차량과 노변장치의 통신에서 처리율 향상을 위한 전송 스케쥴링 방안

임연섭, 황재룡, 김종권
서울대학교 전기컴퓨터공학부
{ylim, jrhwang, ckim}@popeye.snu.ac.kr

RSU scheduling method to improve service ratio in vehicle-to-RSU communication

Yeon-sup Lim, Jaeryong Hwang, Chong-kwon Kim
School of Electrical Engineering and Computer Science, Seoul National University

Abstract

Recently, people have been interested in new intelligent transport system (ITS) architecture, so that vehicular networks are becoming an attractive research area. When vehicles try to access data through a roadside unit (RSU), data scheduling of RSU needs to improve service ratio due to limited bandwidth and service time. In this paper, we propose a service scheduling algorithm based on transmission delays and service deadlines of vehicles in order to improve the service ratio. In our algorithm, we assume the promiscuous operation of wireless nodes and it can make a single transmission of RSU serve multiple requests. We evaluated the performance of our scheme via simulations, and results show that our schemes have better performance than existing algorithms

1. Introduction

Advances in wireless telecommunication and internetworking have enabled the ubiquitous computing connected to the networks. Recently, this ubiquitous computing is also needed in the vehicular area. In these vehicular networks, there is the proposed structure using roadside units like IEEE 802.11 access point. In this structure, vehicles can access data stored in the RSU such as traffic and environmental information, and the RSU can act as a router in traditional networks or just buffer point. In our approach, we focus on the buffer point operation due to its low cost and easy deployment [5].

In this paper, we propose an efficient scheduling method (SRM) for data access in vehicle-to-RSU communication. We use information about transmission delays and deadlines of vehicles to determine which service will be served. The goal of this work is to achieve larger service ratio than traditional queue scheduling methods such as Drop-Tail, RED, and etc.

This paper is organized as follow. We describe system model and assumptions in Section 2. Section 3 presents the algorithm to schedule requested services. Simulation results are given in Section 4. In Section 5, we discuss related issues. Section 6 concludes the paper.

2. Preliminaries

2.1 Environment and Assumptions

We assume situations that a large number of vehicles retrieve data from a RSU when they are in the communication range of the RSU. As shown in Figure 1, each vehicle in the communication range of the RSU sends requests, and the RSU puts the requested data in the service queue if it is providing them. A vehicle can know whether it is in the area of the RSU or not via beacons which the RSU sends periodically. These beacons contain some information which vehicles should know to request data. Each request is consist of 3-tuples: <id, request type, deadline>, where id is the identifier of the vehicle, request type is the type of requested data, and deadline is the timing constraint to receive that the requested data has to be served. In the service queue of the RSU, if specific data are being transmitted, no other data packet can preempt them. For brief simulations and analysis, we assume that there are 25 types of services and that no more packets of same service cannot be put in the queue since the requested service are already in the queue if one service is in the queue. In a real situation, since there are many kinds of services this assumption should be eliminated. However, an approach of the algorithm will be similar.

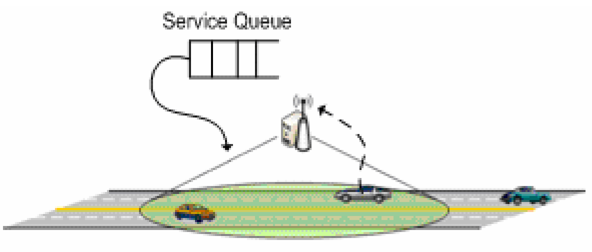

Figure 1. System model of Vehicle-RSU Service Scheduling

In addition, each vehicle can know its service deadline from information of beacons. Since beacons provide information about the service set and transmission time of services, a vehicle can calculate its deadline based on this information.

2.2 Performance issues

Different from traditional scheduling, the scheduling of the RSU has an important feature that each data should be served by the RSU during a short period until vehicles move out of the RSU area. Thus, we should import new metric in order to evaluate scheduling algorithms due to this difference. Yang Zhang et al. introduced service-ratio (which is defined as the ratio of the number of requests served before the service deadline to the total number of arriving requests) as the metric to evaluate algorithms in [5]. We have evaluated our approach based on this metric, too.

3. Service Ratio Improvement of Scheduling

The primary goal of a scheduling scheme is to serve as many requests as possible. We identify parameters that can be used for scheduling:

- Transmission Delay (e_i): When a service of a request is started, it is the spend time until this service finish. If this delay is longer, it disturbs other service for a long time.
- Deadline set (D_i): If a request can not be served before its deadline, it has to be dropped. Thus, the request with an earlier deadline is more urgent than the request with a later deadline. In our algorithm, the deadline of each request is recorded for specific service.
- Requested number (N_i): The number of vehicles that ask for the service i.

These parameters can be expressed as follow:

$$TransmissionDelay = \frac{DataSize}{DataRate}$$

$$Deadline = CurrentTime + timeToMoveOutArea - \frac{DataSize}{DataRate}$$

Also, r_i, the response time of ith service in the queue, can be described with e_j. That is

$$r_i = \sum_{j=1}^{i} e_j$$

where e_j is the transmission delay of jth service.

Thus, if r_i exceeds all deadline of its set, service i need not to be scheduled in the queue. For example, service 0 has deadline set {3, 5, 10} and response time 11, then service 0 cannot be served to any one. With at least one deadline more latter than response time, the service can be served to some one. Accordingly, the goal of the problem is to achieve a sequence maximizing the sum of the count of j which meets $r_i < d_i^j$ for all j where d_i^j denotes jth element of deadline set of service i. We consider this problem as 0-1 knapsack problem with an adjustable maximum weight of knapsack. The 0-1 knapsack problem has two parameters such as weight and value. In our problem, weight and value are same with the transmission delay and the requested number, respectively. Now, we should determine the maximum weight of knapsack. We assume that the request sequence is already sorted by deadline (earliest one of set) increasing order before scheduling. Thus, if we select one service indexed by 1 and it meets its deadline, the maximum weight of knapsack is the deadline of itself and the reward (it means the sum of values) is the requested number of itself. When we select next one indexed by 2, the maximum weight of knapsack is changed to maximum deadline among service 1 and 2. The count of j which meet $r_2 < d_i^j$ is added to the reward. At each step, services which cause the reward decrease are discarded. Following table 1 show the example sequence to calculate the reward and schedule the service. In step 3, service 1 is dropped since the reward including it is less than other sequences.

Table 1. Example Sequence

Step	The sum of weight	The Max. weight of knapsack	Pass Condition	Reward
1	e_1	Max d_1^j	$d_1^j > e_1$	Count of j which passes condition
2	e_1+e_2	Max d_2^j	$d_1^j > e_1$ $d_2^j > e_1 + e_2$	
3	e_2+e_3	Max d_3^j	$d_2^j > e_2$ $d_3^j > e_2 + e_3$	

We have used the iterative solution of 0-1 knapsack to find the maximum of values' sum. When the scheduler includes a service in the queue, scheduler finds the sequence include it or not in order to maximize the reward. If it cannot make any increase of the reward, it will be discarded. (This means that the sequence which includes it based on the deadline priority has less reward than one which excludes it.)

4. Performance Evaluation

We have evaluated the performance by using our simulation program based on C++. The simulation is based on a street scenario. The RSU is at the center of the street and vehicles move towards end of the road and back to the start point during the simulation time. Each vehicle issues requests with a probability p and each request has 0~20 seconds deadline.

A larger p is used to simulate a heavier workload under the same number of vehicles. Similar to [1], the access pattern of each service follows *Zipf* distribution. In the *Zipf* distribution, the access probability of the ith item is represented as following:

$$P_i = \frac{1}{i^{\theta} \sum_{j=1}^{n} \frac{1}{j^{\theta}}}$$

where n denotes the total number of data.

When θ = 0, it becomes the uniform distribution. When θ increases, lower ith data will be requested more. Each vehicle which receives beacon selects required service and sends request message. Figure 2 show CDF of requested probability on various θ.

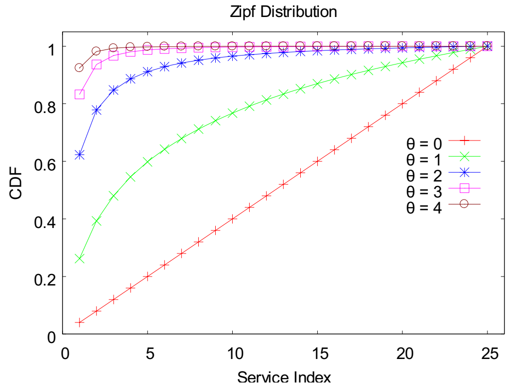

Figure 2. Zipf Distribution

Table 2. Simulation Parameter

Parameter	Value
Simulation time	500 s
Number of vehicles	Various
Transmission Rate	6 Mbps
Data size	75Kbyte ~ 2.25Mbyte
No. of service app	25
Zipf Parameter θ	0.7

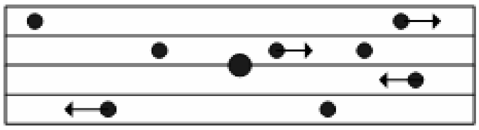

Figure 3. Simulation Layout

We have compared our scheme to First Come First Service (FCFS) and Earliest Deadline First (EDF)-scheduling. FCFS is the strategy that the request with the earliest arrival will be served first and EDF is the one that the request with the earliest deadline will be served first. We assume that FCFS and EDF operate under same assumptions as them of SRM. Without these conditions, FCFS and EDF experience heavy performance degradation when the number of vehicles increases. This is because that FCFS and EDF transmit more packets basically without these assumptions.

Figure 4 and 5 show the effect of the workload (the more number of vehicles, the more service request). As shown in Figure 4, the more service cannot be served when the number of vehicles increases. Under low workload (when the number of vehicles is under 50), the performance of SRM is similar to EDF. However, while the service ratio of other schemes decreases very quickly with the increasing of workload, SRM maintains service ratio larger than others. Figure

5 is the result under heavier workload (request probability is 0.7). As we expected, under the situation of heavier workload, SRM outperform others, too.

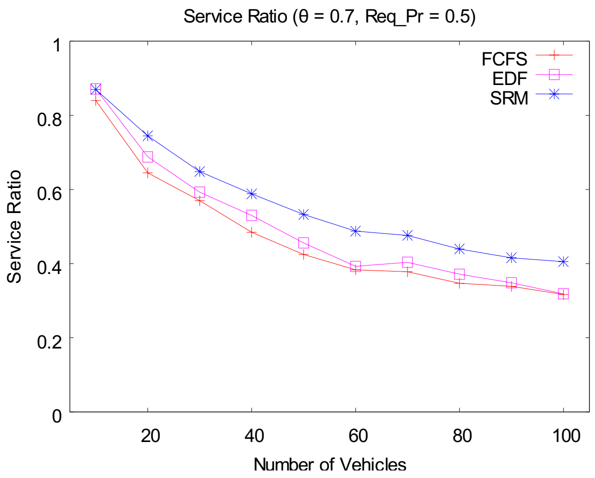

Figure 4. Service Ratio (Req_Pr = 0.5)

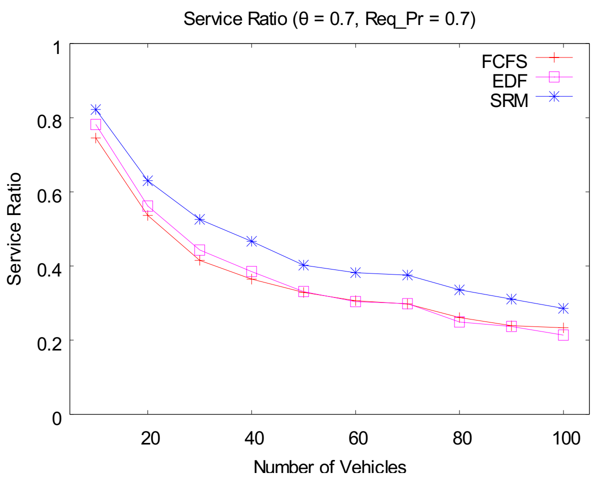

Figure 5. Service Ratio (Req_Pr = 0.7)

5. Related Works

Over the past years, there are a large number of works related to CPU and job scheduling such as first-come-first-serve (FCFS), earliest deadline first (EDF), and so on. EDF scheduling is a dynamic scheduling principle used in real-time operating systems. Whenever a scheduling event occurs, the process which has earliest deadline will be found. This process will then be scheduled for execution next. Wong studied several scheduling algorithm in the broadcasting environments [2]. Later, many broadcast scheduling algorithms have been proposed [3, 4]. In [5], Yang Zhang et al. introduced DSN service scheduling in vehicular networks. Their approach is modified EDF approach in order to meet time constraint of the request. Our scheduling scheme is motivated with this approach.

6. Conclusion

In this paper, we present the scheduling algorithm to improve a service ratio. We could find that a selection of better service sequence can make a great performance improvement. Through simulation results, we identified the effect of workloads to service-ratio and SRM can maintain reasonable operation under heavy workload.

We have evaluated the performance via our own simulator in the condition which we want to observe. Thus, to generalize a performance of our scheme, we are implementing our scheme on NS-2 simulator. This simulation method is expected to show results which are more similar to results of real-world environment. We are also implementing more other schemes to compare with SRM such as DSN scheduling. We expect simulation results based on NS-2 show our schemes outperform these other schemes.

In addition, the computational overhead of our scheme is greater than others'. Therefore, we are studying the technique to reduce the complexity for making scheduling decisions in SRM.

7. Acknowledgement

This work was supported by the Korea Science and Engineering Foundation(KOSEF) grant funded by the Korea government(MOST) (No. R01-2007-000-20154-0)

8. References

[1] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks", *IEEE Transactions on Mobile Computing*, Vol.5, no. 1, January 2006.

[2] J. Wong, "Broadcast delivery", In *Proceeding of the IEEE*, pages 1566-1577, 1988.

[3] C. Su and L. Tassiulas, "Broadcast scheduling for information distribution", In *Infocom' 97*, 1997.

[4] N. Vaidya and S. Hameed. "Scheduling data broadcast in asymmetric communication environments", *Wireless Networks*, Vol. 5, 1999.

[5] Yang Zhang, Zing Zhao and Guohong Cao, "On Scheduling Vehicle-Roadside Data Access", *VANET'07*, September, 2007