Scheduling Soft Aperiodic Messages on FlexRay in-Vehicle Networks

Menglan Hu
School of Computer Engineering
Nanyang Technological University
Singapore
mlhu@ntu.edu.sg

Yang Wang
Faculty of Computer Science
University of New Brunswick
Fredetricton, Canada, E3B 5A3
ywang8@unb.ca

Wei Shi
Faculty of Business and I.T.
University of Ontario Institute of Technology
Canada, L1H 7K4
wei.shi@uoit.ca

***Abstract*—The FlexRay communication protocol is expected to be the de-facto technique standard for the next generation high-speed networks on vehicles. A number of recent studies has thus investigated message scheduling techniques for FlexRay systems. However, most existing work focused on either the scheduling of periodic messages on the static segment or the scheduling of hard aperiodic messages on the dynamic segment while soft aperiodic messages have been neglected. Also, isolated scheduling severely limits the overall performance of all messages including periodic, hard aperiodic and soft aperiodic messages in terms of bandwidth utilization and transmission latency. In order to address these aspects, this paper presents an algorithm, referred to as *Joint Scheduling Algorithm for FlexRay* (JSAF), which jointly schedules soft aperiodic messages together with periodic and hard aperiodic messages in real-time FlexRay systems. The algorithm prioritizes periodic messages and hard aperiodic messages and first schedules them onto the static segment and the dynamic segment, respectively. Soft aperiodic messages are then dynamically scheduled with an online scheduler by utilizing unused time left by periodic messages and hard aperiodic messages. Performance evaluation results are presented to demonstrate the effectiveness and competitiveness of our approaches when compared to existing algorithms.**

***Index Terms*—In-vehicle networks, FlexRay, real-time scheduling, message scheduling, online algorithm, slack stealing.**

I. Introduction

In today's vehicles, a great number of electronic devices including micro sensors, controllers, and actuators, are used to replace mechanical and hydraulic devices. These electronic control units (ECU) require message communicaiton among each other via in-vehicle networks to support task processing. In today's automobiles up to 70 ECUs exchange up to 2500 messages [2], [3].

The FlexRay protocol [1] is a communication standard developed by the FlexRay consortium. It is expected to be the next generation criteria in the automotive industry. The protocol comprises a static segment with Time Division Multiple Access (TDMA) operation, and a dynamic segment with flexible TDMA (FTDMA) operation. As a result, it combines the advantages of time-triggered and event-triggered communication. The FlexRay protocol has been implemented in the major line of new cars. For example, the new BMW-7 series are deployed with FlexRay-based brake system [4].

The message scheduling on the FlexRay communicaiton bus is a critical issue for offering quality-of-service (QoS) guarantees to time-critical applications on in-vehicle networks. However, most existing work focused on either the scheduling of periodic messages on the static segment [9], [13], [10] or the scheduling of hard aperiodic messages on the dynamic segment [7], [8] while soft aperiodic messages have been neglected. Also, the isolated scheduling on either periodic messages or hard aperiodic messages severely limits the overall performance of all messages including periodic, hard aperiodic and soft aperiodic messages in terms of bandwidth utilization and transmission latency.

A recent work on soft aperiodic message scheduling on FlexRay systems is [11], which proposed a scheduling algorithm called HOSA. HOSA adopted "slot pilfering" for scheduling soft aperiodic messages by stealing slots from periodic messages in the static segment. Hence slot pilfering can be regarded as a variant of slack stealing [18]. However, HOSA violates the rules of FlexRay systems, which severely restricts the feasibility of the algorithm. Firstly, since there is no identifiers for static slots in the static segment, to dynamically schedule soft aperiodic messages, we need additional schemes to identify different periodic and soft aperiodic messages at runtime. Secondly, in FlexRay systems a host can transmit messages in a static slot only if it holds the frame ID of the slot. Nevertheless, both constraints have been neglected in [11]. In contrast, our paper proposes to effectively address the points in our algorithm. In addition, we design scheduling policies which effectively select suitable soft aperiodic messages to optimize response time in both the static and dynamic segments, while in [11] such kind of scheduling polices are absent.

To this end, we investigate the problem of jointly scheduling periodic, hard aperiodic and soft aperiodic messages on the FlexRay bus. The goal of the scheduling problem is to guarantee that the deadlines of the periodic and hard aperiodic messages are satisfied while the average response time of the soft aperiodic messages is minimized.

We contribute an algorithm, referred to as *Joint Scheduling Algorithm for FlexRay* (JSAF), which jointly schedules soft aperiodic messages together with periodic and hard aperiodic messages on FlexRay systems. As prior studies have provided solutions for scheduling periodic messages and hard aperiodic messages [9], [7], this paper pays more attention on soft aperiodic messages. Traditionally, the static segment is used

978-1-4799-2084-6/13/$31.00 ©2013 IEEE

for scheduling periodic messages. However, we argue that by adopting slack stealing to schedule soft aperiodic messages onto the unused static slots left by periodic messages, the static segment can be more efficiently utilized. For the dynamic segment, soft aperiodic messages are also scheduled in the slack time left by hard periodic messages. Hence, the algorithm prioritizes periodic messages and hard aperiodic messages and first schedules them onto the static segment and the dynamic segment, respectively. Soft aperiodic messages are then dynamically scheduled with an online scheduler by utilizing unused time left by periodic messages and hard aperiodic messages.

The remainder of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces models, assumptions, and problem formulation. Sections 4 and 5 describe the JSAF algorithm in great detail. Section 6 presents simulation results to evaluate the algorithm, with conclusions following in Section 7.

II. Related Work

Existing studies mainly consider isolated message scheduling on FlexRay bus. In [9] and [7], the message scheduling problems were solved via nonlinear integer programming (NIP) for static segment and dynamic segment, respectively. In particular, [9] applied a signal packing technique which packs multiple periodic signals into a message; [7] proposed to reserve slots for hard aperiodic messages so that flexible medium access of the dynamic segment is preserved while QoS assurance can also be guaranteed. Both papers formulated NIP and decomposed the NIP problems into ILP problems. Another work [10] transformed the message scheduling problem for the static segment into a bin packing problem and again applied ILP to solve it. Nevertheless, this isolated scheduling severely limits the overall performance of all messages including periodic, hard aperiodic and soft aperiodic messages in terms of bandwidth utilization and transmission latency. Also, these studies neglected the scheduling of soft aperiodic message. In contrast, our paper comprehensively studies the joint scheduling of periodic, hard aperiodic and soft aperiodic messages.

A recent study [11] proposed a scheduling algorithm called HOSA, which adopted slack stealing [18] for scheduling soft aperiodic messages by stealing slots from periodic messages in the static segment. The slack stealing technique [18], [19] attempted to service aperiodic task or reduce energy consumption by stealing processing time from periodic and hard aperiodic tasks. However, as we discussed in Section 1, HOSA violates the rules of FlexRay systems, which severely restricts the feasibility of the algorithm. In contrast, our paper proposes effective solutions that obey the rules of FlexRay specification to realize slack stealing for joint message scheduling on FlexRay bus.

Besides, a number of researches have been performed from other perspectives for FlexRay systems. Park et al. [14] proposed a FlexRay network parameter optimization method which can determine the lengths of the static slot and the communication cycle. Another work [17] analyzed the end-to-end

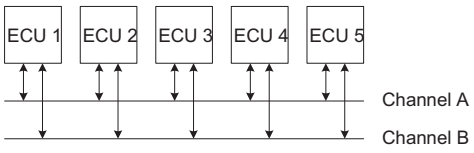


Fig. 1. A FlexRay Cluster

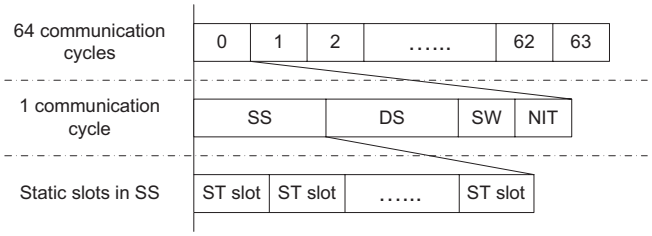


Fig. 2. FlexRay timing hierarchy.

performance of a set of tasks running on different ECUs. A few studies [15], [16] suggested jointly scheduling both tasks and messages on FlexRay systems. Their studies are promising, but are still immature with only integer programming formulations.

III. Problem Formulation

The underlying platform is a FlexRay cluster consisting of a set of hosts (i.e., ECUs), which are connected via FlexRay communication channels, as shown in Fig. 1. Let $H = \{H_1, H_2, \cdots, H_N\}$ denote a list of ECUs in the system. Each ECU H_j $(j = 1, 2, ..., N)$ can process particular tasks which exchange data via messages transferred on the FlexRay channels. FlexRay provides two channels (i.e., Channel A and Channel B shown in Fig. 1), each with a high bandwidth of 10 Mb/s.

The FlexRay protocol [1] is a time-triggered protocol. Its operation is based on repeatedly executed FlexRay cycles with a fixed duration. Fig. 2 shows the timing hierarchy of 64 FlexRay cycles. A FlexRay cycle comprises a static segment (SS), a dynamic segment (DS), a symbol window (SW), and the network idle time (NIT), as shown in Fig. 2.

The static segment and the dynamic segment demonstrate different formats and functionalities in the communication slots. Similar to the Time Triggered Protocol (TTP) in [5], the organization of the static segment is based on a time-division multiple access (TDMA) scheme. It consists of a fixed number of equal size static slots. According to the FlexRay specification, each static slot in each channel of each cycle can only be uniquely assigned to one ECU to transfer one message. Fig. 2 illustrates the time hierarchy of the static segment.

The dynamic segment is similar to ByteFlight [6] and employs the flexible TDMA (FTDMA) approach. Fig. 3 the timing diagram of the DS. The smallest time unit in the dynamic segment is the mini-slot with a duration of T_{MS}, and the dynamic segment contains a fixed number of mini-slots (between 0 and 7986). The dynamic segment consists of consecutive dynamic slots that are superimposed on mini-slots. While all static slots are of equal size, FlexRay allows the duration of a dynamic slot to vary depending on the length

of the frame transmitted in the dynamic slot. That is, in each dynamic slot, a frame with the corresponding frame ID is transmitted, and hence the duration of the dynamic slot is determined by the length of the transmitted frame. Hence, a dynamic slot can contain one or more mini-slots. Once a message is transmitted in a dynamic slot, the length of the dynamic slot is equal to the number of mini-slots needed for message transmission. Otherwise, the duration of the dynamic slot is one mini-slot. In an example shown in Fig. 3, dynamic slots are marked as numbers and messages are marked as letters. We can observe that dynamic slots 1, 3, and 4 are not used and thus last only 1 mini-slot while 2 and 5 are used to send messages a and b.

The scheduling on message delivery depends on the management of the frame ID. Each frame to be transmitted in a cluster has a frame ID assigned to it. A frame ID indicates the slot in which the frame should be transmitted. Each slot in each FlexRay cycle with its corresponding frame ID is uniquely allocated to a host, where multiple frame IDs can be assigned to each host. FlexRay distinguishes the frame IDs among slots in different channels and different cycles. A frame ID is used no more than once in each channel in a communication cycle. FlexRay also distinguishes the frame IDs between static and dynamic segments. Each ECU maintains one slot counter for each segment to follow the progress of static segment and dynamic segment. At the beginning of each segment in each cycle, a slot counter is initialized to 1 and is then incremented. The arbitration procedure ensures that only the frame with a frame ID that equals the current value of the slot counter can be transmitted.

We assume that the lengths of static segment T_{SS}, static slot T_s, dynamic segment T_{DS}, mini-slot T_{MS}, and the communication cycle T_c are known to the scheduler beforehand as previous papers have thoroughly studied how to choose proper values for these parameters [14], [9], [7].

We consider jointly scheduling periodic, hard aperiodic and soft aperiodic messages on FlexRay systems. Each host H_j has a set of periodic messages, a set of hard aperiodic, and a set of soft aperiodic messages to transmit. For the scheduling of periodic messages and hard aperiodic messages, we can utilize the approaches provided by prior studies. Hence this paper pays more attention on soft aperiodic messages. Let M_j be the set of soft aperiodic messages owned by host H_j. A message m_i^j in M_j has a minimal inter-arrival time p_i^j, which characterizes the minimum time interval between two consecutive message generations. Periodic messages with fixed periods are scheduled in the SS and hard aperiodic messages with minimal inter-arrival times are scheduled in the DS. By applying slack stealing, soft aperiodic messages are dynamically scheduled in both segments. The objective of the problem is to jointly schedule all the messages such that the deadlines of the periodic and hard aperiodic messages are satisfied while the average response time of the soft aperiodic messages is minimized.

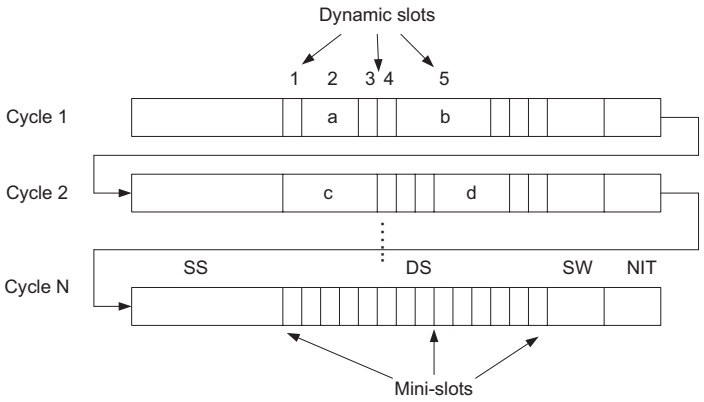


Fig. 3. Timing diagram of the dynamic segment.

IV. Scheduling in Static Segment

The algorithm prioritizes periodic messages and hard aperiodic messages and first schedules them onto the static segment and the dynamic segment, respectively. Soft aperiodic messages are then dynamically scheduled by utilizing unused time left by periodic messages and hard aperiodic messages. This section first describes the design for the static segment while the design for the dynamic segment will be presented in the next Section. The scheduling for the static segment consists of two phases, the offline preprocessing phase, which statically allocates static slots to hosts, and the online scheduling phase, which dynamically schedules periodic messages and soft aperiodic messages at runtime.

Traditionally, the static segment is used for scheduling periodic messages. However, we argue that by adopting slot stealing to schedule soft aperiodic messages onto the unused static slots left by periodic messages, the static segment can be more efficiently utilized. There exist two opportunities for implementing slot stealing. Firstly, since not all static slots are allocated to periodic messages, there may be some idle slots that can be used by soft aperiodic messages. Secondly, since faults may frequently happen, some periodic messages may not become ready on time and accordingly their allocated slots will be wasted. Hence, if some scheduled periodic messages are not ready on time due to faults, their slots can be used by online schedulers for aperiodic soft real-time messages. This mechanism also conforms to the fault-tolerant property of FlexRay.

A. Offline Preprocessing

Since a static slot can only be used by the host with the corresponding frame ID, in the preprocessing phase we consider the allocation of static slots. We first allocate static slots (i.e., frame IDs) to periodic messages according to a static schedule generated by some static scheduling algorithm, e.g., solutions provided in [9], [10]. Since not all static slots are allocated to the periodic messages, remaining slots may be used by soft aperiodic messages. In FlexRay systems, a host can send a message in a slot only if it holds the frame ID of the slot. Since frame IDs are statically allocated,

to use the remaining slots to send aperiodic messages, we should statically allocate the remaining slots (i.e., frame IDs) to hosts owning soft aperiodic messages. We give the hosts with "frequent" soft aperiodic messages high priorities, i.e., we order the hosts according to the descending order of their frequency parameter f_js, which are defined as:

$$f_j = \sum_{i \in M_j} \frac{1}{p_i^j} \quad (1)$$

We then allocate remaining frame IDs to the hosts in a weighted round robin manner, i.e., we sequentially allocate all the remaining frame IDs. For each frame ID, we allocate it to the host with the largest proportion of f_j to the number of frame IDs assigned for soft aperiodic messages. This process is repeated until all remaining frame IDs have been allocated.

B. Online Scheduling

In online scheduling, soft aperiodic messages can be opportunistically transmitted in the static segment. If a static slot is not assigned to any periodic message in the preprocessing phase, the host can dynamically send soft aperiodic messages. Otherwise, if the slot is statically assigned to a periodic message, and the message will be sent in the slot when it periodically arrives. Only if the message does not arrive on time due to faults, the host owning the slot will have opportunities to dynamically schedule a soft aperiodic message in the slot which otherwise will be wasted. If we enable the online scheduler to transmit soft aperiodic messages in the static slots where periodic messages do not arrive due to faults, we need to introduce a 2-byte message ID for periodic messages. Notice that the standard FlexRay specification requires no online scheduler and no identifiers for periodic messages because in each static slot only a statically determined periodic message will be sent. By default the message ID is only used in aperiodic messages. Therefore, to steal slots from slots which are statically allocated to periodic messages, we append a message ID to each periodic message. Then receivers can correctly identify different periodic messages and soft aperiodic messages transmitted in static slots which are statically allocated to periodic messages. As the length of each static slot is up to 128 bytes, the overhead of this message ID is acceptable.

The natural policy for a host to transmit soft aperiodic messages is First-in-First-Out. An alternative policy for efficient bandwidth utilization is to first schedule the largest soft aperiodic message which can be sent in the slots since a slot can only send one message.

V. Scheduling in Dynamic Segment

The scheduling of hard aperiodic messages in the dynamic segment has been addressed in [7], dynamic slots are reserved for hard aperiodic messages so that flexible medium access of the dynamic segment is preserved while QoS assurance can also be guaranteed. Since [7] offered QoS guarantees at the cost of reserving slots by assuming messages' minimum inter-arrival times as periods, their method may lead to poor resource utilization. In this case, we propose to utilize unused time in the dynamic segment for scheduling soft aperiodic messages. The design for the dynamic segment consists of two phases, the offline preprocessing phase, which statically allocates dynamic slots to hosts, and the online scheduling phase, which dynamically schedules hard and soft aperiodic messages at runtime.

A. Offline Preprocessing

Similar to the preprocessing phase for the static segment, in the preprocessing phase for the dynamic segment, we consider the allocation of dynamic slots. After assigning enough dynamic slots to hard aperiodic messages, remaining slots (frame IDs) are allocated to the hosts owning soft aperiodic messages. The dynamic segment of each cycle contains many mini-slots. If no message is transmitted, the duration of a dynamic slot is equal to that of one mini-slot. Hence, if no hard aperiodic message is sent in the whole dynamic segment, considerable mini-slots become unused.

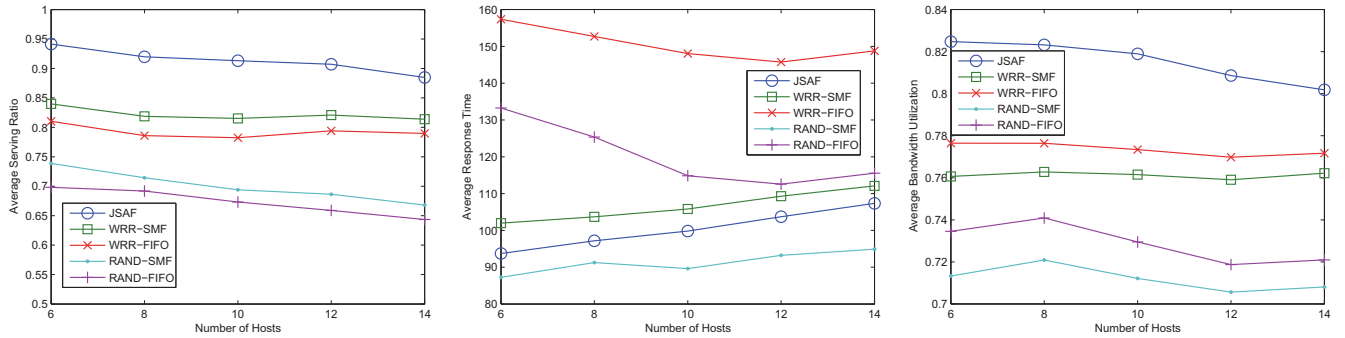
To efficiently utilize the dynamic segment, we allocate the remaining frame IDs left by hard aperiodic messages to the hosts for transmitting soft aperiodic messages. Since the duration of each dynamic slot may vary at runtime depending on messages transmitted, the number of dynamic slots that may appear in a cycle also varies. Even if no message arrives to be transmitted, the slot counter keeps incrementing until the maximum number of mini-slots is reached or the dynamic segment expires. To exhaustively utilize all potential idle mini-slots and deal with the special cases that no hard aperiodic messages arrive, we should allocate enough number of frame IDs to the hosts. Hence we allocate remaining frame IDs to the hosts in a weighted round-robin way, i.e., we sequentially allocate all the remaining frame IDs. For each frame ID, we allocate it to the host with the largest proportion of f_j to the number of assigned frame IDs assigned for soft aperiodic messages. This process repeats until a number equalling to the maximum number of mini-slots of a dynamic segment is allocated, or a predefined maximum number of Frame IDs is reached.

B. Online Scheduling

In online scheduling, in each cycle soft aperiodic messages can be opportunistically scheduled in their slots after hard aperiodic messages are scheduled first. When the slot counter is equal to a dynamic slot, the host owning the slot can dynamically send a soft aperiodic message among all arriving soft aperiodic messages. We implement three scheduling policies for selecting messages for each host which can send a soft message. The first is the First-In-First-Out policy. The second policy selects the soft message with the largest length. The third policy selects the soft message with the smallest length.

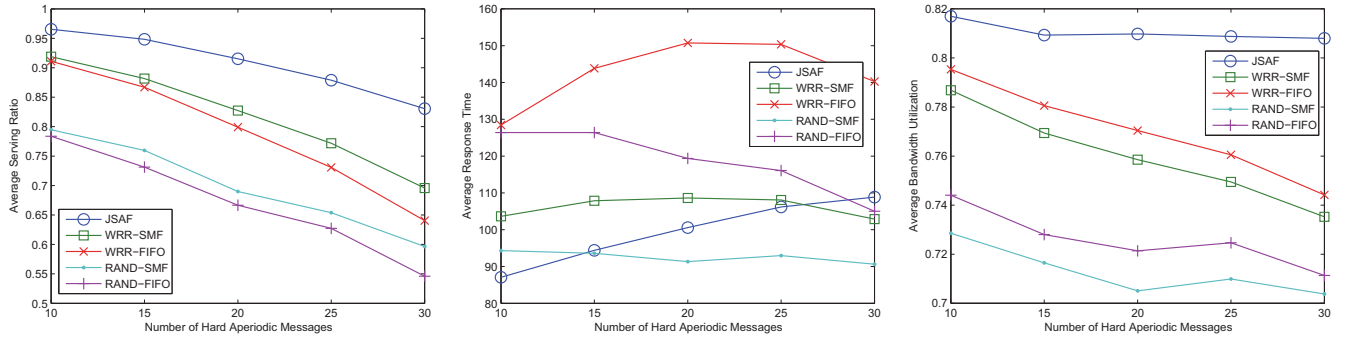
VI. Performance Evaluation

In order to assess the effectiveness of the proposed scheduling algorithm, we will now present a performance evaluation study, carried out by means of a discrete-event simulator. In



(a) Serve ratio versus the number of hosts (b) Response time (in ms) versus the number of hosts (c) Bandwidth utilization versus the number of hosts

Fig. 4. Results with the number of hosts varied.



(a) Serve ratio versus the number of hard aperiodic messages (b) Response time (in ms) versus the number of hard aperiodic messages (c) Bandwidth utilization versus the number of hard aperiodic messages

Fig. 5. Results with the number of hard aperiodic messages varied.

our experiments, we schedule a number of mixed messages including periodic, hard aperiodic and soft aperiodic messages on FlexRay systems.

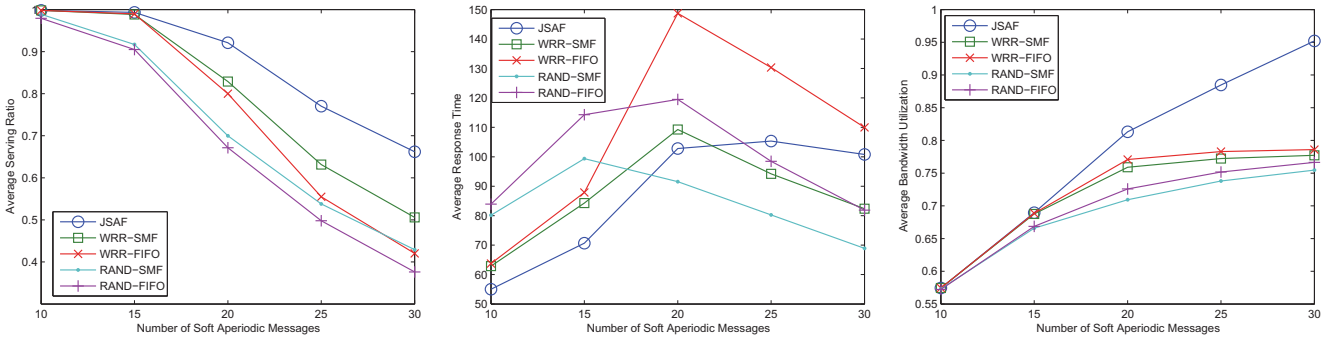
We are interested in three performance metrics. The first metric is average serving ratio, which is defined as the ratio of the number of soft aperiodic messages successfully transmitted to the number of soft aperiodic messages arriving for scheduling. The second metric is average normalized response time, which is defined as the response time of soft aperiodic messages that are successfully transmitted of an algorithm over that of JSAF. The third metric is average bandwidth utilization

Our JSAF algorithms adopts slack stealing scheme for the message scheduling in the static segment. In the dynamic segment, JSAF uses a weighted-round-robin (WRR) policy for offline (i.e., pre-processing) allocating slots to hosts and applies a smallest message first (SMF) policy for online scheduling soft aperiodic messages. Hence, to understand the merits of our algorithms, we compare JSAF with 4 baseline algorithms without slack stealing: WRR-SMF, WRR-FIFO, RAND-SMF, and RAND-FIFO. The nomenclature of the algorithms includes two parts. The first part represents offline slot allocation scheme adopted: WRR and a random allocation policy (RAND). The second part represents online message scheduling policy adopted: SMF and FIFO.

The default simulation configurations are set as follows: Our experiments are performed on 10 ECUs which are connected to a FlexRay bus. We uniformly distribute the messages in the ECUs. The synthetic test cases were generated by randonly varying message parameters, such as periods and deadlines, to cover a wide range of possible scenarios. The message set comprises 20 periodic messages, 20 hard aperiodic messages, and 20 soft aperiodic messages. The periods (inter-arrival times) and deadlines of the periodic (hard aperiodic) messages are varied between 1 to 8 cycles. The communication cycle duration is chosen as 10ms and the static cycle length is 5ms, based on the experiences from the industry [10]. The length of a static slot is 40 macrotick and the number of slots per static segment is 50. The length of a mini-slot is 8 macrotick and the maximum number of dynamic slots per dynamic segment is 200.

In each of the following experiments we vary one interested parameter while fixing other parameters as their initial values to study the effect of the interested parameters. As a first case, we vary the number of hosts in the range [6,14] and Fig. 4 depicts the results, i.e., average serve ratio, average response time, and average bandwidth utilization versus the number of hosts, respectively. We then vary the number of hard aperiodic messages from 10 to 30. Fig. 5 shows the corresponding serve ratio, response time, and bandwidth utilization, respectively. Finally we vary the number of hard aperiodic messages from 10 to 30 and and Fig. 6 depicts the results.

The results in Fig. 4, 5, and 6 show that JSAF constantly



(a) Serve ratio versus the number of soft aperiodic messages (b) Response time (in ms) versus the number of soft aperiodic messages (c) Bandwidth utilization versus the number of soft aperiodic messages

Fig. 6. Results with the number of soft aperiodic messages varied.

outperforms other 4 algorithm combinations by significant margins on serving ratios and bandwidth utilization. This demonstrates the effectiveness of slack stealing. Specifically, Fig. 4 shows that the superiority of JSAF is quite stable with the number of hosts varied, which demonstrates the scalability of our algorithm. In addition, Fig. 5(b) and 6(b) show that when there are less input messages, JSAF also outperforms other 4 algorithm combinations on response time. When the number of messages increases, JSAF still maintains high serving ratios, though in the expense of sacrificing performance on response time. Accordingly, although RAND-SMF delivers better performance than JSAF on response time in some cases, yet JSAF can serve much more messages. Moreover, we can observe from the figures that WRR constantly outperforms RAND on serving ratios and response time. This demonstrate the efficiency of our design in the WRR offline slot allocation scheme. Further, the results show that SMF constantly outperforms FIFO on serving ratios and response time, which provides guidance in choosing online message scheduling policies. That is, for performance consideration we can choose SMF and for fairness consideration choose FIFO.

VII. Conclusions

In this paper we have studied an important joint scheduling problem for handling periodic, hard aperiodic and soft aperiodic messages on real-time FlexRay systems. We have proposed the JSAF algorithm, which is shown to elegantly handle the three types of messages so that the deadlines of the periodic and hard aperiodic messages are satisfied while the average response time of the soft aperiodic messages is minimized. The algorithm prioritizes periodic messages and hard aperiodic messages and first schedules them onto the static segment and the dynamic segment, respectively. Soft aperiodic messages are then dynamically scheduled by utilizing unused time left by periodic messages and hard aperiodic messages. Extensive simulation results with various test configurations have demonstrated the effectiveness and competitiveness of our algorithm.

References

[1] "The flexray communication system specification, version 3.0.1," http://www.flexray.com.

[2] A. Albert, "Comparison of event-triggered and time-triggered concepts with regard to distributed control systems," in *Embedded World*, 2004.

[3] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proc. IEEE*, vol. 93, pp. 1204-1224, 2005.

[4] BMW brake system relies on FlexRay, "http://www.automotivedesignline.com/news/218501196," July 2009.

[5] H. Kopetz and G. Bauer, "The time-triggered architecture," *Proc. IEEE*, vol. 91, no. 1, pp. 1120-126, 2003.

[6] J. Berwanger, M. Peller, and R. Griegbach. (1999) A new high-performance data bus system for safety related applications. [Online]. Available: http://www.byteflight.com/specificati

[7] E. Schmidt and K. Schmidt, "Message scheduling for the flexray protocol: The dynamic segment,"*IEEE Trans. Vehicular Technology*, vol. 58, no. 5, pp. 2160-2169, 2009.

[8] R. Shneider, U. Bordoloi, D. Goswami, and S. Chakraborty. "Optimized schedule synthesis under real-time constraints for the dynamic segment of FlexRay," in *IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)*, 2010.

[9] K. Schmidt and E. Schmidt, "Message scheduling for the flexray protocol: The static segment," *IEEE Trans. Vehicular Technology*, vol. 58, no. 5, pp. 2170-2179, 2009.

[10] M. Lukasiewycz, M. Glab, J. Teich, and P. Milbredt, "Flexray schedule optimization of the static segment," *Proc. CODES+ISSS*, 2009.

[11] Y. Hua, X. Liu, and W. He, "HOSA: Holistic scheduling and analysis for scalable fault-tolerant FlexRay design," in *Proc. IEEE INFOCOM*, 2012.

[12] SAE, "Class C Application Requirements, SAE J2056/1," *SAE Handbook, Soc. Automotive Engineers, Warrendale, PA*, vol. 2, pp. 23.366C 23.371, June, 1993.

[13] B. Tanasa, U.D. Bordoloi, P. Eles, and Z. Peng, "Scheduling for fault-tolerant communication on the static segment of FlexRay," In *Proc. RTSS* 2010.

[14] I. Park and M. Sunwoo, "FlexRay network parameter optimization method for automotive applications," *IEEE Trans. Industrial Electronics*, vol.58, no. 4, pp. 1449-1459, APR. 2011.

[15] M. Lukasiewycz, R. Schneider, D. Goswami, and S. Chakraborty, "Modular scheduling of distributed heterogeneous time-triggered automotive systems," In *Proc. ASP-DAC*, 2012.

[16] M. Lukasiewycz et al., "Schedule integration for time-triggered systems," In *Proc. ASP-DAC*, 2013.

[17] A. Hagiescu, U.D. Bordoloi, S. Chakraborty, P. Sampath, P. Vignesh, V. Ganesan, and S. Ramesh, "Performance analysis of FlexRay-based ECU networks," In *Proc. Design Automation Conf.*, 2007.

[18] J. P. Lehoczky and S. Ramos-Thuel, "An optimal algorithm for scheduling soft-aperiodic tasks in fixed-priority preemptive systems," in *Proc. Real-time Systems Symp.*, pp. 110-123, Dec. 1992.

[19] J. Luo, and N.K. Jha. "Power-conscious joint scheduling of periodic task graphs and aperiodic tasks in distributed real-time embedded systems," in *Proceedings of the 2000 IEEE/ACM international conference on Computer-aided design*, 2000.