A Message Efficient Intersection Control Algorithm Based on VANETs

Wei Ni and Weigang Wu

Department of Computer Science, Sun Yat-sen University,
Guangzhou 510006, China
wuweig@mail.sysu.edu.cn

Abstract. Intelligent traffic management via V2V communications in VANETs is attracting more and more attentions from researchers. In this paper, we design a new algorithm to realize intersection control based on coordination among vehicles via VANETs. We basically adopt the concept of mutual exclusion originally proposed for resource management in computer systems. Vehicles at an intersection compete for the privilege of passing by message exchange. The core of such an approach is the algorithms to coordinate vehicles and control the privilege granting. Following our previously proposed intersection control algorithm in [16], we design a new algorithm that can realize intersection control with much less communication cost. The advantage of our new algorithm is validated by simulations using ns-3.

Keywords: VENETs, distributed mutual excision, traffic control, message cost.

1 Introduction

Traffic control at intersections has been always a key issue in the research and development of ITS [1]. Traditional intelligent intersection control focuses on how to optimize the scheduling of green signal [2][3][4]. Recent efforts on traffic light control focus on adaptive and smart traffic light scheduling, mainly by making use of computational intelligence approaches [5]. Unfortunately, due to the dynamics of traffic load, traffic control systems are large complex nonlinear stochastic systems, so determining the optimal time of green light is very hard even if not impossible [5][6]. Moreover, the complexity of computational intelligence algorithms makes them usually not applicable to real-time traffic light control.

Intelligent intersection control has also be realized via advanced sensing and communication technologies [7] have enabled real-time traffic-response green light control [8][9]. The traffic light is scheduled under a certain control strategy according to real-time traffic data and predefined logic rules. Such approaches rely on the deployment of sensors at the road, which is usually very costly.

Recently, with the fast development of VANETs [10], intersection control via VANETs has become a noteworthy advance. In [11][12], a controller node is placed at the intersection to collect queue length information and compute proper cycle time of traffic signal via the Webster formula. In addition to queue length information,

R.C.-H. Hsu and W. Shangguang (Eds.): IOV 2014, LNCS 8662, pp. 31–41, 2014.

priority of vehicles is considered in [13], and traffic signal is scheduled with quality-of-service provisioning. The concept of "Virtual Traffic Light" is proposed in [14][15], where a vehicle among all the vehicles waiting at intersection is elected as virtual traffic light, which is responsible to schedule all the vehicles in the intersection by sending passing permission to them through V2V communication. These algorithms use VANET mainly as a source of traffic information, and vehicles are controlled in a centralized way.

In our previous work [16], we have proposed a totally different approach, where, vehicles at an intersection can negotiate about the time and order of passing, via V2V communications. More precisely, vehicles need to compete for the privilege of passing via message exchange. Vehicles with conflicting direction need to wait until the winners have passed. A vehicle sends request to others. Then others may reject its request, or permit its preemption. When permissions from others are collected, it can pass the intersection. Since no traffic light facility is necessary, and no optimization of green signal is calculated, such an intersection control approach has two major advantages. 1) It is simple and with low cost because no optimal problem is involved and no centralized facility is necessary. 2) It is efficient and flexible, because the vehicles are directly controlled with real-time information and resources are fully used.

However, how to realize the privilege control in VANET based intersection control is not trivial. The core part is the algorithm to coordinate vehicles. In our previous work [16], we model the problem of VANET based intersection control as the Vehicle Mutual Exclusion for Intersections (VMEI) problem, a variant of the classic mutual exclusion (MUTEX) problem [17], and propose an algorithm to solve the VMEI problem. For the simplicity of presentation, the algorithm proposed in [16] is called MEV, i.e. the mutual excision algorithm for VMEI.

In this paper, we extend our previous work by reducing communication cost. In MEV, each vehicle plays exactly the same role, and participants in the privilege negotiation simply for itself. On the other hand, usually vehicles at the same lane can pass the intersection together upon the grant of one privilege. That is, vehicles may follow its predecessor at the same lane. In such a situation, the following vehicles may not need to fully participant in the privilege competition. Such an observation motivates us to pursue a new design of intersection control algorithm with reduced coordination operations.

In our new design, vehicles are grouped virtually. One group of vehicles will be granted privilege of passing as a whole. Then, only the head vehicle in the group needs to participant in the competition with vehicles at other lanes. The key issue in such a design lies in how to determine and recognize the head of a group. By addressing this issue and others, we design a new algorithm to realize intersection control with low communication cost. The new algorithm is called R-MEV, i.e. MEV with reduced communication cost.

Simulations are conducted to evaluate the performance of our new algorithm. The MEV algorithm is also simulated for comparison purpose. The simulation results show that our new design can significantly reduce communication cost, which confirms the advantage of our work.

The rest of the paper is organized as follows. In Section 2, we describe the system model and preliminaries of our work. Especially, we briefly introduce the VMEI

problem and related definitions. Our new algorithm is presented in Section 3, including message types and detailed operations. Simulation via ns-3 is reported in Section 4, with various performance metrics are measured and analyzed. Finally, Section 5 concludes the paper. Please notice that, the formal proof of our new algorithm is omitted to save space.

2 System Model and Preliminaries

Since this work is based on the algorithm in [16], we adopt the same system model and assumptions. The following definitions are originally proposed in [16].

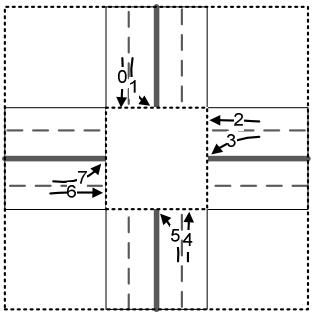

Fig. 1. The Illustration of an Intersection

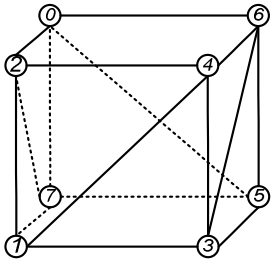

Fig. 2. The Conflict Graph

2.1 System Model and Definitions

1) The Road and Intersection

As shown in Figure 1, we consider an individual intersection with four directions, i.e. north, south, east, and west. In each direction, there are two lanes, for going forward and turning left respectively. For the simplicity of presentation, the lanes are numbered from 0 to 7, and denoted by l_0 to l_7 respectively.

The small dashed rectangle represents the core area of the intersection. A vehicle in this area is called to be "passing" the intersection. The large dashed rectangle represents the queue area. A vehicle in this area is viewed as in the queue to pass the intersection.

Definition 1. *The concurrency/conflict relationship*. According to road rules, vehicles at some pair of lanes may pass the intersection simultaneously. We call such a pair of lanes "concurrent" lanes. On the other hand, if vehicles at two different lanes cannot pass simultaneously, we say they are "conflicting" lanes.

Accordingly, the vehicles at concurrent/conflicting lanes are called concurrent/ conflicting vehicles. The concurrency and conflict relationship is denoted by "$\approx$" and "$\propto$" respectively. For example, we have $l_0 \approx l_4$.

Definition 2. *The conflict graph of lanes, GL.* The relationship among eight lanes can be clearly represented by a conflict graph GL (Fig. 2). The vertices represent the lanes

at an intersection and the edges (dashed and real lines) represent the "conflicts" between lanes. Two lanes without connecting edge are concurrent lanes. GL is exactly a cube with one diagonal line at each side face, as shown in Fig. 2.

Definition 3. *The strong concurrency relationship.* A pair of lanes represented by the two vertices of the same face in the conflict graph is called strong concurrent lanes. The strong concurrency relationship is denoted by "$\cong$". For example, we have $l_2 \cong l_3$.

2) The Vehicle and Wireless Network

Each vehicle has a unique id, which can be the license plate number. The vehicles are AVs driven by autopilots. Also, we assume there are sensors and other necessary devices for collision avoidance and navigation. A vehicle can get the knowledge of its lane number based on the digital map or other methods.

The vehicle is assumed to be able to detect the boundary of the queue/core area when it crosses the boundary. (This can be realized by deploying sensors at the boundary or making use of positioning system like GPS.)

Please notice that we do not assume a vehicle can recognize other vehicles queuing at the intersection by sensing because this is costly and not always feasible.

Wireless communication devices onboard enable vehicles to communicate with each other by sending and receiving messages. The *id* of the vehicle can be used as the address for communication.

We assume that the transmission range of the communicating device is larger than the length of the queue area. That is, the vehicles inside the queue area constitute a one-hop ad hoc network and the each vehicle pair can communicate directly. We assume the wireless channel is FIFO channel. Same as in [16], we do not consider message loss because our work focuses on coordination operations rather than communication protocols.

The procedure of a vehicle passing the intersection can be described by an automaton with three states.

- IDLE: A vehicle is in the idle state if it is out of the queue area. That is, when a vehicle has not entered queue area its state is set to be idle. Also, after a vehicle has passed the intersection (exited the core area), it also becomes "idle".
- WAITING: The state that a vehicle waits for the permit to pass the intersection (i.e. to enter the core area).
- PASSING: The state that a vehicle is moving to pass the intersection. A vehicle is in the passing sate during the time interval between receiving the permit and exiting the core area.

2.2 The VMEI Problem

In the classical mutual exclusion (MUTEX) problem, at most on process can be in the critical section at any moment. All the processes have to compete with each other to get the privilege to access the critical section. This is similar to the traffic control at

the intersection, where the vehicles cannot pass the intersection at the same time and they have to compete to obtain the privilege to access the core area.

On the other hand, the traffic control problem is different from the MUTEX problem because, at the intersection, some lanes can be used simultaneously. That is, not all the vehicles are "competitors" of each other. For example, vehicles at the lane 0 and lane 4 can pass the intersection at the same time.

By considering the new requirements of traffic control, we define a variant of MUTEX, i.e. the problem of Vehicle Mutual Exclusion for Intersections (VMEI).

Definition 4. *The VMEI problem.* Each vehicle at the intersection requests to pass the intersection along the direction as it wants. Accordingly, vehicles queue up at the correct lane when they enter the queue area. To avoid collision or congestion, vehicles can pass the intersection simultaneously if and only if they are in concurrent lanes.

The VMEI problem can be formally defined by correctness properties, which can guarantee that all the vehicles can eventually pass the intersection successfully.

- Safety (*mutual exclusion*): At any moment, if there is more than one vehicle in the core area, they must be concurrent with each other.
- Liveness (*deadlock free*): If there are no vehicles in the core area, some vehicle must be able to enter the core area in a finite time.
- Fairness (*starvation free*): Each vehicle must be able to pass the intersection after a finite number of vehicles do so.

2.3 The MEV Algorithm

To help understanding our new algorithm, we briefly introduce the basic idea of the MEV algorithm in [16]. Please refer to the original paper for details.

Vehicles have different priorities to pass, which is generally determined by the arrival time. A vehicle broadcasts request message and the receivers with higher priority will prevent the sender via response message. If no receivers prevent the sender, it can pass the intersection. We have also designed mechanism to achieve high efficiency by allowing vehicles at concurrent lanes pass simultaneously.

3 The Proposed Algorithm, R-MEV

3.1 Notations and Message Types

Like in MEV, a vehicle undergoes three phases to pass the intersection and correspondingly, there are three states: IDLE, WAITING and PASSING. Each vehicle i maintains the following notations or data structures to execute the distributed algorithm.

- HL_i: High list, the list of vehicles that have a higher priority than i to pass the intersection. The priority is generally, but not always, determined by the arrival time of vehicles.
- LL_i: Low list, the list of vehicles that have a lower priority than i to pass the intersection.

- *CntPmp*: This is a counter to record the number of preemptions occurred at *i*. A preemption means a vehicle allow another with low priority to pass first at some special situation.

Different from MEV, in R-MEV, vehicles play different roles, which is different from MEV. Basically, there are three roles: HEAD, INT, TAIL.

- *HEAD*: The first vehicle of a group, the vehicle that sends REJECT message and FOLLOW message when its group is waiting for passing. If there are no other vehicles in its group, it will also send RELEASE message when it has passed the intersection.
- *TAIL*: The last vehicle of a group, the vehicle that sends REJECT message when its group is passing intersection and RELEASE message when its group has several vehicles.
- *INT*: Vehicles between HEAD and TAIL.

3.2 Algorithm Operations

The pseudo code of the R-MEV is listed below. It basically follows the operations in MEV. The major new operations designed for R-MEV are presented with border.

1) Operations of Request

When vehicle *i* enters the queue area, it switches from IDLE to WAITING and broadcasts a request message REQUEST(*i*, lid_i). Its default role is tail. Then, *i* waits for REJECT messages from others after setting a timeout *tmt*.

Base operations inherited from MEV. When a vehicle *j*, in the WAITING or PASSING state, receives the request message from *i*, a vehicle at some conflicting lane, if some vehicle *k* in *j*'s high list is strong concurrent with *i*, *j* will give way to *i*, i.e. it puts *i* in its high list and will not send REJECT; otherwise, *j* puts *i* in its low list and sends REJECT(*j*, *i*) to *i*. A counter *CntPmp* is used to record how many times a vehicle has been preempted. If the value reaches the threshold *TH*, the vehicle will not give way anymore. The preemption may also cause deadlock if two strong concurrent or conflict vehicles take different actions on the preemption of another vehicle. To avoid such problem, mechanism are designed to coordinate the grant of preemption. When some REJECT(*j*, *k*) is received and *i* has put k to its high list due to preemption, *i* will cancel the preemption of *k*, if a) *j* is a conflicting vehicle and not in *HLi*, b) *j* is strongly concurrent with *i*. (if *j* is in PASSING and it has switched to PASSING by a FOLLOW(*x*, *flt*) message and *j* is not the TAIL in *flt*, or *j* is in WAITING and j is not the HEAD, *j* will not send the REJECT message.) On receiving the REJECT(*j*, *i*) message from *j*, *i* puts *j* in its high list HL_i.

New operations specifically for R-MEV. In operations above, all vehicles at conflicting lanes with response to a REQUEST message, which is in fact not necessary. To avoid such a problem, we divide vehicles into groups and let only the head vehicle of a group handle competitions for privilege of passing the intersection. Other vehicles just need to follow the head and do not need to reply requests. Then, the new algorithm for handling REQUEST is changed as follows.

CoBegin //for a vehicle *i*
On entering the monitoring area:
st_i =WAITING; **role = TAIL;**
broadcast REQUEST(*i*, lid_i);
wait for REJECT from others;
On Receiving REQUEST(*j*, *lid*) from *j*
if((st_i = WAITING|PASSING) and ($lid_i = lid_j \vee lid_i \propto lid_j$)){
if($lid_i = lid_j \wedge role = TAIL$){
if($\exists k, k \in HL_i \wedge lid_i = lid_k$)
role = INT;
else role = HEAD
}
if(($\exists k, k \in HL_i \wedge j \cong k$) and *CntPmp*<*TH*){
add *j* to HL_i;
CntPmp ++;
}**else**{
add *j* to LL_i;
if(role = HEAD $\wedge$ **st_i =WAITING) or**
(role = TAIL $\wedge$ **st_i =PASSING)**
broadcast REJECT(*i*, *j*);
}
}
On Receiving REJECT(*j*, *k*) from *j*
if(st_i=WAITING){
if (*i*=*k*) add *j* to HL_i;
if(*i*≠*k* and($k \in HL_i$ *with preemption*) and
((($j \propto i \wedge j \notin HL_i$) $\vee$ $j \cong i$)){
delete *k* from HL_i;
broadcast REJECT(*i*, *k*);
}
}
On Receiving PERMIT(*j*) or timeout *tmt* occurs (no JECECTs received)
delete *j* from HL_i;
if(HL_i is empty){
st_i=PASSING;
construct the follow list *flt*;
($flt = \{v \mid lid_v = lid_i \wedge v \in LL_i \wedge flt\text{'s } length < NF\}$)
broadcast FOLLOW(*flt*);
}
move and pass the core area;
On Receiving FOLLOW(*j*, *flt*) from *j*
if($i \in flt$){
st_i =PASSING;
move and pass the core area;
}**else if** ($i \propto j$) **{**
delete *j* from HL_i;
delete vehicles in *flt* from HL_i or LL_i;
add the last one in *flt* to HL_i;
}
On exiting the intersection
if(the passing is triggered by a FOLLOW(*x*, *flt*) and *i* is the last in *flt*)
broadcast PERMIT(*i*);
CoEnd

When vehicles *i* and *j* are in the same or conflicting lanes and j is the TAIL, if there exists vehicle *k* which is in the same lanes as *j* and *k* is in *j*'s high list, then *j*'s role turns into INT; otherwise, *j*'s role turns into HEAD. Once roles of vehicles have been identified, our work for reducing reject messages becomes possible and easier. Then if vehicle *k* in *j*'s high list has a strong concurrency relationship with *i*, *j* puts *i* in its high list; otherwise, *j* puts *i* in its low list. And especially when *j* is the HEAD of waiting group or the TAIL of passing group, *j* will send REJECT(*j*, *i*) to *i*.

The new operation divides vehicles into groups, and different vehicles deal with received messages with different responses on the basis of their states and roles. That means when a waiting HEAD vehicle receives the request message from a coming vehicle, it will performance differently from an INT vehicle or a TAIL vehicle. A vehicle can only possess a role at one time and the role can change from time to time as position changes. On the other hand, the correctness of roles' changes ensures the passing of vehicles. To be specific, once TAIL vehicles are in the lanes as coming vehicles, changes of roles occur.

2) Operations of Passing

If no REJECT messages are received before the timeout *tmt* occurs, or the high list becomes empty upon receiving PERMIT messages, i's role should be HEAD, and then *i* switches to be PASSING and starts to pass the core area. If there are other waiting vehicles in the same group (they must be in low list), *i* will add these vehicle to the follow list *flt* and broadcasts FOLLOW(*i*, *flt*).

To avoid starvation, the length of *flt* should be bounded by a threshold *NF*. The value of *NF* can be a static value determined based on the historic data, or a value adaptive to the real time traffic volume. In addition, *NF* means the maximal length of group.

On receiving a FOLLOW(*i*, *flt*) message at *j*, if *j* is included in *flt*, *j* will empty its high list, switch to PASSING and starts to pass the core area of the intersection.

Please notice that, in this case, *j* will not further construct and send FOLLOW message. Such a design is to avoid starvation at other lanes.

If the receiver of FOLLOW(*i*, *flt*), say *w*, is conflicting with *i*, *w* deletes *i* and vehicles in *flt* from its high list or low list, and then puts the last vehicle in *flt* into its high list.

3) Operations of Release

After *i* passes the core area (crosses the exit boundary), *i* will empty its low list. It will also broadcast the PERMIT(*i*) message UNLESS its passing has been triggered by a FOLLOW(*x*, *flt*) message and *i* is not the last in the *flt*.

On receiving the PERMIT(*i*) message at *j*, if *i* is in the high list of *j*, *j* deletes *i* from its high list. If the high list becomes empty, it will switch to PASSING.

4 Performance Evaluation

Simulations are conducted using ns-3. Besides R-MEV, we also simulated two other algorithms for comparison. MEV is the most similar one, and C-MEV is the centralized version of MEV proposed in [16]. We basically follow the system settings in [16]. The area of intersection is 100m×100m. IEEE 802.11 is adopted as the communication protocol, with a transmission range of 200m. The time for a vehicle to pass the intersection is set to be 3s (for going straight), and 4s (for turning left).

Same as in [16], we use four metrics to measure the performance of intersection control algorithms. Basically, R-MEV achieves similar waiting time, queue length, and system throughput as MEV, while R-MEV costs much fewer messages than MEV does. Due to page limit, we report here only the results of message cost and results of other metrics are omitted.

As shown in Fig. 3, the message costs of the three intersection algorithms differ a lot. With the centralized algorithm, a vehicle needs roughly three messages per passing. On the other hand, the distributed algorithms cost more messages. Such message costs are reasonable and the difference is expected. In the centralized algorithm, one vehicle needs communicate with only the central node and at most three messages (one REQUEST, one PERMIT and one RELEASE) are sent for each pass of a vehicle. However, the distributed algorithm needs to send more messages, especially REJECT messages, due to the lack of a centralized control node.

Compared with MEV, R-MEV can significantly reduce message cost, in various traffic cases. In MEV, upon receiving a REQUEST message, almost all waiting vehicles in conflicting lanes will send REJECT messages. In R-MEV, with the new design, only head vehicles need to send REJECT. Since REJECT messages account for a large part of message cost, R-MEV can save a lot of communication. Fig. 3 shows that the reduction can be as much as 50%. This definitely confirms our objective to design R-MEV is well achieved.

(a) Uniform volume (b) Non-uniform volume

Fig. 3. Message cost of intersection control algorithms

5 Conclusions

In this paper, we propose a new algorithm for intersection control via V2V communications. Vehicles at an intersection obtain the privilege of passing by competing with other vehicles. Such an approach is more efficient than traditional ones based on traffic light signals. Compared with our previous work in [16], this paper focuses on reducing communication cost. By letting only the head vehicle in a group handle reply passing requests, the new algorithm can save message cost as much as 50%.

Acknowledgments. This research is partially supported by National Natural Science Foundation of China (No. 61379157), Guangdong Natural Science Foundation (No. S2012010010670), and Pearl River Nova Program of Guangzhou (No. 2011J2200088).

References

1. Day, I.: Scoot-split, cycle and offset optimization technique. TRB committee AHB25 adaptive traffic control (1998)
2. Bingham, E.: Reinforcement learning in neurofuzzy traffic signal control. Eur. J. Operat. Res. 131, 232–241 (2001)
3. Chen, X., Shi, Z.: Real-coded genetic algorithm for signal timings optimization of a signal intersection. In: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 3, pp. 1245–1248. IEEE Press, Beijing (2002)
4. Gokulan, B.P., Srinivasan, D.: Distributed geometric fuzzy multiagent urban traffic signal control. In: 13th International IEEE Conference on Intelligent Transportation Systems, vol. 11(3), pp. 714–727. IEEE Press, Madeira Island (2010)
5. Zhao, D., Dai, Y., Zhang, Z.: Computational Intelligence in Urban Traffic Signal Control: A Survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(4), 485–494 (2012)
6. Zhao, L., Peng, X., Li, L., Li, Z.: A fast signal timing algorithm for individual oversaturated intersections. In: 14th International IEEE Conference on Intelligent Transportation Systems, vol. 12(1), pp. 280–283. IEEE Press, Washington (2011)
7. Tubaishat, M., Zhuang, P., Qi, Q., Shang, Y.: Wireless sensor networks in intelligent transportation systems. Wirel. Commun. Mob. Comput. 9, 287–302 (2009)
8. Lee, J., Park, B.: Development and Evaluation of a Cooperative Vehicle Intersection Control Algorithm Under the Connected Vehicles Environment. In: 15th International IEEE Conference on Intelligent Transportation Systems, vol. 13(1), pp. 81–90. IEEE Press, Anchorage (2012)
9. Zhou, B., Cao, J., Zeng, X., Wu, H.: Adaptive Traffic Light Control in Wireless Sensor Network-Based Intelligent Transportation System. In: IEEE 72nd Vehi. Tech. Conf. Fall, pp. 1–5. IEEE Press, Ottawa (2010)
10. Hartenstein, H., Laberteaux, K.P.: A tutorial survey on vehicular ad hoc networks. IEEE Comm. Mag. 46(6), 164–171 (2008)
11. Gradinescu, V., Gorgorin, C., Diaconescu, R., et al.: Adaptive traffic lights using car-to-car communication. In: IEEE 65th Vehi. Tech. Conf. Spring, pp. 21–25. IEEE Press, Dublin (2007)
12. Prashanth, L.A., Bhatnagar, S.: Reinforcement learning with function approximation for traffic signal control. In: 14th International IEEE Conference on Intelligent Transportation Systems, vol. 12(2), pp. 412–421. IEEE Press, Madeira Island (2011)
13. Wunderlich, R., Liu, C., Elhanany, I., et al.: A novel signal-scheduling algorithm with quality-of-service provisioning for an isolated intersection. In: 13th International IEEE Conference on Intelligent Transportation Systems, vol. 9(3), pp. 536–547. IEEE Press, Beijing (2008)

14. Ferreira, M., Fernandes, R., Conceicao, H., et al.: Self-organized traffic control. In: Proceedings of the Seventh ACM International Workshop on VehiculAr InterNETworking, pp. 85–90. ACM Press, New York (2010)
15. Ferreira, M., d'Orey, P.M.: On the impact of virtual traffic lights on carbon emissions mitigation. In: 15th International IEEE Conference on Intelligent Transportation Systems, vol. 13(1), pp. 284–295. IEEE Press, Anchorage (2012)
16. Wu, W.G., Zhang, J.B., Luo, A.X., Cao, J.N.: Distributed Mutual Exclusion Algorithms for Intersection Traffic Control. IEEE Transactions on Parallel and Distributed Systems. Preprint, online version (2014)
17. Lamport, L.: A Fast Mutual Exclusion Algorithm. ACM Trans. Comput. Syst. 5(1), 1–11 (1987)