sigma_P(j)$ if $\mathcal{R}(i, j) = 1$. Given P that respects the intra-branch orders, the `bubble velocity optimization` algorithm, formally described in Algorithm 2, finds a solution to the optimization of $\mathcal{C}_P$ (which is same as in (5) but with i ranging over the set P rather than $\mathcal{L}$) under the constraints (7), $\bar{v}_i \in [\bar{v}_i^m, \bar{v}_i^M]$, and with partial order P.

The following result shows that, for a partial order that respects the intra-branch order, the algorithm finds the average velocities that optimize the cost function $\mathcal{C}_P$.

Lemma V.2. *(`bubble velocity optimization` algorithm optimizes the schedule given a partial order that respects the intra-branch orders). For each $i \in \{1, \ldots, N\}$, assume the fuel cost function F_i is monotonically decreasing. Let P, with $|P| \leq N$, be a partial order respecting the intra-branch orders and denote by $\bar{v}^P = (\bar{v}_1^P, \ldots, \bar{v}_N^P)$ and C the output of Algorithm 2. Then, $\bar{v}^P$ and C are, respectively, a minimizer and the minimum cost of the optimization problem with the cost function as $\mathcal{C}_P$ (5) under the constraints (7), $\bar{v}_i \in [\bar{v}_i^m, \bar{v}_i^M]$.*

Algorithm 2: `bubble velocity optimization`

Input: Order P

$C \leftarrow 0$
for $h = 1$ **to** $|P|$ **do**
  $i \leftarrow P(h)$ {`bubble` i `is in position` h `in` P}
  if $h = 1$ **then**
    $\bar{v}_i^P \leftarrow \bar{v}_i^M$
  else
    $R \leftarrow \{r \in \{P(1), \ldots, P(h-1)\} : \mathcal{R}(i, r) \neq 2\}$ {`set of bubbles in` P `up to position` $h-1$ `that are non-compatible with bubble` i}
    $\bar{v}_i^P \leftarrow \min_{r \in R}\{\bar{v}_i^M, \dfrac{\bar{v}_r^P}{c_{ri} + b_{ri}\bar{v}_r^P}\}$
  end if {$\bar{v}_i^P$ `is the optimizer for bubble` i}
  $C \leftarrow C + \phi_i(\bar{v}_i^P)$ {`update cost`}
end for

Proof. Given the order P, the constraints (7) reduce to

$$\frac{\bar{v}_j}{c_{ji} + b_{ji}\bar{v}_j} \geq \bar{v}_i$$

where $j = P(h-1)$, $i = P(h)$ and $h \in \{2, \ldots, N\}$. The left-hand side of the inequality is an increasing function of $\bar{v}_j$. Since F_i is a monotonically decreasing function for each i, $\bar{v}_i^P$ takes the maximum possible value. The algorithm computes the components of $\bar{v}^P$ iteratively and the result follows. □

C. Optimal ordering via branch-and-bound

We propose a branch-and-bound algorithm to solve the optimal scheduling problem. We start by providing an informal description.

> *Informal description:* A branch-and-bound algorithm consists of a systematic enumeration of the set of candidate solutions as a rooted tree[2], with the full set at the root. The algorithm explores branches of the tree, which represent subsets of the set of candidate solutions. Before evaluating the candidate solutions of a branch, the branch is checked against upper bounds on the optimal solution, and is discarded if it is determined that it cannot produce a better solution than the best one found so far.

We formally specify each of the components in this description next, starting with the rooted tree. We let P denote any partial order of up to length N, with non-repeating numbers

[2]We make use of basic notions from graph theory, cf. [26], [27]. A digraph of order n is a pair $G = (V, E)$, where V is a set with n elements called nodes and E is a set of ordered pair of nodes called edges. A directed path is an ordered sequence of nodes such that any ordered pair of nodes appearing consecutively is an edge. A cycle is a directed path that starts and ends at the same node and contains no repeated node except for the initial and the final one. A digraph is acyclic if it has no cycles. A directed (or rooted) tree is an acyclic digraph with a node, called root, such that any other node can be reached by one and only one directed path starting at the root. If (i, j) is an edge of a tree, i is the parent of j, and j is the child of i. A node j is called a descendant of a node i if there is a directed path from i to j. Given a tree, a subtree rooted at i is the tree that has i as its root and is composed by all its descendants in the original tree.

drawn from $\{1,\dots,N\}$, and preserving the individual branch orders. With this notation, the empty list $P=\emptyset$ denotes the root of the tree, representing all feasible orders. Similarly, $P=(i_1,\dots,i_h)$ denotes the subtree of all the feasible partial orders in which bubbles i_1 through i_h each enter the intersection only after all the non-compatible bubbles preceding each of them, respectively, in the list has completely crossed the intersection. The order of the remaining bubbles is undetermined.

Our next step is to provide a way to determine a lower bound on the achievable optimal value of any given branch. This follows from the observation that (i) the execution of the `bubble velocity optimization` algorithm finds the optimal value of the average velocity of a bubble given the partial order of *all* the bubbles preceding it, but (ii) one can compute an lower bound for the optimal value even if only part of the order of bubbles preceding it is known. The description in Algorithm 3 of this procedure, termed `bounding optimal bubble velocity` algorithm, relies on eight ordered lists, termed queues, one for each branch. The queue for branch p, $Q_p=(i_{p,1},\dots,i_{p,N_p})$, is initialized to the list of all the bubbles on branch p in their order of arrival (thus $\mathcal{R}(i_{p,s},i_{p,s+1})=1$ for all $s\in\{1,\dots,N_p-1\}$). We denote by H_i^P the upper bound on the average velocity $\bar v_i$ of bubble i obtained by Algorithm 3 given that a non-empty P precedes it. This allows us to lower bound the optimal cost for any partial order in the subtree P in terms of $\bar v_i^P$ and H_i^P as follows,

$$C^P \triangleq \sum_{i\in P}\phi_i(\bar v_i^P)+\sum_{i\in\mathcal{L}\setminus P}\phi_i(H_i^P). \tag{8}$$

This lower bound is precisely what is required to implement a branch-and-bound algorithm to find the optimal schedule for the bubbles.

Algorithm 3: `bounding optimal bubble velocity`

Compute $\{\bar v^P_{P(1)},\dots,\bar v^P_{P(|P|)}\}$ using `bubble velocity optimization` with input P
for $p=1$ **to** 8 **do**
  $\mathcal{Q}_p \leftarrow Q_p \setminus P$ {`pop-out` P `from` Q_p}
  if $\mathcal{Q}_p\neq\emptyset$ **then**
    $i\leftarrow\mathcal{Q}_p(1)$ {i `is first of remaining bubbles in` $\mathcal{Q}_p$}
    $R\leftarrow\{r\in P:\mathcal{R}(i,r)\neq 2\}$ {`set of bubbles in` P `that are non-compatible with bubble` i}
    $H_i^P\leftarrow\min_{r\in R}\{\bar v_i^M,\frac{\bar v_r^P}{c_{ri}+b_{ri}\bar v_r^P}\}$
    for $s=2$ **to** $|\mathcal{Q}_p|$ **do**
      $i\leftarrow\mathcal{Q}_p(s)$
      $j\leftarrow\mathcal{Q}_p(s-1)$
      $H_i^P\leftarrow\min\{\bar v_i^M,\frac{H_j^P}{c_{ji}+b_{ji}H_j^P}\}$
    end for
  end if
end for

Specifically, the branch-and-bound algorithm starts by picking an arbitrary candidate order and computing the cost for it, using the `bubble velocity optimization` algorithm, and storing the two as the current best solution and cost. Then, starting at the root node of the tree of all feasible orders, the algorithm searches (e.g., using depth-first or breadth-first search) for an optimal solution. If at any time a leaf node, which corresponds to a fully determined order, is reached and its cost is better than the current best, then the current best solution and cost are updated. For any other node P in the tree, (8) provides a lower bound C^P on the cost of all the orders represented by the node P. If C^P is greater than the current best known cost, then the subtree P is discarded. This process continues until the algorithm finds the optimal solution. We refer to this process as the `schedule optimization` algorithm.

VI. Local vehicular control

The local vehicular control component of our hierarchical-distributed coordination approach involves two main tasks: (i) compute, for each bubble i, the lower $\bar v_i^m$ and upper $\bar v_i^M$ average velocity bounds, and the upper bound on the intersection occupancy time $\bar\tau_i^{\rm occ}$ that are provided to the scheduler; and (ii) control the vehicles ensuring safety and so that all the vehicles of bubble i cross the intersection within the time interval $[\tau_i,\tau_i+\bar\tau_i^{\rm occ}]$ prescribed by the scheduler. The successful execution of each of these tasks requires an understanding of the vehicle dynamics and the desired safety constraints and the effect of each on the other. Due to the modular nature of the different aspects of our hierarchical-distributed solution, any distributed algorithm that meets the above two requirements can be used for the local vehicular control. As an example, we invoke the controller design from our previous work [24] on vehicle strings under finite-time and safety specifications. For the sake of completeness, we review here the main elements and performance guarantees of this design as needed by the hierarchical-distributed approach.

A. Bounds on average bubble velocity

Recall that $\bar v_i$ is the average velocity of the lead vehicle of bubble i from t_s and until the lead vehicle is supposed to reach the beginning of the intersection at τ_i. Thus, it may seem that computing lower and upper bounds on the achievable average velocity of the lead vehicle in the bubble is sufficient to determine $\bar v_i^M$ and $\bar v_i^m$. However, ignoring the initial conditions of the other vehicles in the bubble in the computation of $\bar v_i^M$ and $\bar v_i^m$ poses the risk of lengthening the guaranteed upper bound $\bar\tau_i^{\rm occ}$ on the occupancy time. The reasoning for this is better explained in terms of earliest times of approach of the vehicles at the intersection.

In bubble i, we let $\tau^e_{i,k}$ be the earliest time vehicle (i,k) can reach the intersection ignoring the other vehicles on the branch. Letting $t_{s_i}=s_iT_{cs}$ be the time at which bubble i was last scheduled, the quantity $\tau^e_{i,k}-t_{s_i}$ is then the time it takes $x^v_{i,k}$ to reach 0 from $x^v_{i,k}(t_{s_i})$ for the trajectory with maximum acceleration until $v^v_{i,k}=v^M$ and zero acceleration thereafter. Thus, we see that if $\tau^e_{i,k}$ for some $k>1$ is significantly greater than $\tau^e_{i,1}$ then the vehicle $(i,1)$ has to slow down to approach the intersection at a time later than $\tau^e_{i,1}$ so that the guaranteed upper bound $\bar\tau_i^{\rm occ}$ on the occupancy time is small enough. Thus, we propose an alternative solution. To do so, we first introduce the notion of safe-following distance.

Definition VI.1. *(Safe-following distance). The maximum braking maneuver (MBM) of a vehicle is a control action that*

sets its acceleration to u_m until the vehicle comes to a stop, at which point its acceleration is set to 0 thereafter. Let $k-1$ and k be the indices of two vehicles on the same branch, with vehicle k immediately following $k-1$, and define

$$\mathcal{D}(v^v_{k-1}(t), v^v_k(t)) \triangleq L + \max\left\{0, \frac{1}{-2u_m}\left((v^v_k(t))^2 - (v^v_{k-1}(t))^2\right)\right\}. \quad (9)$$

The quantity $\mathcal{D}(v^v_{k-1}(t), v^v_k(t))$ is a safe-following distance *at time t for the pair of vehicles $k-1$ and k because if $x^v_{k-1}(t) - x^v_k(t) \geq \mathcal{D}(v^v_{k-1}(t), v^v_k(t))$ and, if each of the two vehicles were to perform the MBM, then the two vehicles would be safely separated, $x^v_{k-1} - L \geq x^v_k$ until they come to a stop.* •

We let $\nu^{\text{nom}}_{\mathcal{S}}$ and $\nu^{\text{nom}}_{\mathcal{T}}$ be the nominal speeds for vehicles when entering the intersection, depending on whether the bubbles they belong to go straight or take a turn. Since vehicles typically need to turn at a lower speed than when going straight $\nu^{\text{nom}}_{\mathcal{T}} \leq \nu^{\text{nom}}_{\mathcal{S}}$. We define $\mathcal{D}^{\text{nom}}_{\mathcal{S}} \triangleq \mathcal{D}(\nu^{\text{nom}}_{\mathcal{S}}, v^M)$ for bubbles that go straight and $\mathcal{D}^{\text{nom}}_{\mathcal{T}} \triangleq \mathcal{D}(\nu^{\text{nom}}_{\mathcal{T}}, v^M)$ for those that take a turn at the intersection. When no distinction needs to be made between the bubbles or when it is clear from the context which of the two variants is being referred to, we drop the qualifying subscript. In each case, $\mathcal{D}^{\text{nom}}$ has the connotation of a safe inter-vehicle distance given a vehicle is traveling at the maximum allowed speed v^M and the vehicle preceding it traveling at a speed greater than or equal to ν^{nom}. Then, we also define $T^{\text{nom}} \triangleq \mathcal{D}^{\text{nom}}/\nu^{\text{nom}}$ as the *nominal inter-vehicle approach time*. In the design of the local vehicular control, we require each vehicle to enter the intersection with a speed of at least ν^{nom}. Then, as we shall see in the sequel, T^{nom} determines the bound on the inter-approach time of any two consecutive vehicles in a bubble.

With this notation in place, we see that the earliest time of approach for vehicle (i,k) forces the earliest time of approach of bubble i, i.e. vehicle $(i,1)$, to be no less than $\tau^e_{i,k} - (k-1)T^{\text{nom}}$. Hence, we define the *earliest time of approach* for the bubble i, τ^e_i as

$$\tau^e_i \triangleq \max\{\tau^e_{i,k} - (k-1)wT^{\text{nom}} : k \in \{1, \ldots, m_i\}\}, \quad (10)$$

where $w \in [0,1]$ is the *prescribed approach times tuning parameter*. Then, we let $\bar{v}^M_i = \frac{-x_i(t_s)}{\tau^e_i}$. Analogous computations with maximum deceleration yield the *latest time of approach* τ^l_i of bubble i, possibly with $\tau^l_i = \infty$, and the corresponding lower bound $\bar{v}^m_i \geq 0$ for the average velocity. Hence, the values we obtain in this way for $\bar{v}^m_i$ and $\bar{v}^M_i$ are, respectively, larger and smaller than the ones we would have obtained if we only took into account the lead vehicle of the bubble.

We refer to a schedule as *feasible* if it satisfies the constraints (4), or equivalently (7), which ensures non-collision between bubbles on two different non-compatible branches. For a given upper bound on the occupancy time and the sets of $\bar{v}^m_i$ and $\bar{v}^M_i$ for $i \in \mathcal{L}$, a feasible schedule may not always exist. Thus, to guarantee the feasibility of the scheduling problem in a simple fashion, we assume that the exit zone length L_e is large enough.

Lemma VI.2. *(Existence of a feasible schedule). If the exit zone length, $L_e \geq \frac{(v^M)^2}{-2u_m} + \frac{\max\{(\nu^{nom}_{\mathcal{S}})^2, (\nu^{nom}_{\mathcal{T}})^2\}}{2u_M}$, then there always exists a feasible schedule, in which each vehicle is able to enter the intersection with a speed of at least $\max\{\nu^{nom}_{\mathcal{S}}, \nu^{nom}_{\mathcal{T}}\}$.*

Proof. Recall, that a schedule is assigned to a bubble when all the vehicles in the bubble are still in the staging or the mid zones. Clearly, the condition on L_e implies that any vehicle in the staging zone or the mid zone ($x^v_k \leq -L_e$) can come to a complete stop and then accelerate to a speed of at least $\max\{\nu^{\text{nom}}_{\mathcal{S}}, \nu^{\text{nom}}_{\mathcal{T}}\}$ before arriving at the beginning of the intersection ($x^v_k = 0$). □

Note that, under the assumptions of Lemma VI.2, the resulting schedule can handle any vehicle inflow, in the sense that all vehicles can go through the intersection without collisions in the intersection. Essentially, this is because each vehicle has the possibility to safely come to a stop before the intersection. As a result, if necessary, vehicles can stop for an arbitrarily long time before the intersection prior to using it.

B. `local vehicular` controller and bound on guaranteed occupancy time

Here, we present the structure of the local vehicle controller, the complete details of which appear in [24]. In particular, this controller is a switched controller that has two modes - the uncoupled mode and the safe-following mode. When the state is in these modes, an uncoupled controller g_{uc} and a safe-following controller g_{sf}, respectively is active. Essentially, a vehicle is in the uncoupled mode if the vehicle immediately preceding it is sufficiently far away and in the safe-following mode otherwise. To make precise whether two vehicles are sufficiently far from each other, we introduce the *coupling set* $\mathcal{V}$ defined by

$$\mathcal{V} \triangleq \{(\mathrm{v}_1, \mathrm{v}_2, \sigma) : \mathrm{v}_2 \geq \mathrm{v}_1 \text{ and } \sigma \in [1, \sigma_0]\}, \quad (11)$$

with v_1 and v_2 being the velocities of the leading vehicle and the following vehicle, respectively; $\sigma_0 > 1$ a design parameter and σ is the *safety ratio* between two consecutive vehicles on the same branch. To be precise, we define the safety ratio between vehicle k and vehicle $k-1$ as

$$\sigma_k(t) \triangleq \frac{x^v_{k-1}(t) - x^v_k(t)}{\mathcal{D}(v^v_{k-1}(t), v^v_k(t))},$$

Thus, the safety ratio is the ratio of the actual inter-vehicle distance to the safe-following distance. The vehicles are at a safe following distance as long as $\sigma > 1$. Intuitively, if $\zeta_k \in \mathcal{V}$, then vehicle k is going at least as fast as the vehicle in front of it, and their safety ratio is close to 1. With this in mind, we define the `local vehicular` controller for vehicle k,

$$u^v_k(t) = \begin{cases} g_{uc}, & \text{if } \zeta_k \notin \mathcal{V},\ v^v_k < v^M, \\ [g_{uc}]^0_{u_m}, & \text{if } \zeta_k \notin \mathcal{V},\ v^v_k = v^M, \\ g_{sf}, & \text{if } \zeta_k \in \mathcal{V},\ v^v_k < v^M, \\ [g_{sf}]^0_{u_m}, & \text{if } \zeta_k \in \mathcal{V},\ v^v_k = v^M. \end{cases} \quad (12)$$

This controller has the vehicle use the safe-following controller when in the coupling set, and the uncoupled controller otherwise.

The last element of the design is the upper bound on the guaranteed occupancy time for a bubble. To obtain this, we first upper bound the inter-approach times of vehicles in a given bubble at the beginning of the intersection.

Proposition VI.3. *(Upper bound on the inter-approach times of vehicles in a bubble at the intersection [24]). For any bubble i and any vehicle $k \in \{2,\dots,m_i\}$, if $v^v_{i,k-1}(T^a_{i,k-1}) \geq \nu^{nom}$, then $v^v_{i,k}(T^a_{i,k}) \geq \nu^{nom}$ and $T^a_{i,k} - T^a_{i,k-1}$ is upper bounded by*

$$T^{\text{iat}} \triangleq \begin{cases} \sigma_0 T^{nom}, & \text{if } \underline{v}^v \geq \nu^{nom}, \\ \max\{\sigma_0 T^{nom}, T^{\text{fol}}(\underline{v}^v)\}, & \text{if } \underline{v}^v < \nu^{nom}, \end{cases}$$

where $\underline{v}^v \triangleq \frac{-u_m v^M}{-u_m + \sigma_0 u_M}$ *and*

$$T^{\text{fol}}(v) \triangleq \frac{(\nu^{nom})^2 - v^2}{2u_M v^M} + \frac{\sigma_0 \mathcal{D}(v, v^M)}{v^M} + \frac{\nu^{nom} - v}{u_M}.$$

Using this result, one can guarantee the following upper bound on the intersection occupancy time of a bubble under the `local vehicular` controller.

Corollary VI.4. *(Guaranteed upper bound on occupancy time of a bubble [24]). For any bubble i, its occupancy time τ_i^{occ} is upper bounded as $\tau_i^{\text{occ}} \leq \bar{\tau}_i^{\text{occ}}$, where*

$$\bar{\tau}_i^{\text{occ}} = (m_i - 1)T^{\text{iat}} + \max\left\{\frac{L+\Delta}{\nu^{nom}}, T^{\text{iat}}\right\}. \tag{13}$$

Remark VI.5. *(Generic notation).* In Proposition VI.3 and in Corollary VI.4, ν^{nom} and Δ are used generically. With $\nu^{\text{nom}} = \nu_s^{\text{nom}}$ and $\Delta = \Delta_s$, we get an upper bound on the occupancy time for bubbles going straight while with $\nu^{\text{nom}} = \nu_\tau^{\text{nom}}$ and $\Delta = \Delta_\tau$, we get an upper bound on the occupancy time for bubbles turning left. •

VII. Provably Safe Optimized Traffic Coordination

This section brings together the discussion above on the individual aspects (dynamic vehicle clustering into bubbles, optimized planning and scheduling of the bubbles, and local distributed control for safety and execution of plans) of our hierarchical-distributed coordination approach to intersection traffic. The following result shows that the design ensures vehicle safety and satisfies the prescribed schedule.

Theorem VII.1. *(Provably safe optimized traffic coordination). Consider a traffic intersection with eight incoming branches operating under Assumptions (i)-(v) in Section II, where the vehicle dynamics are given by (2) under the `local vehicular` controller from [24]. Assume the exit zone length satisfies $L_e \geq -(v^M)^2/2u_m + (\nu^{nom})^2/2u_M$ and that, at initial time $t_0 = 0$, vehicles on each branch $p \in \{1,\dots,8\}$ are within the staging zone. Furthermore, suppose that at each $t_s = sT_{cs}$ for each $s \in \mathbb{N}_0$, the vehicles in the staging zone that are clustered by the `clustering into bubbles` algorithm are in a safe configuration ($\sigma_k(t_s) \geq 1$ for each new vehicle k). Then,*

(i) *each vehicle belongs to some cluster, each bubble is scheduled by the `schedule optimization` algorithm at least once. Moreover, at each t_s, this strategy optimizes the schedule of the bubbles $\mathcal{L}$ given by the `clustering into bubbles` algorithm by minimizing the simplified cost function $\mathcal{C}_\mathcal{L}$,*

(ii) *the schedule assigned to the bubbles respects the non-collision constraints (4), with the occupancy time of each bubble i upper bounded by $\bar{\tau}_i^{\text{occ}}$ as given in (13),*

(iii) *inter-vehicle safety is ensured ($\sigma_k \geq 1$) for all vehicles and for all time subsequent to t_0, and*

(iv) *the first vehicle $(i,1)$ of each bubble i approaches the intersection at τ_i, the bubble uses the intersection only within its allotted time interval $[\tau_i, \tau_i + \bar{\tau}_i^{\text{occ}}]$, and each vehicle travels with a velocity of at least ν^{nom} after approaching the intersection.*

Proof. (i) This claim follows from the `clustering into bubbles` and the `schedule optimization` algorithms. Claim (ii) is ensured by the inclusion of the non-collision constraints (4) in the `schedule optimization` algorithm and the feasibility of the scheduling problem guaranteed by Lemma VI.2. Claim (iii) on inter-vehicular safety is a consequence of [24, Lemma IV.1]: for $\sigma_k \in [1, \sigma_0]$, if $\zeta_k \in \mathcal{V}$, then σ_k either stays constant or increases; if on the other hand $\zeta_k \notin \mathcal{V}$, then it means $v_k^v < v_{k-1}^v$ and $x_{k-1}^v - x_k^v$ increases while $\mathcal{D}(v_{k-1}^v, v_k^v)$ stays constant at L and thus σ_k increases. Thus $\sigma_k(t) \geq 1$ is guaranteed for all vehicles k and for all $t \geq t_s$.

Finally, to show claim (iv), we reason as follows. If no bubble precedes bubble i on its branch, then the vehicle $(i,1)$ approaches the intersection at its designated time $\tau_{i,1} = \tau_i$, with at least a velocity of ν^{nom}. Then, by applying Proposition VI.3 inductively, we see that the last vehicle (i, m_i) of bubble i approaches the intersection with a velocity of at least ν^{nom} and $T^a_{i,m_i} \leq T^a_{i,1} + (m_i - 1)T^{\text{iat}}$ and it takes at most $(L+\Delta)/\nu^{\text{nom}}$ amount of time to go past the intersection. Thus from (13), we see that claim (iv) is satisfied in this case. We again use ν^{nom} and Δ generically as mentioned in Remark VI.5. Now suppose bubble j precedes bubble i on its branch and suppose claim (iv) is true for bubble j. From our reasoning above, $T^a_{j,m_j} \leq T^a_{j,1} + (m_j - 1)T^{\text{iat}} = \tau_j + (m_j - 1)T^{\text{iat}}$. Then, from (4) and (13), we have

$$\tau_i \geq \tau_j + \bar{\tau}_j^{\text{occ}} \geq \tau_j + (m_j - 1)T^{\text{iat}} + T^{\text{iat}} \geq T^a_{j,m_j} + T^{\text{iat}},$$

where in obtaining the second inequality we have used (13). Thus, from [24, Theorem IV.6], we conclude that vehicle $(i,1)$ approaches the intersection at its assigned time τ_i with a velocity of at least ν^{nom}. Hence, by using induction over all vehicles in bubble i and over all bubbles i themselves we conclude that claim (iv) holds. □

Theorem VII.1 does not guarantee the optimal operation of the system at the level of individual vehicles under the proposed hierarchical-distributed coordination approach. However, this result guarantees the optimality at the level of bubbles, on each time $t_s = sT_{cs}$, for the bubbles scheduled at t_s. We believe this is a good compromise in balancing the trade-off between optimal vehicle operation and complexity of planning and control.

Remark VII.2. *(Intersection management without traffic signals).* The results and the algorithms presented in this paper culminate in a system for intersection management that does

not require the traditional traffic signals. In particular, each bubble of vehicles is allotted a time interval during which it can utilize the intersection. The scheduling algorithm explicitly has the constraints that no two bubbles occupy the intersection at the same time. Once the schedule for the bubbles is assigned, the vehicles in the bubbles, with the local vehicular control, automatically ensure that they respect the assigned schedule. As a result, no further traffic signaling is required. Thus, signal switching overhead (yellow/amber period) and vehicle startup delays are avoided. •

VIII. Simulations

This section presents simulations of our proposed hierarchical-distributed design and comparisons with an optimized signal-based traffic coordination approach under different traffic conditions. Table I specifies the system parameters that we keep fixed across all the simulations presented here. The parameters T^{nom} and T^{iat} are computed parameters, while the remaining ones in the table are design choices.

TABLE I
System parameters

Parameter	Symbol	Value
General parameters		
Car length	L	4m
Intersection length (straight)	$\Delta_{\mathcal{S}}$	24m
Intersection length (turn)	$\Delta_{\mathcal{T}}$	23.65m
Staging zone length	L_s	70m
Mid zone length	L_m	70m
Exit zone length	L_e	70m
Max. speed limit	v^M	60km/h
Max. accel.	u_M	3m/s^2
Min. accel.	u_m	$-$4m/s^2
HD algorithm parameters		
Nominal speed of crossing (straight)	$\nu_{\mathcal{S}}^{\text{nom}}$	48km/h
Nominal speed of crossing (turn)	$\nu_{\mathcal{T}}^{\text{nom}}$	38.4km/h
Parameter in vehicle controller	σ_0	1.2
Prescribed approach times tuning parameter	w	0
Nominal inter-vehicle approach time (straight)	$T_{\mathcal{S}}^{\text{nom}}$	$\approx$ 1.23s
Nominal inter-vehicle approach time (turn)	$T_{\mathcal{T}}^{\text{nom}}$	$\approx$ 2.29s
Upper-bound on inter-veh. appr. time (straight)	$T_{\mathcal{S}}^{iat}$	$\approx$ 1.58s
Upper-bound on inter-vehicle approach time (turn)	$T_{\mathcal{T}}^{iat}$	$\approx$ 2.75s
Time period for execution of Algorithm 1	T_{cs}	3.77s
Max. # new bubbles on branch p	$\mathcal{N}_p$	1
Max. # bubbles scheduled	$\mathcal{N}$	10
Signal-based algorithm parameters		
Yellow-time (straight)		4.33s
Yellow-time (turn)		4.3s

A. Dynamic traffic generation

In order to simulate dynamically generated traffic, we spawn new vehicles every T_{cs} units of time according to a Poisson arrival process [28]. The mean rate of spawning new vehicles on branch p is λ_p. Thus, the greater the value of λ_p, the greater is the volume of the traffic generated on branch p. New vehicles spawn at the beginning of the staging zone of each branch. The initial velocity of each vehicle is randomly selected (uniformly) between 0 and v^M. However, in order to ensure safe following, a vehicle enters the staging zone possibly at a time later than its spawning time.

To be precise, let $t_s = sT_{cs}$ be the time at which vehicle k spawns, with velocity $v_k^v(t_s)$. Then, the vehicle k enters the staging zone at time $t \geq t_s$, which is the first time instant after t_s such that vehicle k is at a safe distance from the vehicle preceding it on its branch. If there is no previously defined vehicle on the branch, then the position of the vehicle preceding vehicle k is assumed to be ∞. Recall from Section IV that if $T_{cs} < \frac{L_s}{v^M}$, then vehicles entering the problem domain during the time interval $[t_s - T_{cs}, t_s]$ are within the staging zone at t_s.

B. Optimized signal-based traffic coordination

The vehicle control policy in the simulations with signal-based traffic coordination is given by (12) with the safe-following controller g_{sf} as in our algorithm (cf. [24]) and with $g_{uc} = u_M$. The traffic signaling policy is as follows. We consider four phases for the signals. In two phases traffic from opposite branches that go straight are given the right of away and in the remaining two phases, opposite branches that turn left are given the right of way. The yellow times, the time for which signals are in yellow (transition from green to red), are determined by the maximum amount of time required to fully cross the intersection for those vehicles that cannot come to a full stop before the intersection. For our parameters, the yellow time is 4.33s for the 'straight' phases and 4.3s for the 'turning' phases. Then, the green times, the time duration for which a phase gets a green light, for the different phases are optimized following the Webster's method [29] according to the rate of arrival of incoming traffic on the different branches. The four phases get the right of way (green signal) in a round-robin manner.

C. Results and discussion

We present here three sets of simulations labeled Sim1, Sim2 and Sim3. These simulations explore the algorithms under varying incoming traffic rates. Sim1 and Sim2 correspond to homogeneous traffic conditions, while Sim3 corresponds to the inhomogeneous case. In Sim1 and Sim2, we take $\lambda_p = \lambda$ for all the branches p that go straight and $\lambda_p = \frac{\lambda}{2}$ for branches p which have a turn. Then, we sweep the value of λ over the range $[0.01, 0.5]$. In Sim3, we sweep λ_1 through the range $[0.01, 0.5]$ while the remaining λ_p are held fixed. In all simulations, for each set of λ_p's, 10 trials are conducted each with a simulation time equal to four times the cycle time obtained from the Webster's method for the corresponding rates of arrival of new vehicles.

In Sim1, we simulate the signal-based algorithm and the proposed hierarchical-distributed (HD) algorithm for different values of λ and with $W_T = 1$ in the cost function (3). Figures 3 and 4 show the mean and variance for the cars that cross the intersection during the simulation time (selected as described above) over 10 trials. Figures 3(a) and 3(b) show the average time to cross the intersection in the cases of the signal-based algorithm and our algorithm, respectively. Figures 3(c) and 3(d) show the average cost per car in the cases of the signal-based algorithm and our algorithm, respectively. Finally, Figures 4(a) and 4(b) show the average fuel cost per car for the signal-based algorithm and our algorithm, respectively. The main observation is that for low traffic conditions (small λ),

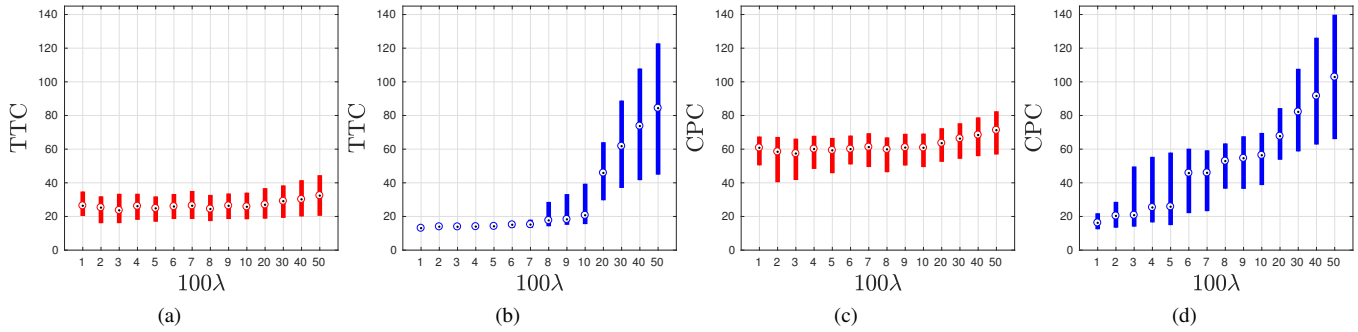

Fig. 3. Results of Sim1. Average time to cross the intersection (TTC) and the average cost per car (CPC), computed with $W_T = 1$ in (3), for the cars that cross the intersection over 10 trials. (a) and (c) are for the signal-based control and (b) and (d) plots are for the HD algorithm.

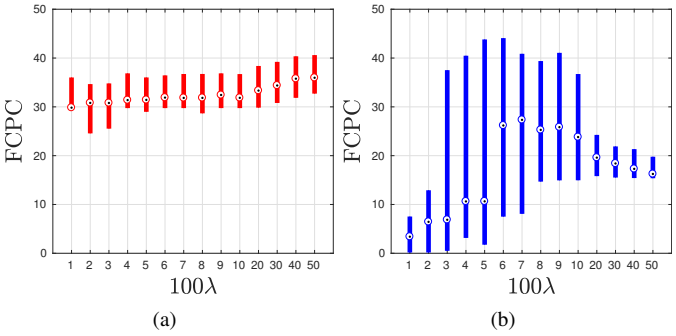

Fig. 4. Results of Sim1. Average fuel cost per car (FCPC) for the cars that cross the intersection over 10 trials. (a) is for the signal-based control and (b) is for the HD algorithm.

the proposed algorithm performs better than the signal-based algorithm. Further, up to $\lambda = 0.07$, the average time to cross the intersection is nearly fixed with very little dispersion. This indicates a critical arrival rate beyond which the intersection is saturated under the proposed algorithm. Beyond this critical arrival rate, the average time to cross the intersection increases very rapidly for the HD algorithm, which results in the average cost per car also rising rapidly. Nevertheless, for up to about $\lambda = 0.1$, the HD algorithm performs better (in terms of time and overall cost) than the signal-based algorithm. The fuel cost per car is in general lower for the HD algorithm. In general, automated intersection management can mimic signal-based control. Thus, simulations such as in Figure 3 can aid in identifying traffic regimes in which to switch to a signal-mimicking algorithm.

In Sim2, we perform simulations for different values of λ and W_T in the cost function (3), varied from 0.01 to 0.5 and 0.1 to 10, respectively. Figure 5 summarizes the results of Sim2. Figure 5(a) shows the average number of cars to cross the intersection (throughput) for the HD algorithm. Figures 5(b) and 5(c) show the ratio (average of 10 trials) of the number of cars that crossed for the signal-based algorithm over the number for the HD algorithm and the ratio of the (average over 10 trials) of the cost per car for the HD algorithm over the signal-base algorithm, respectively. Thus, in these two plots, the HD algorithm does better for those parameters for which the ratios are less than 1. The throughput is consistently better in the signal-based control except for low-density traffic (low λ). In terms of cost, except in the cases with very high density traffic (high λ) and high weightage to travel time in the cost function, the HD algorithm does better than the signal-based control.

Finally, in Sim3, we explore the case of inhomogeneous traffic arrival. In this simulation, we keep the arrival rates on 7 of the 8 branches fixed. On the four branches that turn left, λ_p is set to 0.01. For three of the branches that go straight, λ_p is set to 0.05. For the remaining branch that goes straight, $\lambda_1 = \lambda$ is swept through 0.01 to 0.5. The weight in the cost function (3) is fixed as $W_T = 1$. The results are presented in Figures 6(a) and 7(a). We see that the HD algorithm performs better than the signal-based algorithm even for large values of λ. The performance of the two is comparable for $\lambda = 0.5$.

In summary, we see that the proposed HD algorithm for intersection management performs very well under low traffic conditions. Beyond the saturation point, however, the algorithm performs increasingly (with vehicle arrival rate) worse than the signal-based algorithm. In a way, this is to be expected because the HD algorithm is purely reactive based on the instantaneous traffic scenario and is unaware of the arrival rates of the incoming traffic, whereas the signal timing in the signal-based method is optimized according to the arrival rate of new vehicles. This points out to the need of incorporating the arrival rates of new vehicle into the design of the HD algorithm to make it capable of dealing better with high traffic conditions. We should also note that, in all the simulations, we have assumed that the vehicles are computer controlled, with no communication delays and zero response time. This in itself results in a huge improvement in efficiency compared to the human-driven vehicle scenario of today (and ignores the commonly seen, significant lost time between phases). For example, under high traffic rate conditions, during the red phase there is a significant accumulation of vehicles at the intersection. Here, in contrast to present traffic behavior, there is very little lost time between phases. This is because each time a new green phase begins, a big group of vehicles uses the intersection as one *rigid body*.

IX. Conclusions

We have studied the problem of coordinating traffic at an intersection in order to reduce travel time and improve vehicle energy efficiency while avoiding collisions. Our provably

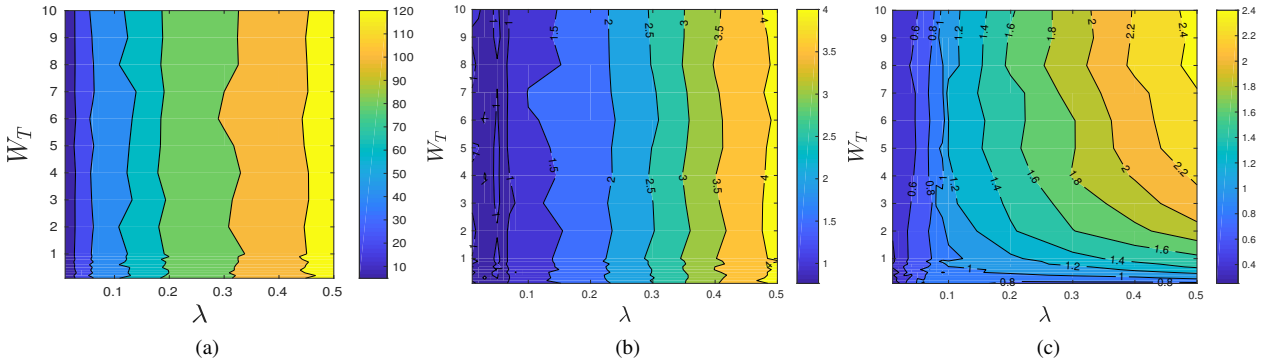

Fig. 5. Results of Sim2 for various values of λ and the weight W_T in the cost function (3). (a) Average (over 10 trials) number of cars that crossed the intersection for the HD algorithm. (b) The ratio of the average (over 10 trials) number of cars that crossed for the signal-based algorithm over that in the HD algorithm. (c) The ratio of the average (over 10 trials) cost taken for the cars that crossed the intersection for the HD algorithm over that in the signal-based algorithm.

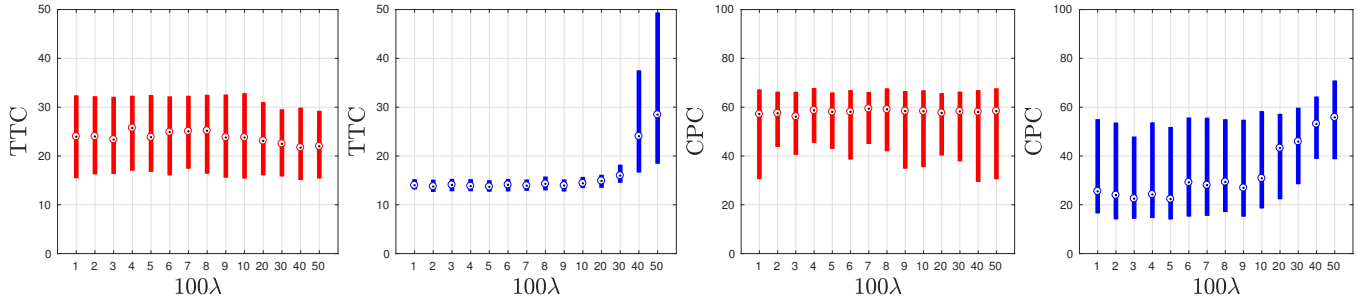

Fig. 6. Simulation results for Sim3 (inhomogeneous traffic arrival). Average time to cross the intersection (TTC) and the average cost per car (CPC), computed with $W_T = 1$ in (3), for the cars that cross the intersection over 10 trials. (a) and (c) are for the signal-based control and (b) and (d) plots are for the HD algorithm.

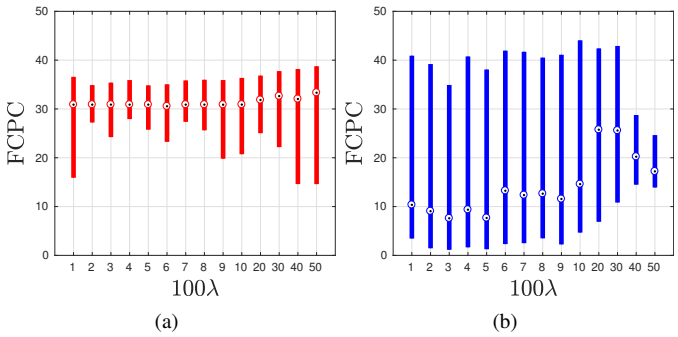

Fig. 7. Simulation results for Sim3 (inhomogeneous traffic arrival). Average fuel cost per car (FCPC) for the cars that cross the intersection over 10 trials. (a) is for the signal-based control and (b) is for the HD algorithm.

correct intersection management solution relies on communication among vehicles and the infrastructure, and combines hierarchical and distributed control to optimally schedule the passage of vehicle bubbles through the intersection. Our dynamic bubble-based approach has the advantage of reducing the complexity of the computationally intensive scheduling problem and making the solution applicable for different traffic conditions. Simultaneously, the modular nature of the major aspects of our design eases the possibility of improvements in the future. Finally, since the central traffic manager at the intersection requires only aggregate data of a bubble, this decomposition provides a certain amount of privacy. We have performed simulations to illustrate the performance of our design and compared it against an optimized signal-based intersection management approach. Our hierarchical-distributed algorithm performs better than signal-based control in terms of cost except for high traffic densities and high weightage to travel time in the cost function. The guaranteed throughput is, however, worse due to the conservativeness of the upper bound on inter-approach times of the vehicles. We believe further analysis would improve this component and yield better throughput. Other future work will study the incorporation of information about incoming traffic density to improve throughput in high traffic conditions, the generalization of the model, and the concept of bubble in particular, to also cars driven by people, the inclusion of privacy preservation requirements. We are also interested in studying the trade-offs between bubble size and the suboptimality of the results. Finally, although traffic management at isolated intersections is useful in many scenarios, the extension to coordinated management for networks of intersections is a non-trivial and important problem, which we will explore in the future.

ACKNOWLEDGMENTS

The research was supported by NSF Award CNS-1446891 and AFOSR Award FA9550-15-1-0108.

REFERENCES

[1] P. Tallapragada and J. Cortés, "Coordinated intersection traffic management," *IFAC-PapersOnLine*, vol. 48, no. 22, pp. 233–239, 2015. *IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Philadelphia, PA.

[2] E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune, "Supervisory control for collision avoidance in vehicular networks using discrete event abstractions," in *American Control Conference*, (Washington, D.C.), pp. 4380–4386, 2013.

[3] E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune, "Supervisory control for collision avoidance in vehicular networks with imperfect measurements," in *IEEE Conf. on Decision and Control*, (Florence, Italy), pp. 6298–6303, 2013.

[4] A. Colombo and D. Del Vecchio, "Least restrictive supervisors for intersection collision avoidance: A scheduling approach," *IEEE Transactions on Automatic Control*, vol. 60, no. 6, pp. 1515–1527, 2015.

[5] M. R. Hafner, D. Cunningham, L. Caminiti, and D. Del Vecchio, "Cooperative collision avoidance at intersections: Algorithms and experiments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1162–1175, 2013.

[6] G. R. de Campos, F. D. Rossa, and A. Colombo, "Safety verification methods for human-driven vehicles at traffic intersections: Optimal driver-adaptive supervisory control," *IEEE Transactions on Human-Machine Systems*, vol. 48, no. 1, pp. 72–84, 2018.

[7] A. Colombo, G. de Campos, and F. D. Rossa, "Control of a city road network: distributed exact verification of traffic safety," *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 4933–4948, 2017.

[8] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *Journal of Artificial Intelligence Research*, vol. 31, pp. 591–656, 2008.

[9] D. Fajardo, T. Au, S. T. Waller, P. Stone, and D. Yang, "Automated intersection control," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2259, pp. 223–232, 2011.

[10] H. Kowshik, D. Caveney, and P. R. Kumar, "Provable systemwide safety in intelligent intersections," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 3, pp. 804–818, 2011.

[11] R. Hult, G. Campos, P. Falcone, and H. Wymeersch, "An approximate solution to the optimal coordination problem for autonomous vehicles at intersections," in *American Control Conference*, (Chicago, IL), pp. 763–768, 2015.

[12] G. R. Campos, P. Falcone, H. Wymeersch, R. Hult, and J. Sjöberg, "Cooperative receding horizon conflict resolution at traffic intersections," in *IEEE Conf. on Decision and Control*, (Los Angeles, CA), pp. 2932–2937, Dec. 2014.

[13] M. A. S. Kamal, J. Imura, T. Hayakawa, A. Ohata, and K. Aihara, "A vehicle-intersection coordination scheme for smooth flows of traffic without using traffic lights," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1136–1147, 2015.

[14] F. Altché and A. D. L. Fortelle, "Analysis of optimal solutions to robot coordination problems to improve autonomous intersection management policies," in *IEEE Intelligent Vehicles Symposium*, pp. 86–91, IEEE, 2016.

[15] M. Levin and D. Rey, "Conflict-point formulation of intersection control for autonomous vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 528–547, 2017.

[16] X. Qian, J. Gregoire, F. Moutarde, and A. D. L. Fortelle, "Priority-based coordination of autonomous and legacy vehicles at intersection," in *IEEE International Conference on Intelligent Transportation Systems*, (Qingdao, China), pp. 1166–1171, 2014.

[17] D. Miculescu and S. Karaman, "Polling-systems-based control of high-performance provably-safe autonomous intersections," in *IEEE Conf. on Decision and Control*, (Los Angeles, CA), pp. 1417–1423, Dec. 2014.

[18] D. Miculescu and S. Karaman, "Polling-systems-based autonomous vehicle coordination in traffic intersections with no traffic signals," *arXiv preprint arXiv:1607.07896*, 2016.

[19] A. Malikopoulos, C. Cassandras, and Y. Zhang, "A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections," *Automatica*, vol. 93, pp. 244–256, 2018.

[20] Q. Jin, G. Wu, K. Boriboonsomsin, and M. Barth, "Advanced intersection management for connected vehicles using a multi-agent systems approach," in *IEEE Intelligent Vehicles Symposium*, (Alcalá de Henares, Spain), pp. 932–937, 2012.

[21] Q. Jin, G. Wu, K. Boriboonsomsin, and M. Barth, "Platoon-based multi-agent intersection management for connected vehicle," in *IEEE International Conference on Intelligent Transportation Systems*, (The Hague, Holland), pp. 1462–1467, 2013.

[22] R. Tachet, P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing, and C. Ratti, "Revisiting street intersections using slot-based systems," *PLOS One*, vol. 11, no. 3, p. e0149607, 2016.

[23] L. Li, D. Wen, and D. Yao, "A survey of traffic control with vehicular communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 425–432, 2014.

[24] P. Tallapragada and J. Cortés, "Distributed control of vehicle strings under finite-time and safety specifications," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1399–1411, 2018.

[25] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982. Presented at the 1957 Institute for Mathematical Statistics Meeting.

[26] R. Diestel, *Graph Theory*, vol. 173 of *Graduate Texts in Mathematics*. Springer, 2 ed., 2000.

[27] F. Bullo, J. Cortés, and S. Martinez, *Distributed Control of Robotic Networks*. Applied Mathematics Series, Princeton University Press, 2009. Electronically available at http://coordinationbook.info.

[28] A. Papoulis and S. U. Pillai, eds., *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 2002.

[29] T. Urbanik, A. Tanaka, B. Lozner, E. Lindstrom, K. Lee, S. Quayle, S. Beaird, S. Tsoi, P. Ryus, D. Gettman, S. Sunkari, K. Balke, and D. Bullock, *Signal Timing Manual*. Transportation Research Board, 2nd ed., 2015.

Pavankumar Tallapragada received the B.E. degree in Instrumentation Engineering from SGGS Institute of Engineering & Technology, Nanded, India in 2005, M.Sc. (Engg.) degree in Instrumentation from the Indian Institute of Science, Bangalore, India in 2007 and the Ph.D. degree in Mechanical Engineering from the University of Maryland, College Park in 2013. He held a postdoctoral position at the University of California, San Diego during 2014 to 2017. He is currently an Assistant Professor in the Department of Electrical Engineering at the Indian Institute of Science, Bengaluru, India. His research interests include event-triggered control, networked control systems, distributed control and networked transportation systems.

Jorge Cortés (M'02-SM'06-F'14) received the Licenciatura degree in mathematics from Universidad de Zaragoza, Zaragoza, Spain, in 1997, and the Ph.D. degree in engineering mathematics from Universidad Carlos III de Madrid, Madrid, Spain, in 2001. He held postdoctoral positions with the University of Twente, Twente, The Netherlands, and the University of Illinois at Urbana-Champaign, Urbana, IL, USA. He was an Assistant Professor with the Department of Applied Mathematics and Statistics, University of California, Santa Cruz, CA, USA, from 2004 to 2007. He is currently a Professor in the Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA, USA. He is the author of Geometric, Control and Numerical Aspects of Nonholonomic Systems (Springer-Verlag, 2002) and co-author (together with F. Bullo and S. Martínez) of Distributed Control of Robotic Networks (Princeton University Press, 2009). At the IEEE Control Systems Society, he has been a Distinguished Lecturer (2010-2014), and is currently its Director of Operations and an elected member (2018-2020) of its Board of Governors. His current research interests include distributed control and optimization, network science, resource-aware control, nonsmooth analysis, reasoning and decision making under uncertainty, network neuroscience, and multi-agent coordination in robotic, power, and transportation networks.