Batch Reservations in Autonomous Intersection Management

(Extended Abstract)

Neda Shahidi
Dept. of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712, U.S.A.
neda@mail.utexas.edu

Tsz-Chiu Au
Dept. of Computer Science
University of Texas at Austin
Austin, Texas 78712, U.S.A.
chiu@cs.utexas.edu

Peter Stone
Dept. of Computer Science
University of Texas at Austin
Austin, Texas 78712, U.S.A.
pstone@cs.utexas.edu

ABSTRACT

The recent robot car competitions and demonstrations have convincingly shown that fully autonomous vehicles are feasible with current or near-future intelligent vehicle technology. Looking ahead to the time when such autonomous cars will be common, Dresner and Stone proposed a new intersection control protocol called *Autonomous Intersection Management* (AIM) and showed that by leveraging the capacities of autonomous vehicles we can devise a reservation-based intersection control protocol that is much more efficient than traffic signals and stop signs. Their proposed protocol, however, handles reservation requests one at a time and does not prioritize reservations according to their relative importance and vehicles' waiting times, causing potentially large inequalities in granting reservations. For example, at an intersection between a main street and an alley, vehicles from the alley can take a very long time to get reservations to enter the intersection. In this research, we introduce a prioritization scheme to prevent uneven reservation assignments in unbalanced traffic. Our experimental results show that our prioritizing scheme outperforms previous intersection control protocols in unbalanced traffic.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—Multiagent systems

General Terms

Algorithms, Performance, Economics, Experimentation, Theory

Keywords

Autonomous vehicles, multiagent systems, coordination

1. INTRODUCTION

The impressive results of the DARPA Urban Challenge in 2007 showed that fully autonomous vehicles are technologically feasible with contemporary hardware. Dresner and Stone proposed a reservation-based approach to autonomous intersection management, and in particular described a *First Come, First Served* (FCFS) policy for an intersection management agent to direct vehicles through an intersection [1, 2]. They showed that FCFS can significantly improve the throughput of an intersection over traffic signals and stop signs. FCFS, however, handles reservation requests one at a time and does not prioritize reservations according to their relative importance and vehicles' waiting times. In many multiagent systems, a poor allocation of resources can lead to starvation—some agents cannot get the resources they need for a very long time or indefinitely. The same is true in AIM: in *unbalanced* traffic—the traffic on a main road is much heavier than the traffic on a crossing road—vehicles from the crossing road can be blocked by the traffic on the main road with heavy traffic, as shown in Figure 1. Unbalanced traffic is very common as many intersections in cities are junctions connecting alleys or side roads to main streets. Therefore, it is necessary to find an autonomous intersection control mechanism that can smoothly and fairly handle this type of traffic. In this paper, we introduce a new intersection control policy called the *batch policy* that can group several reservation requests together and apply prioritization schemes to reorder the requests. The prioritization schemes can enforce that a vehicle from the low traffic road will be given a high priority for reservations if its movement has been blocked for too long.

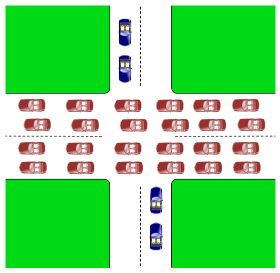

Figure 1: Starvation due to unbalanced traffic. Vehicles from the side road (the vertical direction) cannot get reservations to enter the intersection due to the heavy traffic on the main street (the horizontal direction).

2. BATCH PROCESSING OF REQUESTS

We propose a new class of intersection management policies called *batch policies* that put the request messages on hold upon

Cite as: Batch Reservations in Autonomous Intersection Management (Extended Abstract), Neda Shahidi, Tsz-Chiu Au and Peter Stone, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems – Innovative Applications Track (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. XXX-XXX.
Copyright © 2011, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

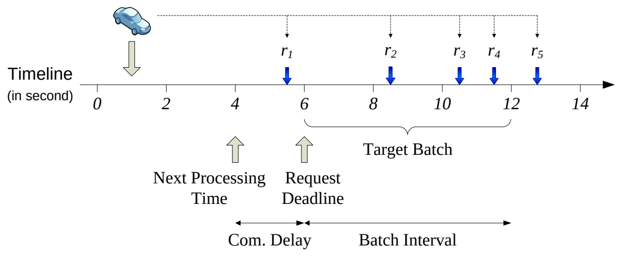

Figure 2: A batch policy

receiving them and then process several request messages at once. The central component of a batch policy is a sorted queue of request messages that acts as a buffer for temporarily storing the incoming request messages. As an example, suppose a vehicle sends a request message r at time 1 second, as shown in Figure 2. The request message contains 5 proposals, each of which is a tuple $r_i = (t_{\mathsf{arrival}}, v_{arrival}, l_{arrival}, l_{exit})$, where t_{arrival} is the arrival time, $v_{arrival}$ is the arrival velocity, $l_{arrival}$ is the arrival lane from which the vehicle arrives at the intersection, and l_{exit} is the exit lane from which the vehicle leaves the intersection. The intersection manager can choose at most one of the proposals to grant a reservation. These proposals, except r_1, will be put in the queue, which is sorted by the proposed arrival times, and they will be processed by the intersection manager at a future time called the *next processing time*, which is denoted by nextProcessingTime. r_1 is not put in the queue because its proposed arrival time is before the *request deadline*, denoted by requestDeadline, which is a time that is very close to the next processing time. r_1 is considered late because by the time the intersection manager finishes processing the request messages at the next processing time, it is possible that the arrival time of r_1 has been passed. The *computation and communication delay* (the com. delay in Figure 2) is the time delay between nextProcessingTime and requestDeadline and is denoted by t_{comm}. Late proposals such as r_1 are processed immediately by the intersection manager, to see if it is possible to grant the reservation between the reservations that were granted at the last processing time. If not, a reject message is sent.

The request handling procedure processes request messages on the queue at the next processing time. The procedure first identifies the *target batch* of request messages on the queue, which is the set of all request messages whose proposed arrival times are before requestDeadline $+ t_{\mathsf{batch}}$, where t_{batch} is the *batch interval* which is 6 seconds in this example. The request messages in the target batch will be removed from the queue and reordered by a *cost function*, which is $f(wait) = a \times (wait)^b$, where *wait* is the estimated amount of time the vehicle has been waiting to enter the intersection. a and b are coefficients specific to the type of vehicles, where $a > 0$ and $b > 1$. The procedure grants reservations according to the new order and then rejects the requests from the vehicles that have no reservation and no remaining request messages on the queue. Finally, both nextProcessingTime and requestDeadline are increased by time t_{proc}, which is called the *processing interval* and is the time between the batch processing of requests.

We conducted an experiment on an intersection between a main road and a side road. Each of the roads has three lanes, and the vehicles on the main road go straight through the intersection without turning while the vehicles at the side road can either turn left, turn right or pass through the intersection. The vehicles are spawned according to a poisson distribution such that the traffic level λ_{main} of the main road is varied from 72 vehicles per hour per lane to 2200 vehicles/hour/lane while the traffic level λ_{side} of the side road

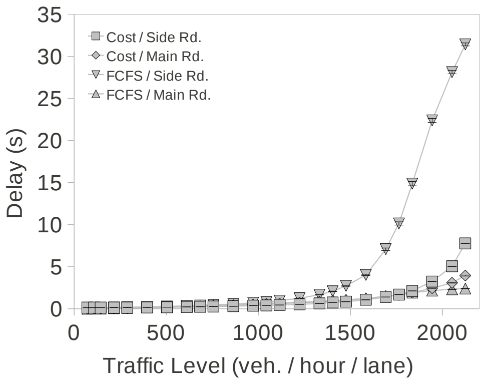

Figure 3: Average delays of the vehicles versus traffic levels of the main road (λ_{main}). The delays of the vehicles on the main road and the side road are shown separately.

is held constant at 540 vehicles/hour/lane. We ran the simulation 100 times, and in each run the total simulation time is 1 hour. The coefficients of the cost function are set to $a = 1.0$ and $b = 2.0$, the batch interval is $t_{\mathsf{batch}} = 3s$, the processing interval is $t_{\mathsf{proc}} = 0.5s$, and the com. delay is $t_{\mathsf{com}} = 0.02s$. We measured the average delay of the vehicles by averaging the time difference of the vehicles with and without other vehicles on the roads (i.e., the time delay due to the presence of traffic and the intersection management policy), and plotted the graph in Figure 3. As can be seen, the delay of vehicles on the main road in FCFS is small (within 3 seconds) at all traffic levels λ_{main}, while the delay of vehicles on the side road increases rapidly as λ_{main} increases. The vehicles on the side road have difficulty getting reservations due to the situation as shown in Figure 1, and this difficulty can be avoided by using the batch processing of requests, which helps to avoid the long delay of the vehicles on the side street. As a result, the delay of the vehicles on the side road is reduced tremendously, at the cost of a very small increase of the delays on the main street (see Figure 3).

3. CONCLUSIONS

As in many multiagent systems, there is a need for a fair allocation of resources in an intersection to ensure that all vehicles can get a reservation to enter the intersection eventually. Here we introduced a prioritization scheme and discussed how to incorporate them into the AIM system via the batch processing of reservation requests. Our experimental results show that our prioritization scheme outperforms FCFS, the best autonomous intersection control protocol in the literature, in unbalanced traffic. We believe this work will serve as an important step towards the development of traffic control systems for autonomous vehicles.

4. ACKNOWLEDGMENTS

This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (IIS-0917122), ONR (N00014-09-1-0658), and the FHWA (DTFH61-07-H-00030).

5. REFERENCES

[1] K. Dresner. *Autonomous Intersection Management*. PhD thesis, The University of Texas at Austin, 2009.

[2] K. Dresner and P. Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research (JAIR)*, March 2008.