CISCAV: Consensus-based Intersection Scheduling for Connected Autonomous Vehicles

Emanuel Regnath
emanuel.regnath@tum.de
Technical University of Munich, Germany

Markus Birkner
markus.birkner@tum.de
Technical University of Munich, Germany

Sebastian Steinhorst
sebastian.steinhorst@tum.de
Technical University of Munich, Germany

***Abstract*—Intelligent vehicles that autonomously plan, communicate, and perform intersection crossings will reduce accidents and delays compared to human drivers. Previously proposed solutions require the use of additional expensive infrastructure, such as centralized Intersection Managers, or are based on theoretical control optimization with over-optimistic predictability of each vehicle's behavior.**

In this paper, we propose a decentralized intersection management that requires no additional infrastructure and can tolerate timing deviations due to unpredictable but detectable events, such as pedestrian movement or ambulances. Vehicles cooperate via direct VANET communication to agree on a schedule that specifies the groups and the order in which approaching vehicles will cross the intersection.

We have implemented our protocol as simulation using *Artery* and *SUMO*. Our experiments show that CISCAV ensures safety and reduces the average delay in high traffic conditions to $32\,\mathrm{s}$ compared to $150\,\mathrm{s}$ for conventional crossing policies such as traffic lights or priority roads.

***Keywords*—Traffic Management, CPS, SUMO, IoT**

I. Introduction

Road intersections are not only a hot spot for accidents but also cause delays and congestions, especially in urban areas.

According to [1], drivers in the US spend an additional 100 hours per year in their vehicles due to congestion and these delays have caused additional costs of 88 billion USD in 2019. This includes the costs for higher fuel consumption, higher wear at certain components and additional costs due to longer delivery times for goods. Apart from the financial impact and delays, the congestions also contribute to environmental damage and diminish the life quality in urban areas due to pollution and smog.

One way to increase the efficiency and safety at intersections is the installation of an Intersection Manager (IM) unit at every intersection, which performs the scheduling of all autonomous vehicles. This approach may be suitable in areas with high traffic volume but it requires a high financial and logistical effort to install such a unit at every intersection in rural areas [2]. Furthermore, a centralized intersection management introduces a single point of failure, which would require a high amount of maintenance and redundancy to guarantee continuous operation.

Another approach could be a cloud-based management engine, which virtually maps every existing intersection and acts as a scheduling service to which all vehicles on the road connect. However, this idea requires a continuous Internet connection via cellular towers and is very sensitive to (temporary) communication failures such as packet collisions and network delays.

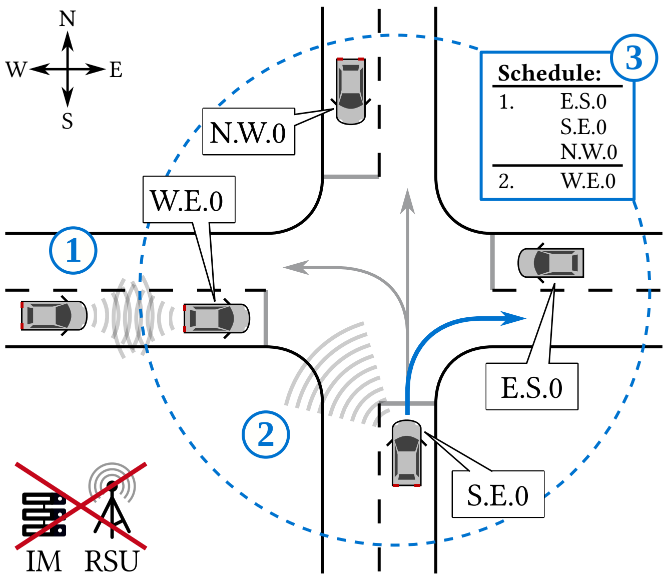

Figure 1: Vehicles agree on schedule groups using three communication phases: ① Intra-Lane Exchange, ② Inter-Lane Exchange, and ③ Schedule Group Consensus. No Road-Side Units (RSU) or central Intersection Managers (IM) are required.

By contrast, a decentralized approach in which vehicles communicate directly with each other would be much more robust to individual failures and does not require any additional communication infrastructure.

A. Scope and Contributions

This paper addresses the scheduling problem for intersections and the required communication on an architectural level. We assume that once the vehicles have agreed on a schedule for crossing the intersection, they can execute a safe crossing autonomously. Therefore, we do not cover the details of physical driving maneuvers such as controller inputs, sensor uncertainties, or data processing.

Instead, we focus on the problem of establishing and agreeing on the order in which vehicles should cross the intersection under the following assumptions:

- Not all vehicles are autonomous but autonomous vehicles are able to detect human-driven vehicles via sensors.
- Autonomous vehicles can directly communicate with each other and there exist no Road-Side Units (RSU).
- Environmental events can force vehicles to slow down or deviate in other ways from the schedule (semi-deterministic).

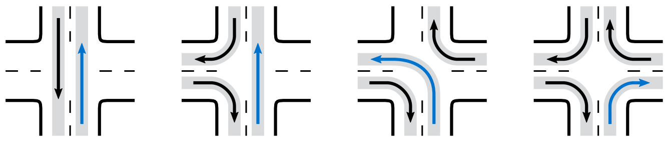

Figure 2: Possible groups of vehicles that can cross an intersection simultaneously. Rotated versions are not shown but are also valid.

- Vehicles are cooperative but selfish, which means they could send fake data if they gain an advantage (semi-cooperative).

We introduce a complete decentralized and offline approach which only uses direct wireless communication between the participating vehicles. Our goal is an algorithm which works even in rural areas without the need for cellular coverage, central traffic management systems, or access to cloud infrastructure. In particular, we

- propose a distributed intersection management scheme which uses consensus to agree on schedule groups (Section III),
- provide a C++ implementation for the realistic simulation framework SUMO (Section IV),
- discuss related approaches from literature and why they are over-optimistic (Section V),
- simulate and evaluate safety and delay (Section VI).

II. Assumptions and Models

Our approach is based on the following assumptions and models, and in the remainder of this paper, we will implicitly make use of them.

A. Intersection Model

We consider an intersection with four roads $R = \{rN, rE, rS, rW\}$ where each road is labeled according to its cardinal direction: rN (north) for the top road, east, south, and west accordingly. Each road has two lanes, one for incoming traffic and one for outgoing traffic. There exists an obligation to drive on the right side of the road. Figure 1 depicts the intersection and the nomenclature of the roads. The intersection is embedded in the center of an area of 400×400 meters, thus each incoming and outgoing lane has a length of 200 meters.

B. Vehicle Model

Vehicles are either 1. human-driven and non-communicating or 2. Connected and Autonomous Vehicles (CAVs). CAVs are able to communicate with each other over a direct, short-range VANET connection, which does not require any additional infrastructure such as Road-Side Units (RSU) or cellular radio towers. Furthermore, CAVs are able to detect human-driven vehicles by onboard sensors.

C. Problem Scenario

A set $V = \{v_1, v_2, ...v_N\}$ of $N \in \mathbb{N}$ vehicles arrives at the intersection. Each vehicle is defined as $v_i = rX.rY.Z$, with the current incoming road $rX \in R$, the desired outgoing road $rY \in R$, and the index $Z \in \mathbb{N}_0$, which counts the vehicles on each incoming road. For example, $v_i = W.S.0$ is the first vehicle on the west road that will arrive at the intersection desiring a right turn towards south. In this paper, the four cardinal directions are used for an intuitive discussion but any number of roads with unique identifier mappings could be used.

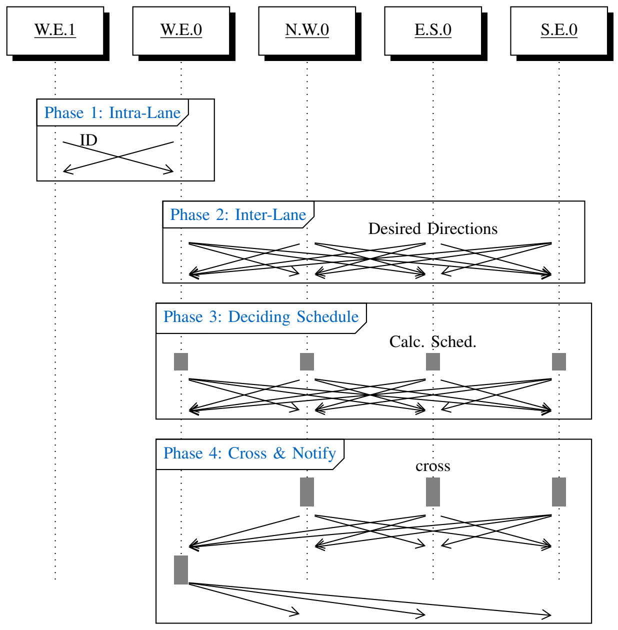

Figure 3: Sequence diagram of our consensus protocol. Arrows indicate messages sent over VANET. Gray bars indicate the time needed for computation or executing an operation.

III. Our Approach

The main idea of this paper is the use of schedule groups specifying the sequence in which vehicles will cross the intersection without exact timing information. Excluding timing requirements from the schedule allows vehicles to focus only on the order and right of way independent of individual vehicle properties that are time-sensitive, such as agility, reaction time, or accuracy of location estimation. Furthermore, excluding time also allows the protocol to tolerate unexpected events that cause vehicles to slow down or stop such as pedestrians or ambulances.

A. Schedule Groups

A schedule group consists of a set of vehicles and assigns each vehicle an order value based on their desired direction, which would be the outgoing lane ID (= cardinal point for simplicity). The combination of directions corresponds to a predefined ordering, which is stored in a look-up table. Vehicles with the same order value can cross the intersection simultaneously. Once all vehicles of the same order have crossed the intersection (and broadcast that), the next order vehicles start crossing until all vehicles of the schedule group have crossed the intersection. For our 4-leg intersection example, Figure 2 shows the possible combinations of vehicles that can cross together and Table I shows examples of concrete schedule groups.

Group	Members	Pattern	Schedule	Order of Crossings
0	carW.E.0	SRLR	2111	1. car N.W.0
	carN.W.0			car E.S.0
	carE.S.0			car S.E.0
	carS.E.0			2. car W.E.0
1	carW.E.1	S—	1—	1. carW.E.1

Table I: The two schedule groups that are formed based on the scenario from Figure 1. The direction pattern is **S**traight-**R**ight-**L**eft-**R**ight which results in a 2111 schedule (predefined look-up table). The first three vehicles of schedule group 0 can cross simultaneously.

B. Virtual MiniMap

While approaching the intersection, other detected vehicles will be added to the list of known vehicles. The state information of all known vehicles will be stored in a virtual MiniMap, which will be continuously updated over time. For example, ID (license plate), velocity, desired direction and index position towards the intersection will be stored here. The index of each vehicle corresponds to the number of vehicles on the same lane that will cross the intersection before them. Therefore, the vehicle next to the intersection has index 0 and the vehicle behind it has index 1. Note that the indices only establish a relative order on each incoming road but are not synchronized with other roads. Once a larger gap occurs and the next vehicle is outside of the communication range of the previous vehicle, it will start with index 0 again.

C. CISCAV Protocol Description

The CISCAV Protocol (**C**onsensus-based **I**ntersection **S**cheduling for **C**onnected **A**utonomous **V**ehicles) is a sequence of the following four phases. Any autonomous vehicle which approaches an intersection has to execute all four phases to reach a consensus on a schedule of crossings.

Phase One: Intra-Lane Information Exchange Before getting near the intersection, each vehicle will already exchange messages with surrounding vehicles on the same lane during driving. The acquired information such as desired direction, velocity, and position of vehicles is stored in the virtual MiniMap.

Phase Two: Inter-Lane Information Exchange Vehicles within $150\,\mathrm{m}$ to the intersection will broadcast messages to exchange information with vehicles on other incoming roads. This new information will also be stored in the MiniMap, which will incrementally create an ordered list of all vehicles for each incoming road.

Phase Three: Deciding Schedule An optimized schedule is created based on the concept of schedule groups. All vehicles which are the first ones on each incoming road that are not already part of an existing schedule group form a new schedule group. Based on the pattern of desired directions, a schedule order is calculated as shown in Table I. Afterwards, every member of a schedule group repeatedly broadcasts the schedule group and updates the MiniMap with received broadcasts. Once all members have received a broadcast with the same schedule from *all* other members, they lock the schedule group.

CAVs which do not participate in this phase will not gain any advantage because they would block themselves as well.

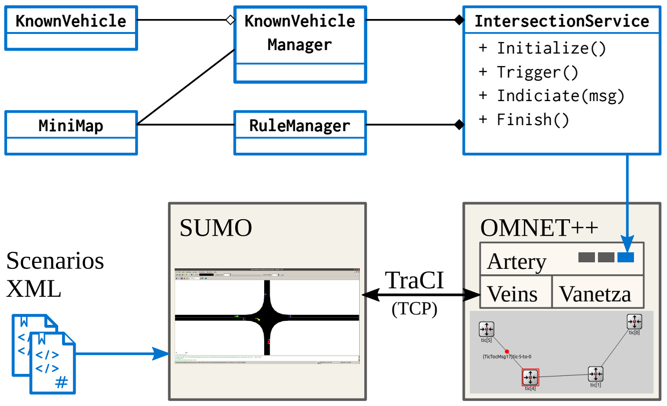

Figure 4: Illustration of our software architecture and its interaction with the simulation environment. The XML scenario files specify the random traffic flows for SUMO, which will spawn vehicles accordingly. Our protocol runs as service in Artery.

Phase Four: Cross & Notify Once all vehicles from one schedule group are next at the intersection, they cross the intersection based on their agreed schedule. They continue to broadcast results and positions to inform new arriving vehicles on the agreed schedule and the states of other vehicles.

Figure 3 shows the message exchange during each phase. Algorithms 1 and 2 describe the pseudo code of our implementation that is discussed in Section IV.

D. Consensus Properties

CISCAV ensures that vehicles agree on the schedule group. In conventional consensus protocols, such as PBFT [3], participants can start in a conflicting state and the protocol must converge to agreement. In our situation, we already have a fixed set of schedules (look-up table) such that every correct vehicle will calculate the same schedule. As discussed in [4], consensus for traffic decisions cannot tolerate any failing vehicle. As soon as a conflict arises, the consensus fails and needs to be repeated until unanimous agreement is reached or otherwise resolved manually.

The only parameter that can cause disagreement is the set of vehicles in a schedule group because some vehicles could try to join when the agreement round has already started. In case a vehicle has decided on a schedule group, it will no longer accept any proposals or votes to extend or change a schedule group. In case a vehicle has not decided yet, the currently proposed schedule group can be extended (but never reduced). Extensions can happen until there is a vehicle from every incoming road in the schedule group. After that there are no further updates possible and every correct vehicle will calculate the same schedule group.

E. Handling Human-Driven Vehicles

Since there is no synchronous timing requirement for the schedules, CISCAV can tolerate human-driven vehicles as well. We assume that human-driven cars can be detected in Phase 2 because they appear on sensors (e.g. Camera, LIDAR) but do not respond to messages. Because of this, it will not be possible for autonomous vehicles to form a schedule

Algorithm 1: `Trigger()`

```
Data: myData = {currRoadId, nextRoadId, frontVeh,
        backVeh, ScheduleGroup}, MiniMap
1  if not near intersection then
2      if frontVeh or backVeh are unset then
3          broadcast myData;
4  if ScheduleGroup is empty then
5      create ScheduleGroup from MiniMap;
6      broadcast myData
7  else if ScheduleGroup is non empty then
8      broadcast ScheduleGroup
9  else if ScheduleGroup is final then
10     while others before me in ScheduleGroup do
11         wait for notifications
12     cross intersection
```

Algorithm 2: `Indicate(IntersectionMessage)`

```
Data: newVeh from recv. IntersectionMessage, MiniMap,
        myData
1  if not near intersection then
2      if newVeh not in MiniMap then MiniMap.add(newVeh);
3      if newVeh in front of me then
4          set myData.frontVeh = newVeh
5      else if newVeh behind me then
6          set myData.backVeh = newVeh
7  else if near intersection then
8      if my ScheduleGroup is final then return;;
9      if newVeh.ScheduleGroup equals my ScheduleGroup
        then
10         set newVeh as agreeing
11         if all vehicles in my ScheduleGroup agree then
12             mark ScheduleGroup as final;
13     else if newVeh.ScheduleGroup extends my
        ScheduleGroup then
14         update my ScheduleGroup
15     else
16         keep my ScheduleGroup
```

group and, as a result, they will fall back to conventional traffic policies. Only in case that all the vehicles next at the intersection are CAVs, a schedule group is formed. In this case the vehicles deviate from the conventional traffic policies and perform a more efficient crossing. In this manner, CISCAV can tolerate any percentage of human-based vehicles and will "kick in" once only CAVs meet.

IV. Implementation

We have implemented CISCAV in `C++` as a message handling service for the simulation frameworks Artery and SUMO, which aim for a realistic modeling on several abstraction layers. Our overall architecture is depicted in Figure 4.

A. Used Frameworks

In detail, we have build CISCAV on top of the following framework stack (from physical- to communication-level):

SUMO or "Simulation of Urban MObility" simulates road traffic on various road network geometries. The geometries are either exported from open street map data or are created by hand. Our custom-designed traffic flow is defined in XML-files which are loaded by the simulator. The application either runs in combination with V2X communication system or it is able to run independently where traffic flow is simulated without communication among the created vehicles [5].

OMNeT++ is a discrete event simulator for network communication [6].

Vanetza implements the ETSI C-ITS protocol suite and supports the specified communication standards [7].

Veins simulates Inter-Vehicular Communication. It connects OMNeT++ and SUMO [8] .

Artery The V2X simulation framework enables communication based on the ETSI ITS-G5 specification. It is the highest abstraction of all the underlying communication and vehicular simulations [9].

B. Software Details

Our main class `IntersectionService` inherits from Artery's `ItsG5Service`, which allows us to send messages and provides callbacks for the initialization of vehicles, receiving of messages, and finalization (simulation end) of vehicles. We use or overwrite these functions to include our protocol. Every vehicle runs its own instance of our service class.

- `Initialize`: Called on vehicle creation. We initialize our internal data structures such as `roadID` and schedule groups with the vehicle itself as the only member.
- `Trigger` (Algorithm 1): Called periodically by the simulation. Here, we broadcast periodic beacons for discovering other approaching vehicles. Additionally, a schedule group is created with all the first vehicles on the other incoming roads which are in a certain range close to the intersection and are not already part of a schedule group. Each position must be verified by at least one other vehicle in order to be accepted in a schedule group. Updated information is broadcast at the end of the function call.
- `Indicate` (Algorithm 2): Called on message reception. Every received message is processed in this function, which updates our data structure and creates a virtual map of the intersection. The updated data is used in the next call of Trigger.
- `Finish`: This function is called when a vehicle exits the simulation. It collects all the diagnostic data.

V. Related Work

A good overview of cooperative IM approaches is presented in two surveys [2] and [19] from 2016 while more recent developments are covered in the introduction of [20].

We found that we can classify existing approaches in roughly two types: 1. Exclusive Reservation and 2. Control Optimization. Each type can be further distinguished into *centralized*, if it uses a central manager which handles the calculation and communication, or *decentralized*, if vehicles calculate individually and communicate directly with each other.

Algorithm			Intersection		Vehicle				Results
Name	**Type**	**Simulation**	**S/L**	**Legs/Near**	**L** [m]	**V** $\left[\frac{\mathrm{m}}{\mathrm{s}}\right]$	**Acc.** $\left[\frac{\mathrm{m}}{\mathrm{s}^2}\right]$	**L-S-R** [%]	**Avg. Delay**
AIM08 [10]	Cent./Res.	Custom	+1	125 m/–	N/A	25.0	N/A	0-100-0	0.2 s@0.5 v/s/ln
Prio14 [11]	Cent./Res.	SUMO/10 min	+3	290 m/50 m	N/A	12.0	$-4\ldots2$	20-70-10	15 s@0.5 v/s/rd
Delay17 [12]	Cent./Res.	SUMO	+1	50 m/50 m	N/A	N/A	N/A...N/A	20-70-10	35 s@0.5 v/s/rd [1]
CSIP19 [13]	Cent./Res.	AutoSim	+2	500 m/N/A	2.6	13.4	N/A...N/A	0-33-66	≈0 s@0.5 v/s/rd
NoStopSign08 [14]	Dec./Res.	Custom	+1	125 m/75 m	N/A	N/A	N/A	15-70-15	20 s@0.4 v/s/ln
MP-IP12 [15]	Dec./Res.	AutoSim/1000 veh.	+2	N/A	< 5	N/A	$N/A\ldots3$	25-50-25	2 s@0.5 v/s/rd
MutEx14 [16]	Dec./Res.	NS-3	+2	50 m/50 m	N/A	N/A	N/A	N/A	19 s@0.5 v/s/rd
VTL15 [17]	Dec./Res.	SUMO+NS3	+4	200 m/N/A	N/A	16.7	$-6\ldots3$	N/A	35 s@N/A
DIMP18 [18]	Dec./Res.	SUMO	+(1/2)	500 m/358 m	5 m	13.41	$-2\ldots1$	N/A	17 s@0.5 v/s
CISCAV (ours)	Dec./Res.	SUMO / 3600 s	+1	200 m/100 m	4.3	13.9	$-7.5\ldots2.9$	33-33-33	32 s@0.5 v/s/rd

Table II: Overview of related work. **Type** is a combination of {Central/Decentral} and {Reservation/Control}. **S/L** indicates the shape of the intersection ($\vdash$, $+$, $\ast$) and the number of *incoming* lanes for each direction. **Legs/Near** is the length of the intersection roads per leg and the distance from the center at which vehicles start broadcasting messages. **L, V, Acc.** states the vehicle *L*ength; max. *V*elocity; and maximum deceleration (negative) and *Acc*eleration considered. **L-S-R** are the probabilities for a vehicle to turn left (L), go straight (S), or turn right (R) in percent. Each related approach did not mention at least one of these parameters (N/A).

Exclusive Reservation: The intersection is divided into discrete *resources*, such as *tiles* or pre-defined *paths*. Vehicles then negotiate and agree on a conflict-free reservation schedule where each vehicle gets exclusive access to resources until it crossed the intersection. Exclusive reservation is safe but does not optimize throughput as vehicles often reserve more resources than actually needed. Furthermore, it is compatible with human-driven cars, which simply can get their own reservation slot based on existing traffic rules.

Control Optimization: Instead of discrete reservation, vehicles try to calculate smooth and collision-free trajectories in the continuous spectrum of possible trajectories, which can be followed by accelerating or braking. The trajectory search is formulated as a control problem with certain constraints (collision-free) and certain optimization objectives (e.g. latency, smooth acceleration). While this approach yields good results in simulations, its computation is expensive (state explosion), introduces hard real-time requirements, and often does not consider failures, such as a vehicle unable to apply acceleration or braking as claimed.

For this reason, we will cover only *Exclusive Reservation* and focus on *decentralized* approaches as these are most comparable to our work. Table II summarizes our findings.

A. Centralized Exclusive Reservation

AIM08 [10] from 2008, divides the intersection into a grid of $n \times n$ tiles. Vehicles request a slot in space-time for crossing from a central intersection manager, which will then assign tiles to vehicles. Each vehicle sends a request including time-of-arrival (TOA), velocity, and size to the IM. The IM calculates a motion profile for the vehicle according to a policy, e.g. "First come, first serve" (FCFS), and sends it back to the vehicle. If no collision-free path can be found, the reservation request is rejected and the vehicle has to slow down, possibly resting until a trajectory becomes available.

Prio14 [11] provides dedicated lanes for left-turn, straight, and right-turn and maximizes vehicle speed. Instead of a custom simulator, Prio14 was implemented in SUMO.

Delay17 [12] provides a delay-tolerant centralized protocol, which considers communication delays and packet losses. Since it was also implemented in SUMO and focuses on real-world conditions, it is one of the few related works for which we consider results comparable to our work. The result stated in the table is for normal message delay (OMNeT). One of the main differences is their central IM, with all the previously mentioned drawbacks.

CSIP19 [13] tries to maximize throughput with a minimum gap between vehicles that is still comfortable for human passengers. Vehicles that approach the intersection are equally spaced and must keep a constant velocity to ensure that no vehicle needs to slow down while crossing the intersection.

B. Decentralized Exclusive Reservations

In **NoStopSign08** [14], each vehicle generates and continually broadcasts a CLAIM (includes direction, arrival time, and exit time), which "reserves" the next free slot that does not conflict with already received claims. If conflicts occur, vehicles can send CANCEL messages and/or updated CLAIM messages. Simulation reaches near-zero delay for spawn rates of $< 0.2\,\mathrm{v/s}$. However, due to the lack of agreement confirmations from other vehicles, the authors reported collisions when the packet loss exceeds 40%.

In **MP-IP12** [15], the intersection has two lanes per direction and is divided into 4×4 tiles. Each vehicle continuously broadcast its state (ENTER, CROSS, EXIT) and the tiles it will occupy updated over time. Vehicle receiving

[1] Authors just state 45s total travel time and we subtracted 10s to estimate the delay because 10s was the total travel time for the lowest spawn rate of 0.1v/s, for which we assume almost no interruptions.

ENTER/CROSS messages drive as far as possible into the intersection and only stop in front of conflicting tiles. Once crossed, vehicles broadcast EXIT to free the tiles. In the published videos of the updated algorithm [13], the vehicles are spawned in a *periodic* pattern with a distance that allows vehicles to cross alternatingly with minimal (hardly visible) speed variations. It remains unclear whether the protocol would work with random spawn patterns.

In **MutEx14** [16], vehicles broadcast their estimated arrival time at the intersection via *REQUEST* messages. Vehicles with shorter arrival times will respond with a *REJECT* message, blocking the vehicle from crossing until all vehicles with shorter arrival times have crossed. If no *REJECT* is received (timeout) or only *PERMIT* messages from vehicles with higher arrival times are received, the vehicle will cross the intersection. Evaluation is purely based on the network simulator NS-3 in combination with static time counting for vehicle movements (e.g. $4\,\mathrm{s}$ to turn left).

Virtual Traffic Light (**VTL15**), is a 2015 patented idea first described in [21] and improved in [17]. The first vehicle arriving at an intersection becomes a temporary IM and assigns itself a virtual red light. It stops and assigns virtual green lights to other vehicles. This simple and elegant idea provides reasonable delays but it is not robust as it does not consider selfish incentives, e.g. vehicles slowing down to *not* become the leader with the red light.

DIMP18 [18] uses clusters and the vehicle closest to intersection will reach an agreement on the schedule. They also limit deceleration to reduce passenger discomfort. However, they measure “waiting time” for vehicles without defining how it is calculated and it is unclear whether their spawn rates are per lane, per road, or per intersection in total.

VI. Evaluation

We evaluate our approach based on the most important metrics *safety* and *delay*. For us, the delay is the *additional time* for a vehicle to completely cross an intersection compared to a crossing where the vehicle has priority from the beginning and can cross without any interactions with other vehicles.

Related work sometimes measures *throughput*. However, this metric is difficult to define for low spawning rates because the exit rate is basically the entry rate. Based on our findings, throughput is only interesting if the spawn rate is maximum. Since intersections normally experience income rates below the maximum, we only measure delay.

A. Experimental Setup

In SUMO, we have created random traffic flows for our 4-leg intersection. Vehicles have equal probabilities ($33.\overline{3}\%$) for each direction: left, straight, or right. We randomly spawn vehicles at the beginning of the roads ($200\,\mathrm{m}$ from the center) with a given probability that is equal for all directions. For example, $0.5\,\mathrm{v/s/rd}$ means that on average 0.5 vehicles will be spawned per second at each road. We spawn vehicles over a period of $3600\,\mathrm{s}$ and run the simulation until every vehicle has crossed the intersection. We have performed this experiment for varying spawn probabilities over 4 crossing policies: Traffic light, priority road, left before right and CISCAV. To be able to compare our work against related approaches, which assume 100% autonomous vehicles, we have only spawned 100% CAVs. However, the design of CISCAV allows to also handle human-driven vehicles as mentioned earlier in Section III-E.

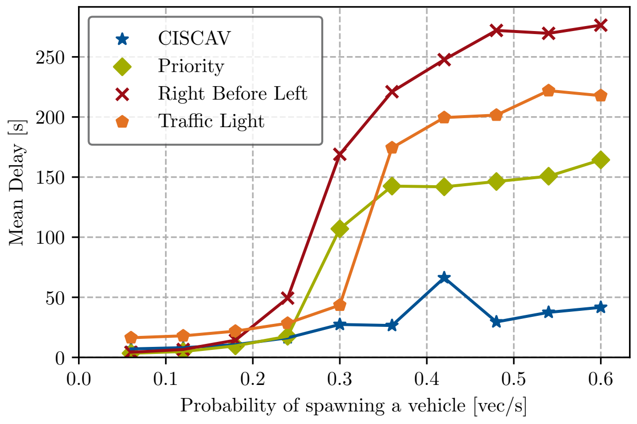

Figure 5: Average delay measurements for all four traffic policies over different spawn probabilities.

B. CISCAV results

During the simulation, no accidents occurred. While we did not explicitly stress-test our design with injected failures, it illustrates that random traffic flow is handled safely under normal conditions. Figure 5 shows the delays for each traffic policy. For a spawn rate of 0.5, CISCAV has an average delay of about $32\,\mathrm{s}$. This is a much higher value than the values reported by most related work. One of the reasons is that CISCAV is currently creating schedule groups in onion-like layers and does not set a lasting priority like a green light for a certain road. Another reason could be that related approaches have different definitions of delay or use over-optimistic assumptions. However, CISCAV outperforms all conventional traffic policies by far and illustrates how much optimization is possible by autonomous intersection scheduling.

C. Reproducing AIM08 Results

Since the AIM08 [10] is such an influential paper, we decided to reproduce results for our intersection model using their open-source AIM simulator version 1.0.3. We have simulated a single +1 intersection for $1000\,\mathrm{s}$ at a spawn rate per road of $1800\,\mathrm{v/h/rd} = 0.5\,\mathrm{v/s/rd}$ using $v_{\max} = 15\,\mathrm{m/s}$ (closest). The simulator outputs entry and exit times of the vehicles in seconds. We have calculated the delay as

$$t_{\text{delay}} = T_{\text{exit}} - T_{\text{entry}} - \Delta T_{\text{free}} \tag{1}$$

with $\Delta T_{\text{free}} = 11\,\mathrm{s}$ as the measured mean travel time from entry to exit for a vehicle without intersection.

To our surprise, the average delay is $\hat{t}_{\text{delay}} \approx 13.5\,\mathrm{s}$, which is far away from the $0.2\,\mathrm{s}$ reported in the original paper. This illustrates the high sensitivity of the delay with regard to simulation parameters such as lane numbers and vehicle velocity.

VII. Discussion

Performance of Related Work: While most related approaches report much lower delay times than CISCAV, they are often based on optimistic assumptions in the best case, and completely unrealistic assumptions in the worst case. For example, in some simulations of [13], vehicles approach the intersection equally spaced at a constant speed and keep minimum safety distances during the crossing. The tendency of some papers to optimize delays on idealized conditions has also been pointed out by [18]. Although performance is important, CISCAV's primary focus lies on reaching a safe agreement on exclusive reservation under unreliable conditions before any actions are taken. This results in delays that cannot compete with delays under ideal conditions but are still much lower than existing policies, such as traffic lights.

Weaknesses of CISCAV: In this first version of CISCAV, we create schedule groups in onion-like layers, which means vehicles from the same lane cannot cross the intersection directly after each other if the other lanes have also incoming vehicles. In high traffic conditions, this is not the most efficient way to reduce average delay and leads to frequent start-stop maneuvers when several vehicles approach the intersection from the same lane. Currently, we do not handle different vehicle types and the fact that large trucks might prevent certain schedule combinations and need to cross smaller intersections alone. Furthermore, our current implementation of CISCAV cannot resolve any permanent communication failures after schedule groups have been confirmed. That is, once a schedule is agreed and the last vehicle of the previous schedule group has left the communication range, the current group will wait for the missing crossing notification forever. However, this is no limitation of the protocol itself and can be resolved easily using sensors.

Strengths of CISCAV: CISCAV is completely decentralized and does not depend on any additional infrastructure. The use of schedule groups focuses purely on the order in which vehicles cross and therefore allows to tolerate arbitrarily vehicle timings and delays. It is designed in a way that allows to be transient for human-operated vehicles. This means that schedules, which deviate from the existing traffic rules, will only be created once only CAVs approach an intersection. We have implemented and evaluated CISCAV in the realistic simulation tool SUMO using C-ITS messages standardized by the EU. In its first version, it performs well in low load scenarios and outperforms current traffic rules.

VIII. Conclusion

CISCAV reaches a distributed agreement on exclusive reservation by creating schedule groups before any actions are taken, allowing it to tolerate arbitrary timing deviations as they can occur in real-world conditions.

In comparison to the sometimes over-optimistic related work, CISCAV gives realistic estimations for the delay of safe autonomous intersection crossing. Our measured average delay of $\leq 32\,\mathrm{s}$ at a very busy intersection supports the hypothesis that communication-based intersection handling can outperform currently existing traffic rules.

CISCAV does not require any additional infrastructure, makes conservative assumptions, and focuses on providing high safety guarantees, which is a fundamental requirement if we want to convince people in the near future to actually trust their lives on autonomous intersection management.

References

[1] INRIX, "Congestion Costs Each American Nearly 100 hours, $1,400 A Year," 03 2020, https://inrix.com/press-releases/2019-traffic-scorecard-us/.

[2] L. Chen and C. Englund, "Cooperative intersection management: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 570–586, 2016.

[3] M. Castro, B. Liskov, and Others, "Practical Byzantine Fault Tolerance," in *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, vol. 99, 1999, pp. 173–186.

[4] E. Regnath and S. Steinhorst, "Cuba: Chained unanimous byzantine agreement for decentralized platoon management," in *Design, Automation and Test in Europe (DATE 2019)*, March 2019, pp. 426–431.

[5] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using SUMO," in *The 21st IEEE Int. Conf. on Intelligent Transportation Systems*. IEEE, 2018.

[6] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," 01 2008, p. 60.

[7] R. Riebl, "Vanetza," June 2015, accessed: 20-05-06.

[8] C. Sommer, R. German, and F. Dressler, "Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis," *IEEE Transactions on Mobile Computing (TMC)*, vol. 10, no. 1, pp. 3–15, January 2011.

[9] R. Riebl, H. Günther, C. Facchi, and L. Wolf, "Artery: Extending Veins for VANET applications," pp. 450–456, 2015.

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

[11] X. Qian, J. Gregoire, F. Moutarde, and A. De La Fortelle, "Priority-based coordination of autonomous and legacy vehicles at intersection," in *17th international IEEE conference on intelligent transportation systems (ITSC)*. IEEE, 2014, pp. 1166–1171.

[12] B. Zheng, C.-W. Lin, H. Liang, S. Shiraishi, W. Li, and Q. Zhu, "Delay-aware design, analysis and verification of intelligent intersection management," in *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2017, pp. 1–8.

[13] S. Aoki and R. R. Rajkumar, "Csip: A synchronous protocol for automated vehicles at road intersections," *ACM Trans. Cyber-Phys. Syst.*, vol. 3, no. 3, 8 2019.

[14] M. VanMiddlesworth, K. Dresner, and P. Stone, "Replacing the stop sign: Unmanaged intersection control for autonomous vehicles," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3*, 2008, pp. 1413–1416.

[15] R. Azimi, G. Bhatia, R. Rajkumar, and P. Mudalige, "Intersection management using vehicular networks," SAE Technical Paper, Tech. Rep., 4 2012.

[16] W. Wu, J. Zhang, A. Luo, and J. Cao, "Distributed mutual exclusion algorithms for intersection traffic control," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 1, pp. 65–74, 2014.

[17] J. Shi, C. Peng, Q. Zhu, P. Duan, Y. Bao, and M. Xie, "There is a will, there is a way: A new mechanism for traffic control based on vtl and vanet," in *2015 IEEE 16th International Symposium on High Assurance Systems Engineering*. IEEE, 2015, pp. 240–246.

[18] X. Liang, T. Yan, J. Lee, and G. Wang, "A distributed intersection management protocol for safety, efficiency, and driver's comfort," *IEEE internet of things journal*, vol. 5, no. 3, pp. 1924–1935, 2018.

[19] J. Rios-Torres and A. A. Malikopoulos, "A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1066–1077, 2016.

[20] A. Mirheli, M. Tajalli, L. Hajibabai, and A. Hajbabaie, "A consensus-based distributed trajectory control in a signal-free intersection," *Transportation research part C: emerging technologies*, vol. 100, 2019.

[21] M. Ferreira, R. Fernandes, H. Conceição, W. Viriyasitavat, and O. K. Tonguz, "Self-organized traffic control," in *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*, 2010.