

VGrid: Vehicular AdHoc Networking and Computing Grid for Intelligent Traffic Control

Jason LeBrun, Joey Anda, Chen-Nee Chuah, Michael Zhang, Dipak Ghosal

ABSTRACT

In this paper, we propose VGrid: an ad hoc networking and computational grid which can be formed by leveraging inter-vehicle and vehicle-to-roadside wireless communications. Rather than just using ad-hoc networks for exchanging data between vehicles, VGrid actively uses pertinent data to perform computations for solving traffic-related problems. The goal is to evolve intelligent transportation engineering from a centralized approach to a distributed approach, in which vehicle computers can cooperate and solve vehicular traffic-flow control problems autonomously. In our work, we present an example application: the merging of two lanes into one. We explore various algorithms to compute the optimal schedule for vehicle arrivals at the merge point using velocity/position information exchanged between vehicles. Our simulation results, using realistic vehicle mobility patterns, show that the proposed VGrid framework and algorithms can provide an increase in throughput as well as a decrease in latency through the merge point.

1. INTRODUCTION

The FCC has recently allocated the 5.85-5.925GHz portion of the spectrum for inter-vehicle communications (IVC) and vehicle-to-roadside communications (VRC), known as Dedicated Short Range Communications (DSRC). This has fueled significant interest in designing new applications, including driver-vehicle safety applications, infotainment applications, and mobile internet services for passengers [1,3]. However, there is a huge untapped opportunity to leverage vehicular ad hoc networks (VANET) to revolutionize the intelligent transportation systems (ITS). Unlike other mobile ad hoc networks (MANETs) that consist of power- and computing limited nodes, such as wireless sensor motes or hand-held devices, VANET has notably different design characteristics. The vehicles have ample power/energy and can be equipped with computing resources (e.g., processor and storage space) resources. On the other hand, their high mobility results in very dynamic channel conditions.

In this work, we propose a new paradigm called VGrid (Vehicular-based Networking and Computing grid), where we leverage DSRC-enabled vehicles to perform *data sensing, relaying, and computing* to support *distributed monitoring and control of vehicular traffic flow*. Many kinds of road-side infrastructures (e.g., fixed sensors) for monitoring highway conditions have been in place. They are used to support obstacle detection/avoidance, speed monitoring, or meter-light control. For example, the existing ITS collect traffic statistics from roadside sensors and send it back to a central location, where computers and/or humans analyze the data to decide the optimal traffic-light schedules or to plan construction and detour routes. A well-designed, distributed traffic management architecture can drastically reduce this “feedback loop”, which occurs over a time period on the order of weeks or months, to something on the order of seconds or minutes.

In VGrid, vehicles play the role of both mobile sensors (collecting data) and mobile routers (relaying data), and are linked together to form a global grid computer. A high density of cars results in a higher density of potential nodes that can be temporarily organized on the fly to perform a distributed computation to solve a single problem. This networking/computing capability can enable vehicle-driver safety applications. For example, to ease the merging of traffic from two lanes to one, the affected vehicles can exchange speed/direction/position information and schedule the best arrival time at complex highway interchanges. The mobile computer grid can also monitor and control ramp metering. Other applications may include: 1) Analyzing traffic congestion on the fly; 2) Computing optimal detour routes for vehicles, based on their destinations; 3) Collaborative tracking of vehicles; 4) Providing a virtual front view under poor weather conditions; and 5) Locating accidents and alleviating congestion through traffic metering or early warning messages

2. THE VGRID FRAMEWORK

We envision an architecture in which the results of computations of various nodes are shared with neighbors, thereby influencing those neighbors’ computations.

We must consider that the applications which utilize the grid computing system will operate in a number of different scopes: ranging from a single car to a platoon of cars that form a peer space, as discussed by Chisalita and Shahmehri [2]. The *platoon scope* is important since many potential VGrid applications will require knowledge of data contained in a local area of interest. However, for some applications, we may need a larger scope that combines a collection of peer spaces to achieve wider-scale results. Finally, we may think in terms of wide-area networking, in cases where data is sent back to a central server, or Internet services are utilized.

First, we need to consider, at the high level, the types of applications that might be implemented in this system. They fall into the following categories: 1) Local grid-style computations (platoon scope); 2) Large-scale distributed problems (inter-platoon scope); 3) Grid-style computations on behalf of another organization (SETI@Home, on the road) (inter-platoon scope); 4) Inter-personal communications (platoon scope); and 5) Providing Internet access to vehicles (wide-area scope). In this paper, we focus our discussion on the first two classes of applications, which will most likely evolve into interesting and completely-distributed autonomous control systems.

3. Application: Two Lane Merge

We consider an application scenario where two lanes merge into a single lane. Given certain traffic intensities in each lane, the goal is to determine the speed of each car such that average latency is minimized and the average throughput is maximized. Throughput is defined as the number of cars to successfully complete the merge in a given amount of time. Latency is defined as the time it

takes for a vehicle to get past the merge point once it has entered the system. Unlike traditional scenarios, each vehicle knows the location and the velocity of all the other vehicles using VGrid. We have considered the following lane merging algorithms and compared them using our simulation tool.

1. **Weighted Fair Queuing (WFQ):** This algorithm is based on the WFQ algorithms proposed for queue management in IP routers [5,6]. In this algorithm, each vehicle uses the information received from the surrounding vehicles to determine whether their current pace will result in a collision. If it is determined that a collision will occur, a dynamic weight is computed for each lane dependent upon their current congestion levels and priority is granted to the vehicle in the more congested lane.

2. **Platoon Scheduling (PS):** In this algorithm, the VGrid platform is used to coordinate the platooning of vehicles in one lane (referred to as the platoon lane) and the scheduling of vehicles in the other lane to fit between the groups at the merge point. When a vehicle enters the system in the platoon lane, it attempts to find vehicles in proximity to which it can form a platoon. Vehicles in the other lane proactively adjust their speed in order to fit between platoons and successfully merge.

3. **Take Turns:** This is a simple form of meter light scheduling. The vehicles merging are scheduled by following a strict alternation between the lanes. We consider this is a base case for comparison.

4. **No Restriction with Collisions (NRC):** In this algorithm there are no restrictions on the incoming vehicles. The vehicles continue through the merge at the maximum possible speed without considering the possibility of collisions. This algorithm is the best in terms of delay for vehicles that do not collide, since they will be allowed to drive through the merge without any scheduling delays.

5. **No Restriction without Collisions (NRI):** The final algorithm is an ideal form of a no-restrictions algorithm (NRI), where vehicles are allowed to continue at their current rate into the merge point, with speed modifications occurring only when necessary to avoid collisions. The speed modifications are made such that the vehicle must only decelerate as much as is necessary so a collision is prevented.

We have developed a Java-based simulation tool to help understand the potential for grid computing in a vehicular environment, where the grid nodes are automobiles that can use wireless communication to share information and solve traffic flow problems. The simulation tool is based upon the publicly available *Cellular Automaton Traffic Simulators* applet developed by Kai Bolay [4]. Throughput results for the various algorithms can be seen in Figure 1, with the VGrid approaches showing a major improvement over simple meter light scheduling techniques (TakeTurns) and non-scheduling techniques (NRC).

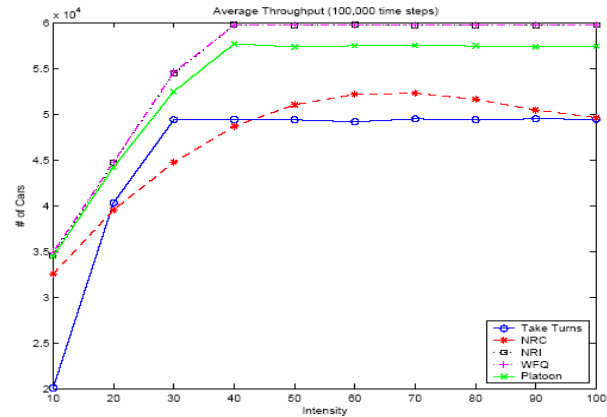


Figure 1: The throughput for varying intensity levels of Lane 2, with Lane 1 maintaining an incoming intensity level of 25%. The curves for NRI and WFQ are coincident.

4. CONCLUSION

While there has been a lot of research into ways to allow computational devices on vehicles to communicate with each other, there is often a necessary interaction with some sort of infrastructure in order to successfully execute traffic safety applications. In our work, we outline a framework for networking and computing grids to allow fully distributed traffic control via vehicular ad-hoc networks. We also outline a case study based on the problem of scheduling vehicles that are trying to merge from two lanes into one. Simulating various methods for scheduling these cars gives preliminary indications that using scheduling algorithms rather than driver insight can improve the throughput and latency of traffic merging scenarios.

5. REFERENCES

- [1] Marc Bechler, Walter J. Franz, and L. Wolf. Mobile Internet Access in FleetNet. <http://www.fleetnet.de/>.
- [2] Ioan Chisalita and Nahid Shahmehri. A peer-to-peer approach to vehicular communication for the support of traffic safety applications. In *ITSC*. IEEE, 2002.
- [3] Drive-thru Internet: IEEE 802.11b for Automobile Users. <http://www.drive-thru-internet.org/index.html>.
- [4] Kai Bolay. Cellular Automaton Traffic Simulators based on the work of Nagel-Schreckenberg (1992), Takayasu (1993), Helbing and Schreckenberg (1999). <http://rcswww.urz.tu-dresden.de/~helbing/RoadApplet/>.
- [5] Alan Demers, Srinivasan Shenker, and Scott Keshav. Analysis and simulation of a fair queueing algorithm. In *Internetworking: Research and Experience*, pages 3–26, September 1990.
- [6] Hui Zhang and Jon C.R. Bennett. Wf2q: Worst-case fair weighted fair queueing. In *INFOCOM*, pages 120–128. IEEE, 1996.