

# Win-Fit: Efficient Intersection Management via Dynamic Vehicle Batching and Scheduling

Guang Chen, Kyoung-Don Kang  
Department of Computer Science  
State University of New York at Binghamton  
{gchen21,kang}@binghamton.edu

**Abstract**—Intersection management is one of the most challenging tasks in traffic control. In this paper, to significantly decrease the average delay for autonomous vehicles to cross an intersection without any collision, vehicles are dynamically grouped and a selected winner group is allowed to drive through the intersection as one batch unlike most existing approaches, which only schedule individual autonomous vehicles one by one. Further, certain vehicles in the non-winner groups are allowed to safely cross the intersection during idle time slots (if any) unused by the winner group to further decrease the delay. Our approach has the lower-bound time complexity, and decreases the average trip delay to cross a simulated intersection by 31% - 95% compared to several state-of-the-art baselines.

## I. INTRODUCTION

In traffic control, intersection management is one of the most challenging tasks. In the USA, 787,236 intersection-related vehicle crashes, 36% of all the reported ones, were observed in 2010 [1]. A number of approaches have been developed to improve the intersection throughput, i.e., the total number of cars that pass an intersection per unit time, for cooperative autonomous vehicles, while avoiding collisions. Vehicle-to-vehicle (V2V) or vehicle-to-infrastructure (V2I) communications based DSRC [18] and WAVE [19] standards are utilized by autonomous vehicles to interact with each other or an infrastructure node to exchange the driving information and share the intersection state to decide, if each vehicle can drive through the intersection or it has to stop based on certain criteria [2-6,9,14,15]. In most existing approaches, including [2-6,9,14,15], intersection scheduling decisions are made on the basis of *individual vehicles* in either a centralized or distributed manner.

In this paper, we show that approaches that only consider individual vehicles one by one largely increase the average trip delay especially when many vehicles from different approach lanes need to cross a busy intersection. To address the problem, we present a new approach, called Win-Fit, which *dynamically groups autonomous vehicles and schedules them as batches to significantly decrease the average intersection trip delay*, while avoiding collisions. In our approach, the vehicles in each lane forms one or more groups based on the inter-car distance. In each lane approaching the intersection, the first group closest to the intersection is determined. The lead vehicle in the first group reports the group information, such as the speed, the distance to the intersection, the number and headings of the cars in the group, and the physical length of the group, to the intersection management agent (IMA) that runs our winner

selection algorithm, called Win, in the infrastructure node. Given the information, Win picks the first group in one of the approach lanes as the winner, *if the group can entirely pass the intersection as one batch incurring the shortest average waiting time for the cars in the other approach lanes*. By doing this instead of handling individual cars from all directions one by one, we decrease the chance for collisions/conflicts between cars coming from different directions and the amount of the required waiting time for safety, while avoiding potential starvation of the cars in the low priority (non-winner) groups. In addition, we devise another algorithm, called Fit, which cooperates with Win to allow some vehicles of the low priority groups to *safely enter the cells in the intersection unoccupied by the winner group during its idle time slots* (if any) to further decrease the overall intersection trip delay. The IMA broadcasts the intersection trip schedule derived by Win-Fit to the first group in each approach lane. Win-Fit is re-executed right before the winner group leaves the intersection.

Win-Fit is lightweight. Its time complexity is  $O(n)$  for  $n$  vehicles that need to cross the intersection. Note that this is the lower bound time complexity, since each car has to be considered at least once to determine when it can travel the intersection safely. For performance evaluation, we have implemented Win-Fit and several state-of-the-art baselines for intersection management in a popular open source traffic simulator, SUMO [17]. Compared to the baselines, our approach reduces the average intersection trip delay, i.e., the average time taken by a vehicle to pass the intersection, by 31% - 95% by dynamically grouping and scheduling vehicles as batches, and utilizing idle intersection cells unused otherwise.

In Section II, the background and motivation for Win-Fit are described. In Section III, an overview of Win-Fit is given. In Sections IV and V, the Win and Fit algorithms are described, respectively. Also, the time complexity is analyzed. In Section VI, the performance of our approach is compared to those of several baselines via a simulation study. Related work is discussed in Section VII. Finally, the paper is concluded and future work issues are discussed in Section VIII.

## II. BACKGROUND AND MOTIVATION

### A. Background

In [2], it is shown that there are two ways to increase the intersection throughput:

- Increase the average number of vehicles inside the intersection at any moment of time.
- Decrease the average time a vehicle spends at the intersection.

The average number of vehicles is maximized, if each vehicle can drive through the intersection without having to slow down or stop. However, this may not always be possible, since some cars may have to wait outside the intersection to ensure that no vehicles will collide with each other. If the waiting time of the vehicles can be decreased without raising safety issues, the average trip delay will be decreased. As a result, the traffic throughput will be increased. In this paper, we devise Win-Fit to decrease the intersection trip delay and increase the number of cars in the intersection, while avoiding crashes.

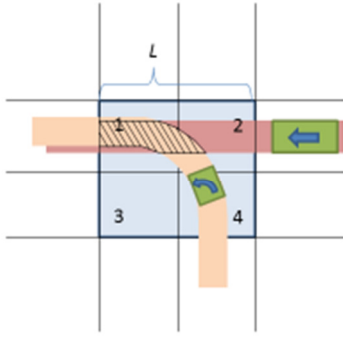


Fig. 1. Conflicting cell example

In this paper, without loss of generality, we consider a 4-way square intersection of width  $L$  divided into four cells of the same size as shown in Figure 1. Each direction has one approach lane for coming vehicles and another lane for leaving vehicles. The track of a vehicle is the list of the cells it needs to drive through in sequence to cross the intersection. If two vehicles from different directions have overlapped tracks, e.g., the shaded area in Figure 3, all the cells in the overlapped track are defined as conflicting cells (CCs) for them. They can collide if they simultaneously enter any CC. For example, the CCs for the two vehicles in Figure 3 are cells 1 and 2.

In this paper, we require each vehicle to maintain a constant velocity  $v$  at the intersection except for the following cases.

- A vehicle has to decelerate and stop at the edge of the intersection, i.e., the border between an approach lane and the intersection, if necessary, to avoid a collision with a conflicting vehicle.
- A vehicle has to decelerate and stop when the vehicle in front of it stops.
- A vehicle can accelerate upon leaving the intersection.

In our approach, a vehicle approaching the intersection measures its distance to the next car in front using, for example, an in-car Lidar (light detecting and ranging) sensor. If the distance between the two cars is not bigger than the specified *inter-car distance threshold*,  $\theta_i$  (e.g., 30 meters), it joins the group to which the front car belongs via V2V

communications, similar to [7,8,13]. If there is no car in front or the front car is farther away than the threshold, it becomes the first car of a new group. (Win-Fit does *not* require vehicles to group with each other and travel together when they are not at an intersection.) Also, the lead vehicle in the first group in each lane collects and transmits the following data for every vehicle in the group to the IMA via V2I communications to decrease the communication load, similar to [8,13]:

- Vehicle identification number
- Vehicle length (Unlike [2-5] that assume the constant vehicle length for intersection management, our approach considers actual vehicle lengths.)
- From direction
- To direction, i.e., heading
- Velocity

In addition, the lead vehicle in each lane reports its distance to edge of the intersection, e.g.,  $S1$  in Figure 2, derived using in-car/roadside sensors and the time it has spent waiting to enter the intersection. Note that the vehicle ID, length, and from/to direction are either fixed or determined and remains constant when a vehicle approaches the intersection. Thus, these data have to be successfully sent to the IMA just once. Also, the vehicle speed does not have to be repeatedly reported to the IMA after it becomes equal to the required speed  $v$ , e.g., 15 mph (miles per hour), unless the speed changes later.

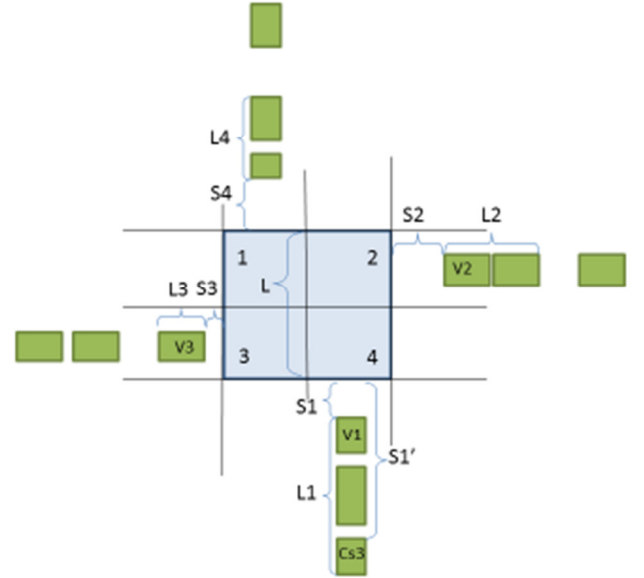


Fig. 2. Vehicle groups at an intersection

It is known that the reliability of wireless vehicular communications is nearly 100% [4], if a vehicle initiates a data transmission that is repeated until a successful transmission (at the frequency of 10Hz in [4]) as soon as its distance to the intersection becomes equal to the specified value (20 meters in [4]). In this paper, the lead vehicle in each approach lane takes a similar approach to transmit the aforementioned data to the IMA, which also broadcasts the intersection trip schedule derived by Win-Fin to the approaching vehicles in a similar manner. If a vehicle is not informed when it can enter the intersection until it reaches the edge of the intersection, it stops

at the edge and requests the IMA to retransmit the information. In this way, we focus on exploring efficient approaches for cooperative autonomous vehicles to cross the intersection with the substantially decreased delay, which is a complementary problem to V2V/V2I communications.

Using the information provided by the vehicles, Win-Fit computes  $L_i$  that is the physical length of group  $G_i$  where  $1 \leq i \leq 4$ . In this paper, as shown Figure 4,  $S_i$  and  $S'_i$  represent the distance from the lead vehicle and last vehicle in  $G_i$  to the edge of the intersection, respectively. Thus,  $L_i = S'_i - S_i + LLV_i$  where  $LLV_i$  is the length of the last vehicle in  $G_i$ . For example, in Figure 4, the length of the group from the south is:  $L_1 = S'_1 - S_1 + \text{len}(Cs_3)$  where  $\text{len}(Cs_3)$  is the length of the last car in the group coming from the south.

### B. Motivation for Win-Fit: Individual Cars vs. Groups

In most existing intersection traffic management methods including [2-6,9,14-16], vehicles' intersection usages are scheduled on the basis of individual vehicles without grouping them and scheduling the groups as batches. We observe that these approaches often introduce additional delays to avoid collisions between streams of cars coming from different directions. For example, consider Figure 1 that shows the initial state of an intersection. In this example, we make simplifying assumptions for the clarity of the presentation: 1) vehicles are only coming from the east and south; 2) there are  $n$  cars in each lane; 3) the distance between two consecutive cars is constant,  $D$ ; 4) the length of each car is  $S$ ; and 5) every car is traveling straight without turning at velocity  $v$ .

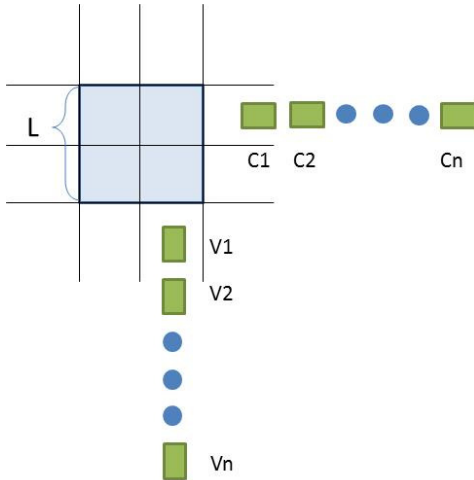


Fig. 3. Individual Cars vs. Groups

In the FCFS (First Come First Serve) protocol widely used to schedule autonomous vehicles for intersection management [2-6], the vehicles in Figure 1 will enter the intersection in the following order:  $(V_1, C_1, V_2, C_2, \dots, V_n, C_n)$ , because they are scheduled based on their distances to the intersection when they move at the same speed  $v$ . By examining this example, we observe that there is always only one vehicle in the intersection at a time until all  $2n$  vehicles leave the intersection to avoid any collision. Although the actual ordering could be different, this observation applies to the other methods that consider individual vehicles separately.

On the other hand, if the vehicles from the south, i.e.,  $(V_1, V_2, \dots, V_n)$ , are grouped together and allowed to pass the intersection as a batch, the average number of vehicles in the intersection at a time is  $\frac{L}{S+D} \geq 1$ , if  $L \geq S + D$ . Thus, more than one car can be in the intersection, increasing the average number of vehicles per unit time. As a result, the intersection throughput is increased. One may argue that this approach can increase the waiting time for the cars in the low priority group. In Win-Fit, this observation is taken into account in addition to dynamic batch scheduling to significantly decrease the average intersection trip delay for every vehicle.

### III. AN OVERVIEW OF WIN-FIT

The high level control flow of our approach is depicted in Figure 4. For intersection scheduling, one group per lane is considered at a time in Win. If there is more than one group in an approach lane, the one that is geographically closest to the intersection is considered. To decrease the waiting time required for safety, Win picks a group that incurs the shortest waiting time for the groups of cars in the other approach lanes as the winner. Fit allows certain lead vehicles of the low priority groups to safely use the cells in the intersection unoccupied by the winner group during otherwise idle time slots (if any) to further decrease the average trip delay as discussed before.

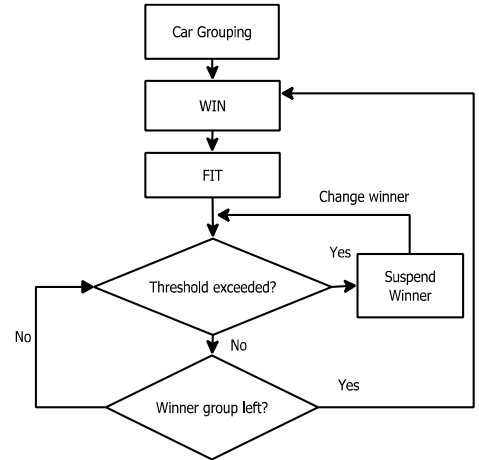


Fig. 4. Control Flow in Win-Fit

More specifically, Win and Fit cooperate to construct four linked lists, called the *time slot lists (TSLs)*, for the north, east, west, and south approach lanes to inform the vehicles when they can safely cross the intersection, while substantially decreasing the average intersection trip delay. The TSLs are recomputed when either the winner group is about to leave the intersection or any lead vehicle of a low priority group has waited for more than the specified waiting time threshold,  $\theta_w$ , such as 30s. In the former, our approach picks a new winner group and updates the TSLs. In the latter, to avoid unbounded starvation, it takes the following steps:

1. The remaining cars of the current winner group are required to stop without entering the intersection.

2. The starved group whose lead car waited more than  $\theta_w$  becomes the new winner as illustrated in Figure 4. If more than one low priority group has waited longer than  $\theta_w$ , the group that has waited for the longest time is selected as the new winner. (A tie is broken randomly.)
3. The wait time is reset to 0 for both the previous and new winner group.

#### IV. WIN: WINNER SELECTION ALGORITHM

The input to Win consists of  $(L_1, \dots, L_4)$  and  $(S_1, \dots, S_4)$  as described in Algorithm 1. In Algorithm 1, if a vehicle can drive through the intersection without stopping or slowing down before any car ever enters a CC, it is detached from its group and allowed to pass the intersection immediately. If there are no more individual winners, Win selects a winner group. It first filters out any group too far away from the intersection. Specifically, if  $S_i > \theta_d$  where  $\theta_d$  is the predefined *car-to-intersection distance threshold*, e.g., 50 meters, the lead vehicle in lane  $i$  is too far away from the intersection and, therefore,  $G_i$  is excluded from the winner selection. If  $G_i$  is not filtered out, Win temporarily hypothesizes that group  $G_i$  is selected as the winner and computes the average delay,  $D_i$ , which should be tolerated by all the cars in the other groups approaching the intersection from the other directions to wait for  $G_i$  to entirely drive through the intersection. Specifically, the time between the first car in  $G_i$  enters the intersection and the last car in the group leaves it is:  $\frac{L+S_i+L_i}{v}$ . For example, in Figure 4, it will take  $\frac{L+S_1+L_1}{v}$  time for  $G_1$  (the group from the south) to drive through the intersection. This is repeated for every approach lane  $i$  with  $S_i \leq \theta_d$ .  $G_i$  is selected as the winner, if it has the smallest  $D_i$  as described in Algorithm 1. If there is a tie between two groups, a winner is picked randomly to break the tie.

**Algorithm 1.** Win-Fit

**Input:**  $(L_1, \dots, L_4)$  and  $(S_1, \dots, S_4)$

**Output:** *Winning group to go across the intersection first*

**while** (the earliest arriving vehicle can pass with no conflict)  
 Select it as the winner and inform it to drive through

```
for ( $i = 1; i \leq 4; i++$ )
```

**if  $S_i \leq \theta_d$ , compute  $D_i$  using Eq. (1) and Eq. (2)**

Select the group  $G_i$  with the smallest  $D_i$  as the winner;

Call Fit();

Broadcast the winner and time slot lists (TSLs)

Each non-winner group needs to advance to and wait at the edge of the intersection until the winner group leaves the intersection. For example, suppose  $G_1$  is the winner. While  $G_1$  is driving across the intersection, group  $G_2$  coming from the east drives forward for the distance of  $S_2$  and waits at the edge until  $G_1$  leaves the intersection. If  $\frac{L+S_1+L_1}{v} > \frac{S_2}{v}$ , the delay

experienced by  $G_2$  waiting for  $G_1$  to leave is  $\frac{L+S_1+L_1}{v} - \frac{S_2}{v}$ . Otherwise, the delay is 0. Thus, if  $G_i$  is selected as the winner, the average delay experienced by the cars in group  $j$  ( $i \neq j$ ) waiting for  $G_i$  to leave is:

$$d_j = \begin{cases} \frac{L+S_i+L_i-S_j}{v}, & \text{if } \frac{L+S_i+L_i}{v} > \frac{S_j}{v} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Thus, the average delay that will be experienced by all the other groups, if  $G_i$  is selected as the winner is:

$$D_i = \frac{\sum_{i \neq j} d_j \times n_j}{\sum_{i \neq j} n_j} \quad (2)$$

where  $n_j$  is the number of the cars in group  $G_j (i \neq j)$ .

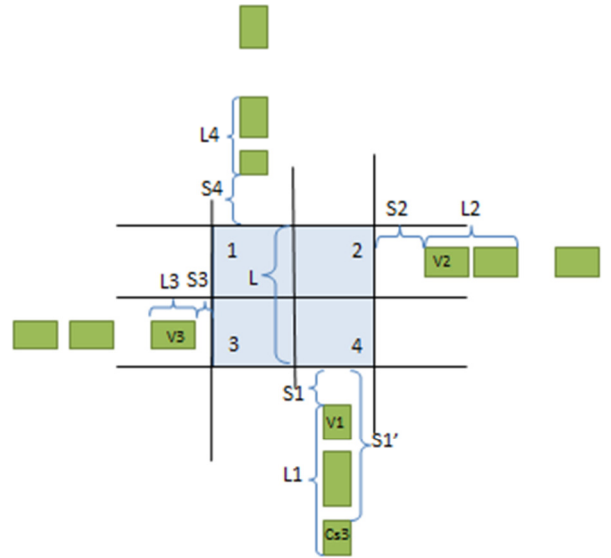


Fig. 5. Vehicle groups at an intersection

In Algorithm 1,  $G_i$  is selected as the winner, if it has the smallest  $D_i$ . The winner group  $G_i$  travels at velocity  $v$  until it completely leaves the intersection. During this process, the individual vehicles in the group may head to different directions (either going straight or turning left or right) in an arbitrary manner.

The worst case time complexity of Algorithm 1 is  $O(n)$  where  $n = n_1 + \dots + n_4$ . Given  $(L_1, \dots, L_4)$  and  $(S_1, \dots, S_4)$ , Win is  $O(1)$ , since it only needs to compute Eq. (1) and Eq. (2) for each direction to select a winner group. The time complexity of Fit is  $O(n)$  that is the lower bound to build the TSLs for  $n$  vehicles that drive through the intersection. Thus, Win-Fit is lightweight and easily deployable in embedded infrastructure nodes at intersections with limited computational resources. A detailed description of Fit and its time complexity analysis follows.

## V. FIT ALGORITHM

We observe that one or more cell may not always be used by the vehicles in the winner group. For example, suppose the group from the south is selected as the winner by Win. If every vehicle from the south turns right, then only one cell, i.e., cell 4, will be occupied as depicted in Figure 6 (a). If no vehicle of this group turns left, only cells 2 and 4 will be occupied as illustrated in Figure 6 (b). Otherwise, three cells are used as shown in Figure 6 (c). Based on the observation, Fit systematically allows the low priority, i.e., non-winner, group cars to use idle cells to increase the average number of the cars in the intersection, while avoiding any collision.

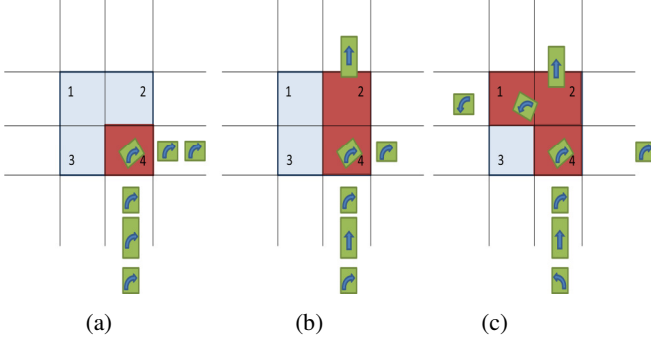


Fig. 6. Different situations of intersection cell occupation

After running the Win algorithm, the IMA builds an initial TSL for the winner group per cell in the intersection to indicate which vehicle can enter which cell at which time with no safety concern. Figure 7 shows an example where the group from the south, which consists of cars 1 – 6, is selected as the. As shown in the TSLs for cells 4, 2, and 1 used by the winner group, vehicles 2 and 6 will take a left turn, vehicles 3 and 5 are going straight, and vehicles 1 and 4 are taking a right turn.

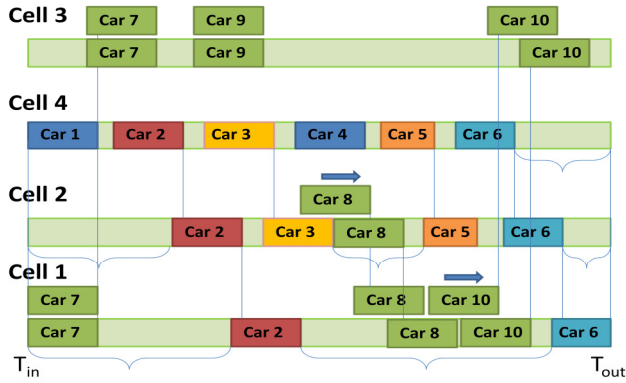


Fig. 7. An example scenario for fit-schedule algorithm

Subsequently, Fit reclaims idle time slots for the vehicles in the low priority groups. From Figure 7, it can be observed that there are relatively big idle time intervals in cells 1 and 2 unused by the six cars from the south. Further, they never occupy cell 3. For example, vehicle 7 can drive straight from the north to south using the idle time slots for cells 1 and 3 in Figure 7. (The projection of each rectangle representing a car

in a low-priority group to the horizontal axis in the list stands for the time slot where a vehicle is entirely or partly inside a specific cell.) Vehicle 8 going straight from the east to west can pass the intersection using two unused time slots for cells 2 and 1 after waiting for a small amount of time as illustrated in Figure 7. Note that the time slot assigned to car 8 to cross the intersection is moved forward to fit into the schedule in Figure 7 to avoid a collision with car 3 in the winner group. Similarly, car 10 going straight from the north to south can travel the intersection after the time slot forwarding in Figure 7. Finally, it is clear that any vehicle that travels from the west to south can use cell 3 with no safety concern.

### Algorithm 2. Fit Algorithm

**Input:** Winner group  $G_i$  (selected by Win in Algorithm 1)  
**Output:** Time slot lists (TSLs)

```

Construct the TSLs for  $G_i$  in lane  $i$ 
 $t_{in}$  = time at which the first car in  $G_i$  enters the intersection
 $t_{out}$  = time at which  $G_i$  will leave the intersection
 $t_{now} = t_{in}$ 
while  $t_{now} < t_{out}$  {
  for ( $j = 1; j \leq 4$  and  $j \neq i; j++$ ) {
    Select the closest car  $C_j$  to the intersection
    if (direction  $j$  is excluded from anymore fit)
      continue;
    else
      Find time slot  $TS_{j,k}$  to be used by  $C_j$  to enter
      cell  $k$  ( $1 \leq k \leq 4$ )
  }
  for ( $j = 1; j \leq 4$  and  $j \neq i; j++$ ) {
    if ( $TS_{j,k}$  of  $C_k$  can't fit into the TSL)
      Move  $TS_{j,k}$  forward along the time axis of
      the TSL
    if (end of  $TS_{j,k} > t_{out}$ )
      Exclude the direction  $j$  from any further
      consideration
  }
}

```

In Algorithm 2, the IMA computes the TSLs for the vehicles in the winner group  $G_i$ , e.g., cars 1 – 6 in Figure 7, as discussed before. The remaining unoccupied time slots (if any) are assigned to the individual vehicles in the other three lanes. In particular, the lead vehicles in the low priority groups, e.g., vehicles 7, 8, and 9 in Figure 7, are considered as candidates. Once the lead vehicle in a low priority lane has been fit into the TSL, the following vehicle in the same lane is deemed as a new candidate. The candidate vehicles that need to use the same cell are ordered in non-descending order of the tentative time slots they will use to enter the cell by driving at speed  $v$ . Subsequently, their tentative time slots are compared to the ones that will be used by the cars in  $G_i$ . If any conflict is found in any cell, the time slot of the vehicle in the low priority group, e.g., car 8 in Figure 7, will be forwarded to avoid a collision. If two or more lead cars of the low priority groups, e.g., cars 8



and 10 in Figure 7, intend to use the same cell, FCFS is applied to order them based on their arrival times to the intersection. (A tie is broken randomly.) If it is impossible to fit the candidate car in  $G_i$  into the TSL by the time the entire winner group leaves the intersection,  $t_{out}$ , it is declared as a failed candidate and the group from the same direction is excluded by Algorithm 2 from any further consideration.

In this paper,  $t_{out} = \text{minimum}(\frac{L+S_i+L_i}{v}, t_{in} + \theta_w)$  where  $\theta_w$  is the maximum threshold waiting time to avoid starvation (defined in Section II). Thus, it is constant. Also, the total number of the time slots that need to be considered by Algorithm 2 is constant:  $c = \frac{t_{out}-t_{in}}{L/2v}$ . Hence, the time complexity of Algorithm 2 is  $O(n)$  for total  $n$  vehicles that need to cross the intersection. As it takes at least  $O(n)$  time to build the TSLs for  $n$  cars, Win-Fit has the lower-bound time complexity.

## VI. PERFORMANCE EVALUATION

### A. Experimental Settings

For performance evaluation, we use an open source traffic simulator, called SUMO (Simulation of Urban Mobility) [17], to implement Win-Fit and several baselines representing the state of the art. Using SUMO, road networks that compose an intersection are generated and different traffic volumes are applied to the simulated intersection for performance evaluation. SUMO uses its own car-following-model to set the acceleration, velocity, and position of a vehicle depending on the distance to the vehicle in front of it and the preceding vehicle's speed.

In this paper, a four-way perfect-cross intersection is simulated and the *average trip delay experienced by every vehicle* to cross the simulated intersection is measured. Each direction has two lanes: one for coming vehicles and the other for leaving cars. The width of the intersection is 8 meters, and the constant velocity  $v$  is set to 15 mph as required for a blind intersection. A simulation run is 15 minutes long. The probability of a vehicle to go straight, turn left, or turn right is 1/3 each. Further, we use the following variables for performance evaluation.

- $\lambda$ : The total vehicle arrival rate at an intersection (vehicles/hour) from all four directions;
- $\theta_i$ : The inter-car distance threshold used to group cars in each lane (defined in Section II);
- $\theta_w$ : The maximum waiting time threshold used to avoid starvation (Section III); and
- $\theta_d$ : The car-to-intersection distance threshold used by Win (Section IV).

In our simulation study, three sets of experiments are performed for increasing  $\lambda$  values. In this paper,  $\theta_i = 30$  meters,  $\theta_d = 50$  meters, and  $\theta_w = 30$  seconds. We have used different values of  $\theta_i$  and  $\theta_w$  that may affect Win-Fit's performance too. For the  $\theta_i$  and  $\theta_w$  values used in this paper, Win-Fit showed intermediate performance. Details are omitted due to space limitations.

In the next subsection, the results for three sets of experiments are presented. In the first two experimental sets, the traffic intensity across the four directions is identical; that is,

each lane has  $\frac{1}{4}$  of the total traffic. In the third set, the performance is evaluated for a major-minor intersection.

### B. Results

#### Exp. Set 1: Trip Delays for the Increasing Traffic Volume.

In this set of experiments, the average trip delay of Win-Fit is compared to the delays of three baselines: traditional traffic lights with the 30 and 60 second long green light as well as the FCFS intersection management scheme. FCFS discussed in Section II is widely applied to individually schedule self-driving cars for autonomous intersection management [2-6,9,14,15]. In Figure 8, the average trip delay is shown for  $\lambda$  increased from 100 cars/hour to 1500 cars/hour by 100 cars/hour at a time. Win-Fit decreases the average trip delay by up to approximately 18s (31%), 24s (38%), and 29s (48.9%) compared to the traffic lights with the 30s green light, traffic lights with the 60s green light, and individual FIFO scheme, respectively. (In this paper, FCFS and FIFO are used interchangeably.) Although FIFO supports the short trip delay for the relatively low traffic volumes, its trip delay increases rapidly for the higher traffic volume in Figure 8. This is because it allows only one vehicle to pass the intersection at a time, increasing the average trip delay as discussed in Section II. Its trip time becomes even longer than those of the traffic lights for the traffic volume of 1000 cars/hour or higher.

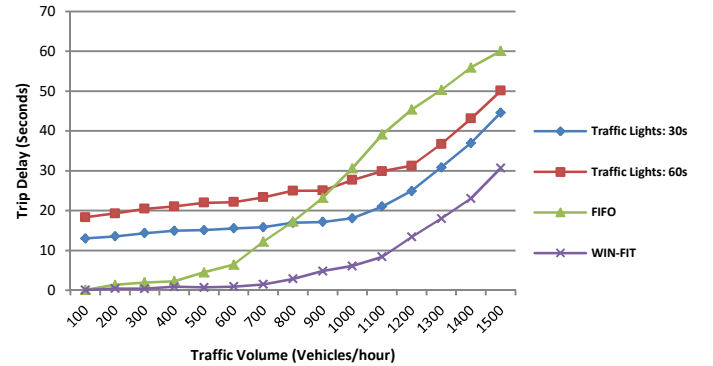


Fig. 8. Trip Delay Comparisons

#### Exp. Set 2: Trip Delays for Various Group-Based Winner Selection Techniques.

In this set of the experiments, we compare the performance of Win-Fit and several other baselines that schedule groups of autonomous vehicles, since Win-Fit outperformed the individual FIFO scheme in the previous set of experiments. Specifically, we use the longest, shortest, biggest#, smallest#, and G-FIFO (Group-based FIFO) baselines that assign the highest priority to the group that has the longest physical length, shortest length, largest number of cars, smallest number of cars, and arrives to the intersection at the earliest time (in terms of the lead vehicle's arrival), respectively. The winner selection strategy of the longest baseline is similar to [15]. In the baselines, the cars in the selected winner group pass the intersection as a batch, similar

to Win-Fit. Essentially, the major differences between Win-Fit and the baselines are the criteria to select a winner group.

As shown in Figure 9, the smallest# and shortest baselines show the worst performance. The longest and G-FIFO baselines in Figure 9 show the closest performance to Win-Fit. Their trip delays are comparable to that of Win-Fit until  $\lambda \leq 1200$  cars/hour. However, their delays increase as  $\lambda$  is increased beyond 1200 cars/hour. For 1500 cars/hour, Win-Fit decreases the trip delay by approximately 40% and 66% compared to the longest and smallest# baselines that show the second shortest delay and the longest delay, respectively. This is because the baselines only consider the physical length, number of the cars, or arrival time of each group rather than directly considering the average trip delay for all vehicles. Due to the relatively superior performance of Win-Fit compared to the other group-based winner selection strategies, we only compare the performance of Win-Fit to those of the traffic lights and individual FIFO scheme next.

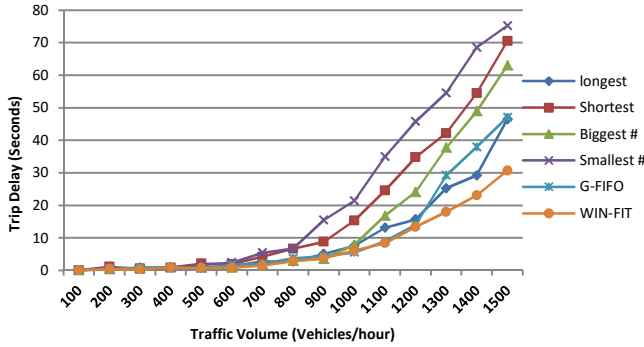


Fig. 9. Trip Delays for Different Group-Based Winner Selection Methods

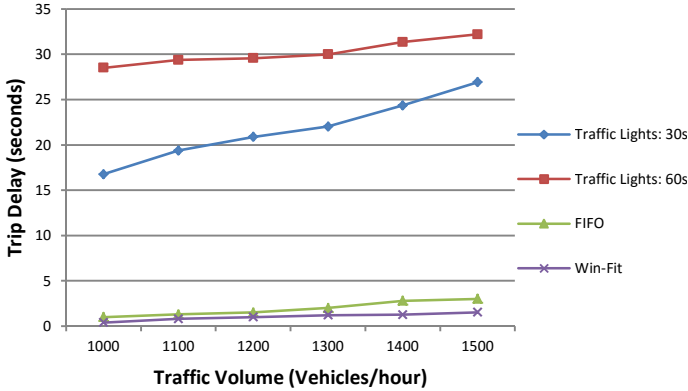


Fig. 10. Trip Delays for the Major-Minor Intersection Setting

**Exp. Set 3: Trip Delays at a Major-Minor Intersection.** In this set of experiments, the traffic volume from the north or south direction, which are minor roads, is fixed at 50 vehicles per hour, while the east and west lanes equally produce all the other traffic flows. As plotted in Figure 10, our protocol decreases the average trip delay by up to approximately 94%,

95%, and 49%, compared to the traffic lights with the 30s green light, 60s green light, and FIFO, respectively. Interestingly, the trip delays of the traffic light methods are several times larger than those of FIFO and Win-Fit, because the vehicles in the major lanes often suffer from the large waiting time caused by the fixed traffic light schedules. On the other hand, FIFO shows the relatively good performance in Figure 10, since cars come from the major lanes at the significantly higher arrival rate and, therefore, a large fraction of them approaches the intersection earlier than the cars from the minor roads do. Overall, Win-Fit achieves the shortest trip delay, since it schedules the group incurring the smallest waiting time to pass the intersection first by directly considering the current traffic status at the intersection. Also, unoccupied cells are reclaimed and reallocated to the vehicles in the low priority groups.

## VII. RELATED WORK

Research on autonomous vehicle is very active with fruitful results. In this section, we review recent work closely related to ours. To improve the intersection throughput without collision, many approaches are developed. In [10, 11, 12], traffic signals are changed adaptively according to the intersection condition to minimize the queue length and fuel consumption. Intersection management for autonomous vehicles has recently been investigated. In [2-5], autonomous intersection management is supported on the basis of individual self-driving cars. In [6], it is shown that an autonomous vehicle can adhere to the FCFS policy via a mixed reality simulation. In [13,14], intersection management is formulated as a mutual exclusion problem and FCFS is extended to lock only a minimal number of lanes required for safety.

In [15], if two vehicles from different directions have any conflict, the car in the lane with more vehicles is assigned a higher priority to cross the intersection (similar to the longest baseline in Section VI). In [7], traffic signal timing is controlled at major-minor intersections to let a group of cars on the major road to pass an intersection faster by reasonably increasing the delay experienced by the vehicles on the minor road. In [12], fairness is considered via adaptive green light sequencing at a signalized intersection. Different from them, Win-Fit is not limited to major-minor intersections. Also, it directly handles potential starvation without requiring traffic lights. In summary, Win-Fit is complementary to these advanced approaches. It could be combined with them to further enhance the intersection throughput and safety. A thorough investigation is reserved for future work.

## VIII. CONCLUSION AND FUTURE WORK

Intersection management is one of most challenging tasks in traffic control. Improving the throughput of the traffic at intersections, avoiding collisions, is desirable in terms of the travel time, safety, fuel consumption, and pollution. However, traditional intersection management methods, e.g., traffic lights and stop signs, are subject to long trip delays and accidents. In this paper, we present a novel intersection management scheme to group autonomous vehicles and schedule them in batches to decrease the average intersection

trip delay and utilize otherwise unoccupied idle cells to increase the average number of cars in the intersection, ensuring the safety and avoiding starvation. In the future, we will investigate more advanced approaches to further decrease the intersection trip delay, while ensuring the safety.

#### ACKNOWLEDGEMENTS

This work was supported, in part, by U.S. National Science Foundation grants CNS-1526932 and CNS-1097511. K. D. Kang is the corresponding author.

#### REFERENCES

- [1] U.S. Department of Transportation, National Highway Traffic Safety Administration, "Crash Factors in Intersection-Related Crashes", 2010. <http://www-nrd.nhtsa.dot.gov/Pubs/811366.pdf>
- [2] R. Azimi, G. Bhatia, R. Rajkumar, P. Mudalige, "Vehicular networks for collision avoidance at intersections", SAE World Congress, 2011.
- [3] R. Azimi, G. Bhatia, R. Rajkumar, P. Mudalige "Intersection management using vehicular networks", SAE World Congress, 2012.
- [4] R. Azimi, G. Bhatia, R. Rajkumar, P. Mudalige "Reliable intersection protocols using vehicular networks", ICCPS, 2013.
- [5] R. Azimi, G. Bhatia, R. Rajkumar, P. Mudalige "STIP: Spatio-Temporal Intersection Protocols for Autonomous Vehicles", ICCPS, 2014.
- [6] T.-C. Au, M. Quinlan, N. Sturca, J. Zhu, P. Stone, "Planning for Improving Throughput in Autonomous Intersection Management", ICAPS, 2010.
- [7] Y. Jiang, S. Li, D. E. Shamo, "A platoon-based traffic signal timing algorithm for major-minor intersection types", Transportation Research Part B: Methodological, 40 (7), 2006, pages 543–562.
- [8] Q. Jin, G. Wu, K. Boriboosomsin, and M. Barth, "Platoon-Based Multi-Agent Intersection Management for Connected Vehicle", IEEE ITSC, 2013.
- [9] L. C. Bento, R. Parafita, and U. Nunes, "Intelligent traffic management at intersections supported by V2V and V2I communications", IEEE ITSC, 2012.
- [10] C. Priemer, B. Friedrich, "A Decentralized Adaptive Traffic Signal Control Using V2I Communication Data", International IEEE ITSC, 2009.
- [11] H. Rakha, and R. K. Kamalanathsharma, "Eco-driving at signalized intersections using V2I communication", IEEE ITSC, 2011.
- [12] B. Zhou, J. Cao, X. Zeng, and H. Wu, "Adaptive Traffic Light Control in Wireless Sensor Network-Based Intelligent Transportation System", IEEE VTC, 2010.
- [13] W. Ni and W. Wu, "A Message Efficient Intersection Control Algorithm Based on VANETs", International Conference on Internet of Vehicles – Technologies and Services, 2014.
- [14] W. G. Wu, J. B. Zhang, A. X. Luo, J. N. Cao, "Distributed Mutual Exclusion Algorithms for Intersection Traffic Control", IEEE TPDS, 26(1), pp. 65 - 74, 2014.
- [15] J. Lee and B. Park, "Development and Evaluation of a Cooperative Vehicle Intersection Control Algorithm Under the Connected Vehicles Environment", IEEE ITS Transactions, 13(1), pp. 81- 89, 2012.
- [16] V. Milanés, J. Pérez, Enrique Onieva, and C. González, "Controller for Urban Intersections Based on Wireless Communications and Fuzzy Logic", IEEE ITS Transactions, 11(1), pp. 243-248, Mar. 2010.
- [17] Simulation of Urban Mobility, <http://sourceforge.net/projects/sumo/>
- [18] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States", Proceedings of the IEEE, vol. 99, issue 7, page 1162 – 1182, 2011.
- [19] W. Xiang, Y. Huang, S. Majhi, "The Design of a Wireless Access for Vehicular Environment (WAVE) Prototype for Intelligent Transportation System (ITS) and Vehicular Infrastructure Integration (VII)", IEEE VTC, 2008.